Effective Multi-Modal and Multi-Domain Graph-Based Recommender Systems via Self-Supervised Learning

ZIXUAN YI

Submitted in fulfilment of the requirements for the
Degree of Doctor of Philosophy

School of Computing Science
College of Science and Engineering
University of Glasgow

November 2025

Abstract

The rapid expansion of digital applications and services has created an urgent need for sophisticated top-K recommendation models capable of matching users with items aligned to their interests. Graph-based recommender systems have become a cornerstone for tackling this information overload. However, their effectiveness is often hindered by fundamental limitations, including vulnerability to noisy interactions, the so-called over-smoothing problem, and an inability to sufficiently leverage rich and complex information sources. In particular, many existing models underperform in effectively mining and integrating supervisory signals from diverse modalities (e.g., text, images) and domains (e.g., books, movies). This thesis argues that the top-K recommendation effectiveness of graph-based recommendation can be enhanced by proposing novel Self-Supervised Learning (SSL) techniques designed to explicitly mine and integrate multi-modality and multi-domain signals.

This thesis aims to address existing research gaps in the literature by proposing a suite of novel graph-based recommender techniques using the SSL paradigm. It addresses three primary challenges: (i) enhancing the fundamental expressive power of graph neural architectures to alleviate the over-smoothing problem (i.e., representations becoming indistinguishable after repeated graph aggregation operations), and the incapability of existing approaches to denoise noisy implicit interactions in top-K recommendation; (ii) enhancing the modality encoding capabilities to address the insufficient modality fusion and the isolated multi-modal recommendation pipeline in top-K multi-modal recommendation; and (iii) improving the knowledge transfer capability and generalisability of graph-based recommender systems for top-K recommendation in multi-domain settings.

To enhance the expressiveness of graph neural architectures, we propose two novel graph-based recommender models at different architectural levels. First, Positional Graph Contrastive Learning (PGCL) operates at the message-passing level and integrates graph positional encodings (e.g., Laplacian eigenvector) into a new graph message-passing function. This architecture, trained with an SSL loss, generates highly distinguishable user and item embeddings, thereby improving the expressive power of graph-based recommender systems while alleviating the over-smoothing problem. Second, the Diffusion Graph Transformer (DiffGT) leverages a new graph transformer model at the overall architecture level to denoise the noisy implicit user-item interactions within a diffusion process. In particular, DiffGT applies an SSL loss to maximise the agreement between the original user/item embeddings and the denoised user/item embeddings during model training, thereby improving the model expressiveness and yielding an improved top-K recommendation performance.

To address the insufficient modality encoding issue, we first focus on enhancing the modality fusion within multi-modal graph-based recommender systems. In particular, we propose the Multi-modal Graph Contrastive Learning (MMGCL) model, which introduces modality-specific graph augmentations as positive samples and a modality-aware negative sampling strategy in an SSL loss. This enhances the modality fusion process, unlike the existing approaches that often treat each modality with equal importance. In addition, we further enhance the modality fusion by using Large Multi-modal (LMM) encoders. We show that by using SSL to enable deep modality alignment across modalities, these LMM encoders significantly outperform the shallow alignment methods common in existing graph-based recommender systems. On the other hand, in addition to addressing modality fusion, there are still longstanding isolation problems – isolated feature extraction process and isolated modality encoding process – that impede the effective mining of self-supervised signals across multiple modalities and remain unresolved in existing multi-modal graph-based recommender systems. To address these isolation problems, we introduce the Unified multi-modal Graph Transformer (UGT), a novel end-to-end architecture that uses SSL to unify the multi-modal representations into the same semantic space, thereby enhancing top-K multi-modal recommendation within a unified graph transformer architecture.

Next, to address the insufficient domain transfer capabilities in existing cross-domain models, we introduce two novel graph-based approaches. The first, Personalised Graph Prompt-based Recommendation (PGPRec), is an ID-based approach that enables effective and parameter-efficient cross-domain knowledge transfer. Specifically, PGPRec first uses SSL to pre-train a graph encoder, ensuring that it learns high-quality and generalisable knowledge across domains. This knowledge is then effectively transferred from a single source domain to a target domain via personalised and item-wise graph prompts. To further enhance generalisation and reduce reliance on ID-based features, inspired by the model soup paradigm, we propose AdapterSoupRec, which leverages multi-modal large language models (MLLMs) to generate universal item representations. In particular, we use SSL and cross-entropy losses to enable MLLMs to generate highly generalisable representations and effective model configurations. These configurations are then combined via a weighted average (i.e., the 'model soup' technique) to create a more effective set of model parameters, thereby achieving improved top-K recommendation in a multi-domain setting.

Overall, this thesis contributes novel and effective SSL-enhanced graph-based recommender models that systematically address the challenges of limited architectural expressiveness, insufficient modality encoding, and insufficient domain transfer capabilities. Our extensive experimental evaluations on numerous real-world datasets validate the thesis statement, demonstrating that recommendation effectiveness is significantly enhanced by explicitly mining more supervision signals from diverse modalities and domains. These contributions make progress towards the development of effective graph-based recommender systems and pave the way for further future directions of research in top-K recommender systems.

Acknowledgements

This PhD has been a truly remarkable journey. I still remember the moment of the first fist-bump with my first supervisor, Iadh Ounis, for my first paper acceptance. During these four years, I have been fortunate and received immense support, making this thesis possible.

I would like to express my sincere gratitude to my supervisors, Iadh Ounis and Craig Macdonald. They guided me on the path to becoming an independent researcher and offered me invaluable opportunities to engage with the research community. I truly appreciate their dedication to building an environment of high scientific standards, where I could conduct this research.

I must thank my wife, Yanlin Xu. Her love, care, and endless support have been my foundation throughout these years. I am so grateful for the joy she has brought to our lives with our wonderful daughter, Sihan (Missy) Yi. To my daughter, Missy: your arrival has been the greatest gift of my life. I am also deeply grateful to my parents (Liwei Ji, Ping Yi) and my parents-in-law (Lei Wang, Huandong Xu). They encouraged me to take this journey and to face any challenge with bravery.

I would also like to thank my friends and colleagues at the Terrier team for their great company and support: Zaiqiao Meng, Xi Wang, Anjie Fang, Siwei Liu, Sarawoot Kongyoung, Hitarth Narvala, Xiao Wang, Yaxiong Wu, Javier Sanz-Cruzado Puig, Thomas Janich, Aleksandr Petrov, Giuseppe Spillo, Jack McKechnie, Andreas Chari, Andrew Parry, Lubingzhi Guo, Xinhao Yi, Jinyuan Fang, Shen Dong, Zeyuan Meng, and many others. It has been an honour and a pleasure to work with them.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivations

With the rapid expansion of digital applications and services, such as e-commerce platforms (e.g., Amazon, Alibaba), social media networks (e.g., Facebook, X), and multimedia content platforms (e.g., TikTok, Instagram), users increasingly depend on personalised top-K recommender systems to access relevant digital content that matches their interests (Yuan, 2024). Platforms like Amazon and Alibaba offer an extensive range of items, from consumer goods to multimedia content, to cater to diverse consumer needs. Meanwhile, TikTok and Instagram manage large volumes of multimedia data streams to facilitate user engagement and interaction. These platforms aim to recommend content that is not only relevant but also engages users by accurately predicting their preferences (Deldjoo et al., 2020). However, the complexity of recommendation tasks increases as modern digital platforms deal with heterogeneous and multi-modal data, that encompasses not just textual descriptions, visual images, and acoustic signals, but also user interactions from diverse domains (Kaminskas and Ricci, 2012). Specifically, using multi-modal and cross-domain data in recommender systems refers to the integration and usage of description information from multiple data sources or types of user and item sides. For example, items might include a textual title, categories and a textual description, while the user profiles might include ages and occupations. In addition, users often engage with multiple services across various domains, such as watching movies on streaming platforms, reviewing products on e-commerce sites, and discussing interests on social networks. Cross-domain data involves information collected from the aforementioned varied services, thus enabling top-K recommender systems to estimate user behaviours and preferences more effectively by leveraging from related but distinct contexts (Zhang et al., 2023b). Indeed, existing works have demonstrated that both multi-modal and multi-domain data can enable recommender systems to generate more effective recommendations (Liu et al., 2024b). Therefore, there is a pressing need for recommender systems to demonstrate strong generalisation capabilities across multiple modalities and domains while maintaining personalised accuracy (Zang et al., 2022).

Traditional top-K recommender systems primarily rely on collaborative filtering (CF) techniques, which estimate user preferences by modelling similar patterns of historical interactions between users and items (Su and Khoshgoftaar, 2009). Conventional CF methods either use user-item matrix factorisation or neighbourhood-based similarity calculations to generate recommendations for the target users. These CF methods, while foundational, often yield shallow embeddings that may not capture complex or fine-grained user preferences as effectively as more sophisticated models, such as graph neural network-based recommendation models (Marcuzzo et al., 2022). Moreover, exiting CF methods often fail to incorporate external signals beyond user-item interactions, leading to sparse, non-informative embeddings (Han et al., 2022). This is because conventional CF methods rely heavily on ID-based representations where each user and item is indexed arbitrarily, thus lacking the capability to incorporate meaningful contextual information, such as user demographics, item features, or user interactions across different domains (e.g., using data from a video platform to enhance recommendations on a music platform) (Lin et al., 2024). As a result, CF methods struggle in scenarios with sparse interaction data, limiting their effectiveness in capturing more fine-grained user preferences. These issues highlight the necessity for richer and more contextualised representations, to enhance the performance of top-K recommender systems.

Graph representation learning has recently gained significant attention across various research fields, demonstrating its capacity to process and interpret graph-structured data effectively within specific recommendation contexts (Ju et al., 2024). This representation learning method focuses on extracting latent representations from nodes, edges, and entire graphs or subgraphs, thus encapsulating the relationships and features within the graph structure (Chikwendu et al., 2023). Notably, Graph Neural Networks (GNNs) have emerged as a promising approach for modelling the dependencies across nodes and specific types of graphs, such as user-item interaction graphs and knowledge graphs (Welling and Kipf, 2016). This effectiveness of graph-based models stems from their ability to propagate and aggregate features from neighbouring nodes to enhance the representation of each target node through multiple graph convolutional layers, resulting in refined node representations. Consequently, graph-based models are highly effective in tasks such as node classification and link prediction, where understanding and leveraging the relationships within the graphs are crucial (Wu et al., 2020b). In the context of recommendation, graph-based models typically model user-item interactions within a bipartite graph that represents users and items as nodes, with edges reflecting their historical interactions. Unlike traditional CF methods, which depend on pre-defined similarity metrics, graph-based models propagate and aggregate collaborative signals across multiple interaction layers, effectively capturing high-order connectivity patterns between users and items on the user-item interaction graph (Wu et al., 2022). In addition, graph-based recommender systems can further enhance the estimation of user preferences by incorporating a variety of knowledge sources, such as different modalities and domains (Gao et al., 2023). These graph-based recommender systems extend their capabilities beyond using a single interaction graph by leveraging multiple streams of interaction graphs, with each graph representing various types of interactions or dimensions specific to different modalities or domains. Therefore, the use of graph-based recommender systems is a promising research direction for improving top-K recommendation accuracy by enabling better representation learning of user-item interactions and exploring more complex relationships across various modalities and domains.

Although graph-based approaches have shown significant effectiveness, graph-based recommender systems still present several challenges (Anand and Maurya, 2025). One of the most critical challenges in graph-based recommender systems is the over-smoothing problem, where repeated graph aggregation operations between connected nodes result in indistinguishable user and item embeddings (Li et al., 2024). As the number of GNN layers increases, user/item node embeddings become increasingly similar, thus reducing the model's ability to capture fine-grained user preferences. This over-smoothing problem also leads to the loss of diversity in recommendations, as different users receive increasingly similar and homogeneous item recommendations, thus impeding effective personalisation for the target users (Liu et al., 2024c). However, effective recommender systems should suggest items that are relevant yet distinctly personalised to ensure meaningful recommendations for the target users. Several approaches have been proposed to mitigate over-smoothing, including residual connections and skip connections techniques that maintain representation diversity (Wang et al., 2023b; Zhang et al., 2024c). On the other hand, implicitly collected user interactions, while abundant, often include noisy false-positive and false-negative interactions (Wang et al., 2021c). These noisy interactions pose a significant challenge for graph-based recommender systems, which aggregate the users' or items' neighbours based on these implicit user-item interactions to capture the users' profiles. The prevalent noise disrupts the accurate modelling of the users' profiles, leading to a gap between the observed interactions and actual user preferences, which in turn harms the recommendation accuracy. As such, there remains a need for more expressive graph architectures that can both effectively denoise the noisy implicit interactions for improved effectiveness while maintaining representation distinctiveness. Consequently, it is still unclear how to enhance existing GNN architectures to effectively denoise noisy interactions and distinguish the users/items with the same neighbours after the graph convolution operations within the graph-based recommender systems. To this end, Self-supervised learning (SSL) techniques offer a promising approach to develop such advanced graph architectures. By providing additional contrastive signals, SSL have been shown to preserve user-item uniqueness while still benefiting from graph-based models' relational modelling capabilities (Xie et al., 2022b).

Another challenge of graph-based recommender systems is the insufficient modality encoding issue. Modern recommender systems typically integrate various data modalities – such as text, images, and audio – to develop richer item and user representations. Existing widely used graph-based recommender systems, such as LightGCN (He et al., 2020), rely solely on learnable user and item embeddings without using weight matrices or deeper attention layers. This simple architecture, by design, lacks the capacity to process and fuse the diverse features from different modalities, such as visual or textual data. As a result, they struggle to effectively integrate additional modalities, limiting their ability to capture richer contextual information for improved recommendations. Unlike deep learning models in natural language processing (NLP) and computer vision (CV), which explicitly model relationships between textual and visual information, graph-based recommenders lack dedicated components for multi-modal feature fusion for effective user-item representation learning. For instance, e-commerce platforms frequently include item descriptions, product images, and user-generated reviews. A user's preference might be influenced by both textual content and visual aesthetics. However, traditional graph-based recommenders often treat these modalities separately rather than learning a unified multi-modal representation. Furthermore, existing methods usually concatenate or average multi-modal embeddings rather than modelling their interactions explicitly. Therefore, this problem of insufficient modality encoding in graph-based recommender systems remains unresolved. Advanced multi-modal representation learning techniques (Radford et al., 2021; Zong et al., 2024), such as SSL, can assist recommender systems in capturing cross-modal dependencies, allowing recommender systems to learn more holistic and coherent user preferences. For example, SSL can explicitly align visual and textual embeddings by maximising the agreement between an item image and its corresponding description in the latent space, thereby addressing the gap of how to leverage these rich multi-modal representations to improve recommendation accuracy.

Furthermore, recommender systems deployed in real-world applications require generalisation across multiple domains to capture diverse user interests. For example, a user who frequently purchases books on an e-commerce site may also show interest in related products such as literary-themed posters or speciality stationery. However, traditional graph-based recommender systems do not typically transfer knowledge from one domain to another due to the commonly observed data sparsity issue, which refers to an insufficient number of user-item interactions, thus hindering accurate recommendation generation (Chen et al., 2024). To address this, cross-domain recommendation techniques have been developed to leverage user-item interactions from related source domains to enhance recommendations in a target domain (Zhao et al., 2024). However, many conventional graph-based approaches primarily rely on simple transfer learning techniques that do not fully exploit the graph structure of user-item interactions. This challenge refers to the insufficient domain transfer capability of graph-based recommender systems (Ayemowa et al., 2024). While SSL techniques (such as SSL-based pre-training) have shown great promise in facilitating cross-domain adaptation in NLP field, it remains unclear how to establish an effective transfer learning paradigm for graph-based recommender systems. Therefore, it is worth addressing this gap by investigating how to use SSL-enhanced transfer learning paradigms to achieve an effective cross-domain and multi-domain knowledge transfer, thereby improving the effectiveness and generalisability of graph-based recommender systems.

1.2 Thesis Statement

The statement of this thesis is that the graph-based recommender systems can be effectively enhanced by obtaining self-supervised supervision signals from diverse domains and modalities using a more advanced graph neural architecture. Specifically, this thesis argues that by effectively mining self-supervised signals from multiple item modalities, graph-based recommender systems can gain richer user and item representations, thereby improving the quality of recommendations. Additionally, leveraging multi-modal semantics across abundant source domains through self-supervised learning approaches can further enhance the effectiveness and efficiency of graph-based recommender systems, enabling an improved performance across various target domains. Finally, this thesis hypothesises that by developing more expressive graph neural architectures, graph-based recommender systems can more effectively incorporate complex and structured information, captured through self-supervised learning, into comprehensive user and item embeddings, thus addressing limited expressiveness inherent to existing graph-based models.

1.3 Contributions

The summary of our contributions is described below:

- **Self-supervised Graph Architecture (Chapter 4)**: To address the limitations of graph-based recommender systems mentioned in Section 1.1, particularly the over-smoothing problem and the noisy implicit interaction issue, we introduce two novel recommendation models. To alleviate the over-smoothing problem, we first propose a novel model named Positional Graph Contrastive Learning (PGCL) for top-K recommendation, which aims to use graph positional encoding techniques at the feature propagation level to improve the expressive power of graph-based recommender systems and ensure the effectiveness of position-enhanced user/item representations using an SSL-based contrastive learning manner. Specifically, PGCL provides additional positional information (i.e., Laplacian eigenvectors and the random walk operators) to existing graph recommenders by injecting the learned graph positional encoding into each feature propagation layer. By integrating learned graph positional encodings, our proposed PGCL model not only successfully alleviates the over-smoothing problem but also significantly enhances the performance of existing graph-based recommender systems, leading to a more expressive and effective graph architecture.

  In addition, to denoise implicit interactions in recommender systems, we propose a novel graph transformer model, namely Diffusion Graph Transformer (DiffGT), at the overall architectural level. DiffGT leverages a diffusion process, which includes a forward phase for gradually introducing noise to implicit interactions, followed by a reverse process to iteratively refine the representations of the users' hidden preferences (i.e., a denoising process). In contrast to PGCL's positional encoding techniques, the DiffGT model proposes a graph transformer at the overall architectural level instead of integrating positional encodings at the feature propagation level, leveraging an attention mechanism to learn global-level dependencies between users and items (Vaswani et al., 2017). Moreover, we introduce the anisotropic and directional noise in the forward process of diffusion to better model the uncertainty inherent in implicit feedback, and leverage a linear transformer to denoise the obtained noisy embeddings for efficient purposes. We also contrast the original and denoised embeddings in a self-supervised loss to generate additional supervision signals for a more effective diffusion process during training.

By comparing our PGCL and DiffGT model with existing graph-based recommender systems, we aim to answer the following research question: Can we leverage more effective graph architectures to improve the effectiveness of graph-based recommender systems, specifically by addressing two key aspects: (1) mitigating the over-smoothing problem using positional encodings within an SSL framework (PGCL), and (2) denoising implicit user interactions via a diffusion-based graph transformer (DiffGT)?

- **Self-supervised Modality Fusion using Modal-specific Graph Augmentations and Modality Alignment (Chapter 5)**: To address the insufficient modality encoding issue mentioned in Section 1.1, we aim to enhance modality fusion by explicitly modelling modality-specific contributions in the top-K multi-modal recommendation task. As such, we first propose a Multi-modal Graph Contrastive Learning (MMGCL) model which explicitly manipulates diverse item modality features (visual, acoustic, textual) in an SSL manner. Specifically, we propose two graph augmentations as positive samples, namely modality edge dropout and modality masking, to preserve the main intentions of the target users within the user-item interaction graph. Moreover, we devise a modality-aware negative sampling method, denoted challenging negative, to generate hard negative samples that differ from positive samples in just one modality. We then perform SSL to maximise the agreement between the positive samples and minimise the agreement between the positive and negative samples, thereby facilitating more effective user/item representation learning in the top-K multi-modal recommendation scenario.

  Despite that MMGCL has shown its effectiveness in the multi-modal recommendation task, the existing multi-modal graph-based models primarily depend on the features extracted individually from different media through pre-trained modality-specific encoders, and exhibit only shallow alignments between different modalities, thereby limiting these systems' ability to capture the underlying relationships between the modalities. Therefore, we explore more effective multi-modal encoders and the benefits of deep alignment between modalities to enhance the top-K recommendation task in a multi-modal setting. We first investigate the use of large multi-modal (LMM) encoders such as CLIP, VLMo, and BEiT-3 for deep modality alignment by pre-training the item image and description pair using SSL losses. Our findings demonstrate that SSL-trained deep alignment significantly outperforms traditional modality-specific encoders that exhibit shallow alignment. Compared with MMGCL that leverages modality-specific encoders, the use of LMM encoders not only improves the effectiveness of MMGCL but also enhances all recent multi-modal graph-based models, such as MMGCN, SLMRec, LATTICE, in an end-to-end training strategy. Besides, the use of SSL-enhanced LMM encoders does enhance the contribution of each modality by achieving a deeper alignment across diverse modalities more effectively than all existing recommendation models that use modal-specific encoders.

By comparing our MMGCL model and the LMM-enhanced scheme with existing multi-modal graph-based recommender systems, we aim to answer the following research question: Can we facilitate more effective multi-modal representation learning to improve the effectiveness of multi-modal graph-based recommender systems, specifically by investigating (1) dedicated graph augmentations and modality-aware negative sampling for multi-modal contrastive learning (MMGCL), and (2) the performance benefits of self-supervised deep modality alignment using large multi-modal encoders?

- **Self-supervised Unified Architecture for Multi-modal Representation Learning (Chapter 6):** Despite that the MMGCL model and LMM-enhanced graph-based models enable more effective user/item representation learning, there are still longstanding isolation problems – isolated feature extraction process and isolated modality encoding process – that remain unsolved in the multi-modal recommendation task. Firstly, an isolated extraction process underestimates the importance of effective feature extraction in multi-modal recommendations, potentially incorporating non-relevant information, which is harmful to item representations. Second, an isolated modality encoding process produces disjoint embeddings for item modalities due to the individual processing of each modality, which leads to a suboptimal fusion of user/item representations for an effective user preferences prediction. To address both aforementioned isolated processes to enable the consistent extraction and cohesive fusion of joint multi-modal features, we propose a novel model, namely Unified multi-modal Graph Transformer (UGT), which leverages a graph transformer architecture to directly extract aligned multi-modal features from raw data. Specifically, UGT uses a multi-way transformer for the unified extraction of aligned multi-modal features and a unified GNN with an attentive fusion mechanism for improved modality integration. Compared to the MMGCL model and LMM-enhanced approach, UGT is a unified architecture to resolve the isolated feature extraction problem in multi-modal recommendations and the newly designed unified GNN acts as the fusion component to uniformly fuse multi-modal features in a unified manner to address the problem of isolated modality encoding in the existing multi-modal recommender systems. By applying SSL in UGT's multi-modal item features, UGT successfully aligns these embeddings to a unified and cohesive semantic space, thereby improving the recommendation accuracy in the top-K recommendation task. Besides, our UGT model enhances the contribution of each modality in multi-modal recommendation, preventing the dominance of a single modality as observed in in the existing multi-modal recommender systems. Moreover, our UGT model also exhibits a strong out-of-domain performance from an e-commerce scenario to a micro-video recommendation scenario. We also note that UGT can generalise to the multi-domain recommendation scenario using encoded multi-modal features to bridge user/item representations across domains.

  By comparing our UGT model with seven existing multi-modal graph-based recommender systems, we examine the performance of our unified graph transformer model on three real-world datasets to answer the following research question: Does our UGT model enable more effective multi-modal representation learning to improve the effectiveness of multi-modal graph-based recommender systems?

- **Self-supervised Graph Prompting for Cross-domain Knowledge Transfer (Chapter 7):** To address the insufficient domain transfer capability of graph-based recommender systems mentioned in Section 1.1, we propose a Personalised Graph Prompt-based Recommendation (PGPRec) framework to effectively transfer structural graph knowledge from source domains to target domains. Specifically, we propose the first personalised and item-wise graph prompts based on relevant items to perform parameter-efficient prompt-tuning in the top-K cross-domain recommendation task. In particular, we apply SSL-based contrastive learning to generate the pre-trained embeddings for overlapping users, to allow an increased generalisability in the pre-training stage and to ensure an effective prompt-tuning stage.

  By comparing our PGPRec approach with existing cross-domain and graph-based recommender systems, we aim to answer the following research question: Can we leverage more effective cross-domain representation learning to improve the effectiveness of graph-based recommender systems, specifically by investigating personalised graph prompts for effective cross-domain knowledge transfer?

- **Self-supervised Universal Representation Learning for Multi-domain Knowledge Transfer (Chapter 8):** To further enhance the domain transfer capability of graph-based recommender systems, unlike PGPRec that focuses on ID-based features, we propose the AdapterSoupRec model, which leverages multi-modal large language models (MLLMs) to generate universal item representations across multiple domains. In particular, we use SSL and cross-entropy losses to train MLLMs to generate effective model configurations, so as to obtain a final set of MLLM parameters via weighted average (i.e., the 'model soup' technique) for an improved generalisation. This well-trained MLLM is then coupled with a graph adapter for domain-specific adaptation in the top-K multi-domain recommendation task.

By comparing our AdapterSoupRec model with existing cross-domain, multi-domain and multi-modal baselines, we aim to answer the following research question: Can we leverage more effective multi-domain representation learning to improve the effectiveness of graph-based recommender systems, specifically by investigating SSL-enhanced MLLMs combined with graph adapters for improved multi-domain generalisation?

- **A Unified Recommender System with Multiple Modalities and Domains (Chapter 9):** With the development of the proposed graph architectures and methods from Chapter 4 to 8, we present a composite step forward, demonstrating how the core principles of this thesis can be synthesised and advanced within a new, more powerful Large Language Model (LLM) paradigm while retaining the advantages of each method. We introduce GollaRec, a recommendation model whose novelty lies not in integrating previous models, but in enhancing their foundational concepts into more sophisticated forms. Specifically, GollaRec represents a significant extension of the graph prompting strategy from Chapter 7 with a more sophisticated Graph-of-Thought (GoT) prompting technique that demonstrates a personalised, multi-step reasoning chain for the LLM in top-K recommendation tasks. The multi-modal representations from Chapters 5 and 6 are conceptually extended from being input features to becoming integral components of these reasoning "thoughts," allowing the LLM to reason with both visual and textual data. GollaRec also presents an architectural improvement from specialised graph transformers (c.f. Chapter 4) to a graph-LLM architecture that leverages enhanced principles of prompting, multi-modal fusion and multi-domain adaptation for a new state-of-the-art in recommendation.

  By comparing our GollaRec model with existing general, cross-domain, multi-domain and multi-modal baselines, we aim to answer the research question: Can our unified graph-based recommender system achieve enhanced top-K recommendation performance with multiple modalities and domains?

In summary, this thesis advances the field of graph-based recommendation by presenting a suite of novel SSL-based recommendation models that address the insufficient modality and domain mining problems. We introduce expressive architectures to address the problems of over-smoothing and noisy implicit interactions, establish novel techniques for integrating multi-modal signals, and propose effective methods for multi-domain knowledge transfer. The scope of this work focuses on improving top-K recommendation by developing these more expressive and generalisable graph-based models, achieved by pioneering novel SSL techniques tailored to each specific recommendation scenario. These contributions collectively validate the thesis statement (Section 1.2), demonstrating that recommendation effectiveness is significantly enhanced by mining and integrating supervision signals from diverse modalities and domains within more expressive architectures.

1.4 Origins of Material

Most of the material presented in this thesis is derived from peer-reviewed works published or in review at international conferences and journals, detailed as follows:
Chapter 4: This chapter presents novel graph neural architectures focused on expressive graph representation learning:

- The Positional Graph Contrastive Learning (PGCL) model was published in the Proceedings of the 45th European Conference on Information Retrieval (ECIR), 2023;
- The Diffusion Graph Transformer (DiffGT) model is under review at the ACM Transactions on Recommender Systems (TORS).

Chapter 5: We introduce our proposed techniques that enable effective modality fusion in the graph-based recommender systems through the use of self-supervised modality-specific graph augmentations and self-supervised modality alignment:

- The Multi-modal Graph Contrastive Learning (MMGCL) model appeared in the Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2022;
- The work on Large Multi-modal Encoders for Recommendation (LMM4Rec) was accepted by the ACM Transactions on Recommender Systems (TORS).

Chapter 6: We propose an effective unified architecture to overcome the isolation problems in the traditional multi-modal recommendation pipeline by leveraging self-supervised learning to align visual and textual embeddings within a common semantic space:

- The Unified Graph Transformer (UGT) model was published in the Proceedings of the 18th ACM Conference on Recommender Systems (RecSys), 2024;
- An extended version of UGT was accepted by the ACM Transactions on Recommender Systems (TORS) special issue "Highlights of RecSys 2024".

Chapter 7: We propose a graph prompt-tuning technique to enhance the cross-domain transfer capability of graph-based recommender systems:

- The Personalised Graph Prompt-tuning Recommendation (PGPRec) model was accepted in the ACM Transactions on Information Systems (TOIS);

Chapter 8: We propose an SSL-enhanced MLLM to model generalisable universal representations across multiple domains, so as to further enhance the domain transfer capability of graph-based recommender systems:

- The AdapterSoupRec model was published in the Proceedings of the 47th European Conference on Information Retrieval (ECIR), 2025.

Chapter 9: We further propose an advanced and unified graph-based recommender system that retains the advantages of all previous proposed models:

- The GollaRec model was published in the Proceedings of the 2025 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

1.5 Thesis Outline

This thesis focuses on two main research directions, namely graph-based recommender systems and self-supervised learning (SSL). In particular, we aim to enhance the top-K recommendation performance of graph-based recommender systems through effectively mining more supervision signals from multiple modalities and domains within an advanced graph architecture. The remainder of this thesis is organised as follows:

- Chapter 2 describes the background for this thesis. We first define the top-K recommendation task and introduce a taxonomy of existing approaches – classifying them into collaborative filtering, content-based, and graph-based paradigms – to explicitly define the scope of this thesis. We then detail the fundamental architectures of recommender systems, with a specific emphasis on graph-based models. Furthermore, we introduce Graph Neural Networks (GNNs) and Self-Supervised Learning (SSL), discussing how these techniques are integrated into the top-K recommendation scenario. The chapter concludes by outlining the evaluation metrics used throughout the thesis.

- Chapter 3 discusses the related work in general, multi-modal, and multi-domain recommendation, with a specific focus on graph-based approaches. First, we describe conventional graph-based models and SSL-based techniques for top-K recommendation, highlighting critical unresolved challenges such as the over-smoothing problem and the inability to capture noisy implicit interactions. Next, we discuss multi-modal recommendation, identifying key limitations in existing graph-based models, particularly insufficient modality fusion and the longstanding isolation problems in feature extraction and encoding. Finally, we discuss the state-of-the-art recommendation models in cross-domain and multi-domain recommendation, identifying the insufficient domain transfer capabilities that hinder the generalisability of current graph-based recommender systems.

- Chapter 4 introduces expressive graph neural architectures, focusing on techniques designed to capture complex and structured information through SSL. In particular, two key proposed models are presented: Positional Graph Contrastive Learning (PGCL) and Diffusion Graph Transformer (DiffGT). PGCL uses SSL to enrich user/item embeddings with graph positional encodings to alleviate the over-smoothing problem in graph-based recommender systems with improved effectiveness. We evaluate PGCL on real-world datasets (i.e., publicly available Yelp, Gowalla and Amazon-Kindle datasets) and discuss its performance in comparison to state-of-the-art recommendation approaches (NGCF, LightGCN, SGL, SimGCL). DiffGT leverages SSL to ensure an effective diffusion process with a graph transformer architecture to handle noise inherent in user-item interactions, demonstrating significant improvements in recommendation effectiveness. We evaluate the performance of DiffGT in comparison to ten established and state-of-the-art recommendation approaches (MF, CDAE, MultiVAE, DiffRec, CVAE, HIRE, LightGCN, SGL, GiffCF, SimGCL, GFormer) on the Yelp, MovieLens-1M and FourSquare datasets.

- Chapter 5 presents the investigation on enhancing modality fusion for the top-K multi-domain recommendation task. It introduces modality-specific graph augmentations (i.e., modality masking and modality edge dropout) in the multi-modal graph contrastive learning (MMGCL) model and investigates the use of deep modality alignment methods by using large multi-modal (LMM) encoders as feature extractors. MMGCL explicitly leverages SSL to contrast modality-specific augmentations to obtain richer user/item embeddings on the Tiktok and MovieLens-1M datasets, whereas the use of LMM encoder explores SSL-trained deep modality alignment techniques using state-of-the-art multi-modal encoders (CLIP, VLMo, BEiT-3) to significantly improve the alignment of visual and textual item features, resulting in enhanced recommendation performance on three Amazon Review datasets (Sports, Clothing, Electronic).

- Chapter 6 continues the self-supervised mining of rich signals from multiple modalities, deepening the investigations of Chapter 6. It specifically addresses the limitations of isolated feature extraction and modality encoding within the recommendation pipeline by introducing the Unified Multi-modal Graph Transformer (UGT). The UGT model overcomes the isolated processes of feature extraction and modality encoding into a unified end-to-end architecture. UGT consistently outperforms nine state-of-the-art baselines (VBPR, MMGCN, MMGCL, SLMRec, LATTICE, BM3, FREEDOM, PGMT, LightGT) by effectively leveraging a multi-way transformer and a unified GNNs to produce more coherent and aligned user/item representations, which significantly improve recommendations in both standard and cold-start scenarios on three Amazon Review datasets (Sports, Clothing, Electronics).

- Chapter 7 introduces the Personalised Graph Prompt-based Recommendation (PGPRec) framework for top-K cross-domain recommendation. Specifically, PGPRec first uses SSL to pre-train a graph encoder to learn generalisable user representations in the source domain, and then combines these representations with item-wise prompts to facilitate effective domain adaptation while achieving state-of-the-art recommendation accuracy on four Amazon Review datasets (Sports, Clothing, Electronics and Phone).

- Chapter 8 presents a further step by leveraging multi-modal semantic features to model universal representations across multiple domains instead of modelling ID-based features used in PGPRec. This chapter introduces AdapterSoupRec, which combines an SSL-trained MLLM with a graph adapter to balance general multi-domain knowledge with domain-specific adaptation, producing improved performance gains across diverse e-commerce scenarios (Pantry, Sports, Electronics).

- Chapter 9 presents GollaRec, an advanced and unified recommender system that extends the core principles established throughout this thesis. By leveraging Graph-of-Thought (GoT) prompting within Multi-modal Large Language Models (MLLMs), integrates visual, textual, and graph-structured data within a unified recommendation architecture. This approach significantly enhances MLLM to capture and leverage complex relational structures and multi-modal contexts in top-K general and multi-domain recommendation in the multi-modal setting. The empirical evaluations highlight the effectiveness of GollaRec, demonstrating superior performance against multiple state-of-the-art baselines (VBPR, MMGCL, BM3, CLIP, BEiT-3, LLaVA, P5, LMRecSys and TALLRec).

- Chapter 10 concludes the thesis by highlighting the key contributions of each chapter and reflecting on the overall achievements of the research. We also propose potential future directions for our research.

Chapter 2

Background

In Chapter 1, we presented an overview of graph-based recommender systems, highlighting their functionalities and inherent limitations. To alleviate those limitations – the issues of the less expressive architecture with over-smoothing and noisy implicit interactions, the insufficient modality encoding issue and the insufficient transfer capability issue – we proposed to use self-supervised graph architectures, self-supervised mining techniques of multiple modalities, as well as self-supervised mining techniques of multiple domains in Section 1.3. In this chapter, we first provide an overview of the recommendation task in general, followed by the fundamental notations and conceptualisations of graph-based recommender systems on which the thesis is built. In general, we describe the existing techniques, especially those recommender systems that our proposed methods rely on for modelling the complex dependencies of user-item interactions in a collaborative filtering manner (Anand and Maurya, 2025). In particular, we first describe some classic recommender approaches, including the Collaborative Filtering (CF)-based and content-based approaches in Section 2.1.2. Next, we focus on graph-based recommender systems by first introducing some background information on graph representation learning , such as graph data structure and graph-based transformations, followed by a discussion of their relationships between recommender systems in Section 2.2. Next, we introduce self-supervised learning along with its application to graph-based recommender systems in Section 2.3. Finally, we describe commonly-used evaluation methods and metrics in Section 2.4.

2.1 Overview of Recommender Systems

Recommender systems are essential tools designed to assist users in navigating the overwhelming volume of choices available in modern digital environments (Garapati and Chakraborty, 2025). With the exponential growth of online content – ranging from products and movies to venues and services – users often struggle to identify items that align with their preferences. To address this, recommender systems aim to personalise the user experience by filtering out non-relevant options and highlighting those that are likely to be of interest. This personalisation not only enhances user satisfaction but also drives business value by increasing engagement and conversions (Venice et al., 2025).

In general, recommender systems generate top-K personalised recommendations by leveraging the users' historical interactions (e.g., clicks, views, likes, ratings), item attributes (e.g., genre), and additional contextual signals, such as temporal patterns (e.g., in session-based recommendation) or spatial information (e.g., in point-of-interest recommendation). User feedback falls into two main categories: explicit (e.g., ratings) and implicit (e.g., clicks, views, purchases). While explicit feedback provides clearer supervision signals for learning user preferences, it is increasingly sparse due to privacy concerns and the user's reluctance to provide such input (Chen et al., 2024). As a result, most modern recommender systems primarily rely on implicit feedback, which only indicates whether a user interacted with an item, without clarifying whether the interaction reflects a positive or negative preference.

2.1.1 Explicit vs. Implicit Feedback

User feedback plays a pivotal role in shaping the effectiveness of recommender systems. We generally classify this feedback into two primary types: explicit or implicit.

2.1.1.1 Explicit feedback

Explicit feedback represents direct input from users, such as numerical ratings (e.g., a 1-5 star rating for a movie) or written reviews. Despite explicit feedback offering clear, quantifiable reflections of user preferences, explicit feedback is typically sparse due to the user's typical reluctance in providing such input to the corresponding systems (Zhou et al., 2025). To be more specific, explicit feedback-based recommender systems function by collecting explicit feedback from users, such as ratings for movies or products, as illustrated in Figure 2.1. The main objective involves predicting ratings for items a user has not yet interacted with. This process is akin to filling missing values in a sparse user-item matrix $R \in \mathbb{R}^{N_u \times N_i}$, where N_u denotes the number of users and N_i denotes the number of items. In this matrix, R_{ui} represents the explicit rating user u has given to item i. Most entries in R are unknown (or 0) because a user typically interacts with only a small fraction of all available items. Techniques like matrix factorisation and deep learning are commonly employed to predict these unknown ratings based on known values (Ahmadian et al., 2025).

2.1.1.2 Implicit feedback

Implicit feedback, in contrast, is abundant and indistinct, encompassing user actions such as clicks, views, purchases, and browsing behaviours. Due to its ease of collection and rich availability, implicit feedback forms the basis of most modern recommender systems. Unlike explicit feedback systems, implicit feedback-based recommender systems infer user preferences

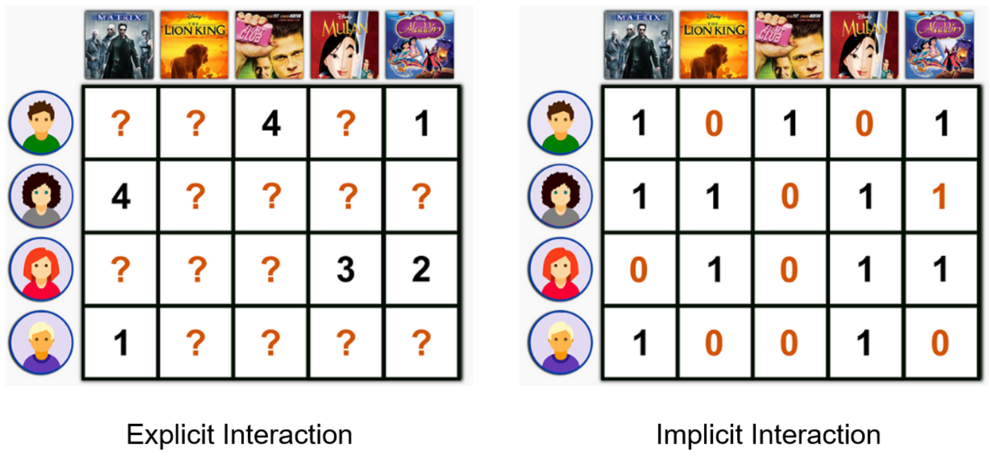


Figure 2.1: Explicit and implicit feedback of users.

from any user behaviours such as viewing history, click patterns, and purchase records. Indeed, these systems interpret all interactions as a positive indication of user preferences, although the absence of interaction might not necessarily imply disinterest. As discussed in Chapter 1, a key challenge with implicit feedback-based recommender systems is the noisy implicit interactions. Since these systems need to determine the degree of interest from inherently ambiguous interactions – such as a brief view or a mistaken click – the data often contains false positives (interactions that do not reflect true preference) and false negatives (items a user likes but has not interacted with). To mitigate this problem, common methods for implicit feedback often employ complex models that can capture patterns related to interaction frequency and context, thereby aiming to estimate user preferences more accurately despite the noise (Sguerra et al., 2025). Formally, as shown in Figure 2.1, we represent user-item interactions with implicit feedback using a binary interaction matrix $R \in \mathbb{R}^{N_u \times N_i}$:

$$R_{ui} = \begin{cases} 1, & \text{if user } u \text{ interacted with item } i \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

2.1.2 Collaborative Filtering vs. Content-Based Recommender Systems

The basic models for recommender systems generally leverage two types of data to generate effective recommendations to users, namely (1) the user-item interactions and (2) the content information about users and/or items (e.g., item images or descriptions).

2.1.2.1 Collaborative Filtering-based (CF) recommender systems

CF models recommend items to users by exploiting similarities in historical user-item interaction patterns without relying on item descriptions or user profiles. A foundational method within CF is Matrix Factorisation (MF) (Koren et al., 2009), which decomposes the user-item interaction matrix R into low-dimensional user and item matrices. The predicted interaction $\hat{r}_{ui}$ between user u and item i is computed as follows:

$$\hat{r}_{ui} = e_u \cdot e_i, \tag{2.2}$$

where e_u and e_i represent latent embeddings for users and items, respectively To optimise embeddings, MF typically minimises a regularised squared error loss:

$$\mathcal{L}(\Theta) = \frac{1}{2} \sum_{(u,i) \in D} (r_{ui} - \hat{r}_{ui})^2 + \frac{\lambda}{2} \|\Theta\|^2, \tag{2.3}$$

where λ regulates the complexity, and $\|\Theta\|^2$ denotes the norm of all model parameters.

However, when dealing with implicit feedback, where interactions are binary (e.g., a click indicates interest, but no click does not necessarily mean disinterest), the squared error loss is less appropriate. This is because implicit feedback typically only provides positive signals. To address this, Bayesian Personalised Ranking (BPR) (Rendle et al., 2009) introduces a pairwise ranking approach. BPR optimises for the correct relative ordering of items for a given user rather than predicting explicit ratings. Indeed, BPR assumes that a user prefers an interacted item over a non-interacted one. Specifically, for each user u, BPR samples a triplet (u, i, j), where i is an item user u has interacted with (a positive sample), and j is an item user u has not interacted with (a negative sample). BPR's objective is to maximise the probability that the positive item i is ranked higher than the negative item j for user u:

$$\mathcal{L}_{BPR}(\Theta) = - \sum_{(u,i,j) \in D_S} \ln \sigma(\hat{x}_{uij}) + \lambda \|\Theta\|^2, \tag{2.4}$$

where $D_S = \{(u, i, j) | i \in I_u^+, j \in I_u^-\}$ represents the set of all triplets, with I_u^+ being items user u interacted with and I_u^- being items user u did not interact with. Here, $\hat{x}_{uij} = \hat{r}_{ui} - \hat{r}_{uj}$ represents the predicted difference in preference between item i and item j for user u, and $\sigma(\cdot)$ is the sigmoid function. By optimising this ranking-based loss, BPR effectively learns latent factors that preserve the relative order of items, making it highly effective for implicit feedback scenarios (Wu, 2002). The ability of collaborative filtering methods, like BPR, to capture user interests and item characteristics by learning from interactions is fundamental to designing effective recommender systems. Indeed, many advanced deep learning-based recommendation approaches (He et al., 2020; Milogradskii et al., 2024; Zhang et al., 2019) build upon or incorporate principles from BPR.

2.1.2.2 Content-based recommender systems

Content-based recommender systems recommend items similar to those a user liked in the past based on visual or textual features associated with the compared items. For instance, if a user likes a movie with certain actors or genres, the system suggests movies with similar attributes. A key advantage of content-based recommender systems is their ability to recommend new items to users, even if those items have no interaction history (the so-called "cold-start" item problem), as long as their content features are well-defined (Panda and Ray, 2022). These content-based recommender systems can also provide transparent recommendations by explaining why an item was recommended (e.g., "Because you liked movies starring Actor X") (Javed et al., 2021). However, these systems rely heavily on the availability and quality of item metadata and can struggle with over-specialisation, recommending only items very similar to past preferences, thereby limiting user exposure to diverse content (Yago et al., 2018). These systems also face a cold-start problem for new users who lack sufficient interaction history to build a reliable content-based profile (Volkovs et al., 2017).

Typically, content-based recommender systems operate by representing both users and items in a common feature space. For textual content such as item descriptions, items are often represented as vectors where each dimension corresponds to a term (word). A common method for weighting these terms is Term Frequency-Inverse Document Frequency (TF-IDF). User profiles are then constructed by aggregating the TF-IDF vectors of items they liked. Recommendations are made by calculating the *cosine similarity* between the user vector and the item vectors. Formally, a user profile P_u can be represented as a weighted sum of the feature vectors F_i of items $i \in I_u^+$ that user u has positively interacted with:

$$P_u = \sum_{i \in I_u^+} w_i F_i \tag{2.5}$$

where w_i can be a weight based on the user's rating or an implicit interaction strength. The similarity between a user profile P_u and a candidate item j (with feature vector F_j) can then be computed using a similarity metric, often cosine similarity:

$$\text{Similarity}(P_u, F_j) = \frac{P_u \cdot F_j}{||P_u|| ||F_j||} \tag{2.6}$$

The content-based recommender system then recommends items with the highest similarity scores. Despite their simplicity, these methods exhibit limitations in their ability to semantically interpret complex features and adapt to diverse modalities (De Gemmis et al., 2015).

In modern content-based recommendation, particularly for handling rich, unstructured content like images, audio, or complex text, *deep learning models* (Wang et al., 2020b) have become paramount. These models act as powerful *feature extractors*. For instance:

- Convolutional Neural Networks (CNNs) (Wang et al., 2020b; Wu et al., 2019b) are widely used to extract hierarchical visual features from item images.

- Recurrent Neural Networks (RNNs) (Kanwal et al., 2021; Wang et al., 2020b) or Transformer models (Roy et al., 2024; Zivic et al., 2024) are used to capture sequential and contextual information from textual descriptions or user reviews.

These deep learning models are widely used feature extractors to transform raw content into dense, low-dimensional vector representations (embeddings). The objective is to embed items in a latent space where semantically or visually similar items are close to each other. It is worth noting that all content-based (multi-modal) recommender systems leverage these feature extractors as an isolated feature extraction process within the recommendation workflow (Agarwal, 2023). In this thesis, we aim to unify the content-based (multi-modal) recommendation workflow to improve the modality encoding capability of top-K recommender systems.

A widely used loss function for learning these discriminative embeddings is the *Triplet Loss* (Yuan et al., 2020). This loss function aims to ensure that an anchor example's embedding is closer to a positive example's embedding than it is to a negative example's embedding by at least a certain margin. Specifically, for an anchor $\mathcal{A}$, a positive sample $\mathcal{P}$ (similar to $\mathcal{A}$), and a negative sample $\mathcal{N}$ (dissimilar to A), the Triplet Loss is defined as follows:

$$L_{\text{triplet}}(\mathcal{A}, \mathcal{P}, \mathcal{N}) = \max(0, \|f(\mathcal{A}) - f(\mathcal{P})\|_2^2 - \|f(\mathcal{A}) - f(\mathcal{N})\|_2^2 + \alpha) \tag{2.7}$$

where $f(\cdot)$ denotes the deep neural network (e.g., CNN, Transformer) acting as the embedding function, $\|\cdot\|_2^2$ is the squared Euclidean distance, and α is a margin parameter. This margin ensures that the distance between the anchor and the negative example is sufficiently greater than the distance between the anchor and the positive examples. After obtaining these learned embeddings, the recommendation process typically involves calculating the similarity (e.g., cosine similarity) between a user's learned profile embedding (often aggregated from the liked items' embeddings) and candidate item embeddings. This approach enables robust item retrieval and similarity-based recommendations across various modalities, facilitating both *uni-modal* scenarios (e.g., image-to-image matching) and *cross-modal* scenarios (e.g., text-to-image matching).

On the other hand, in the following sections, we introduce advanced graph representation learning and self-supervised learning, both of which play a complementary and increasingly significant role alongside collaborative filtering and content-based approaches in enhancing top-K recommender systems. We further describe their architectures and the sophisticated losses. These concepts form the critical building blocks for the methods proposed in this thesis.

2.2 Graph Representation Learning

As introduced in Section 2.1.2, Matrix Factorisation and BPR primarily learn latent embeddings from direct user-item interactions as established baselines for collaborative filtering. However, real-world recommendation scenarios often involve complex, high-order relationships that simple matrix decomposition might miss (Hasan et al., 2024). To capture these complex dependencies and leverage the structural information inherent in user-item interaction data, Graph

Neural Networks (GNNs) have emerged as a highly effective approach in modern collaborative filtering (Wu et al., 2022). Specifically, GNNs model user-item interactions as a bipartite graph, where users and items are represented as nodes, and historical interactions form the edges between these nodes. Unlike traditional MF, which operates on a flattened matrix, GNNs explicitly propagate and aggregate information across the graph structure. This allows them to effectively capture high-order collaborative signals by iteratively refining user and item embeddings based on their neighbours' representations across multiple layers. For instance, a GNN can learn that two users are similar not just because they interacted with the same item, but because they interacted with items that share similar users. In the following sections, we introduce the preliminaries of GNN, its message passing mechanism, and its typical graph neural architecture.

2.2.1 Preliminaries

To formally describe GNNs, we first introduce essential graph notations. A graph is typically defined as $G = (V, E)$, where V is the set of nodes (or vertices) and E is the set of edges (or links) connecting pairs of nodes. For recommender systems, nodes typically represent users and items, while edges denote interactions (e.g., clicks, purchases).

- **Adjacency Matrix (A)**: For a graph with N nodes, its structure is often represented by an $N \times N$ **adjacency matrix** A. If an edge exists between node i and node j, then $A_{ij} = 1$; otherwise, $A_{ij} = 0$. In undirected graphs, $A_{ij} = A_{ji}$. For bipartite user-item graphs, we can construct a full adjacency matrix including user-user, item-item, and user-item connections, or focus on the user-item interaction sub-matrix.

- **Degree Matrix (D)**: The **degree matrix** D is an $N \times N$ diagonal matrix where D_{ii} represents the degree of node i, which is the number of edges connected to node i. Formally, $D_{ii} = \sum_j A_{ij}$.

- **Laplacian Matrix (L)**: The **Laplacian matrix** is a fundamental matrix in graph theory, defined as $L = D - A$. It plays a crucial role in spectral graph theory and serves as the basis for many graph convolutional operations.

- **Normalised Laplacian Matrix ($\mathcal{L}$)**: To prevent features from having vastly different scales when aggregated from neighbours with varying degrees, a **normalised Laplacian matrix** is often used. The symmetrically normalised Laplacian is defined as follows:

$$\mathcal{L} = I - D^{-1/2} A D^{-1/2}$$

  where I is the identity matrix. This normalisation allows to stabilise the learning process and improve performance.

2.2.2 Message Passing

The core idea behind GNNs in recommendation is the *message passing paradigm*. This iterative process allows information to flow across the graph, enabling each node to gather and integrate features from its local neighbourhood (He et al., 2020). Indeed, each node collects "messages" from its direct neighbours, aggregates these messages, and then updates its own representation. This process is repeated for multiple layers, allowing information to propagate to further reaches of the graph, thereby capturing higher-order connectivity patterns. A general form of a GNN layer for node v can be expressed as follows:

$$\mathbf{m}_v^{(l)} = \text{AGGREGATE}^{(l)} \left(\left\{ \mathbf{h}_u^{(l-1)} \mid u \in \mathcal{N}(v) \right\} \right) \tag{2.8}$$

$$\mathbf{h}_v^{(l)} = \text{COMBINE}^{(l)} \left(\mathbf{h}_v^{(l-1)}, \mathbf{m}_v^{(l)} \right) \tag{2.9}$$

where $\mathbf{h}_v^{(l)}$ is the representation of node v at layer l, $\mathcal{N}(v)$ denotes the set of neighbours of node v, $\mathbf{m}_v^{(l)}$ is the aggregated message for node v at layer l. AGGREGATE$^{(l)}(\cdot)$ is an aggregation function (e.g., sum, mean, max-pooling) that combines neighbour embeddings, and COMBINE$^{(l)}(\cdot)$ is a combination function (e.g., concatenation, addition, linear transformation) that merges the aggregated neighbourhood information with the node's representation from the previous layer.

2.2.3 Graph Convolutional Networks (GCNs)

GCNs (Welling and Kipf, 2016) are a specific and widely adopted type of GNNs that adapt convolutional operations from grid-like data (e.g., images) to graph-structured data. The core idea is to learn node representations by convolving features of a node with features of its neighbours. The layer-wise propagation rule for a GCN can be expressed as follows:

$$\mathbf{H}^{(l+1)} = \sigma \left(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} \mathbf{H}^{(l)} \mathbf{W}^{(l)} \right) \tag{2.10}$$

where $\mathbf{H}^{(l)}$ is the matrix of node embeddings at layer l, with $\mathbf{H}^{(0)}$ typically being the initial feature matrix (e.g., one-hot IDs or content features). $\mathbf{W}^{(l)}$ is a learnable weight matrix for layer l. $\sigma(\cdot)$ is an activation function (e.g., ReLU). $\tilde{A} = A + I$ is the adjacency matrix with added self-loops (to include the node's own features in its aggregation) and $\tilde{D}$ is the diagonal degree matrix of $\tilde{A}$, where $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. The term $\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$ effectively normalises the aggregated features, preventing exploding or vanishing gradients and helping to preserve the scale of features during propagation. This operation allows each node to aggregate information from its neighbourhood, weighted by its connectivity.

In the context of recommendation, the graph typically begins with initial user and item embeddings (Deng, 2022). Through multiple GCN layers, these embeddings are iteratively refined by aggregating information from connected users and items. The final user embeddings $\mathbf{e}_u$ and item embeddings $\mathbf{e}_i$ obtained after L layers are then used to predict interaction scores, often via a simple inner product, similar to MF: $\hat{r}_{ui} = \mathbf{e}_u \cdot \mathbf{e}_i$. The entire GNN model is optimised end-to-end using ranking losses such as BPR (c.f. Equation 2.4) or other suitable objectives. GNNs also offer the flexibility to incorporate additional side information (e.g., item attributes, user demographics) directly into the initial node features or the message passing mechanism, further enriching user and item representations. Consequently, GNNs have demonstrated a superior performance in capturing fine-grained user preferences and complex relationships in a collaborative filtering manner (Wu et al., 2022). These inherent capabilities of GNNs in modelling complex relational data lay the groundwork for our thesis, where we specifically address their limitations in expressiveness, capability for multi-modal and multi-domain learning through novel self-supervised graph neural architectures.

Building on the promising capabilities of GNNs in recommender systems, we explore the Self-Supervised Learning (SSL) paradigm in Section 2.3. In the next section, we introduce its foundational concepts and primary methods, demonstrating how SSL can further enhance graph-based recommender systems.

2.3 Self-Supervised Learning

While supervised deep learning has achieved tremendous success, its significant reliance on extensive, manually labelled data presents a notable limitation for many real-world applications (Ahmed et al., 2023; Sarker, 2021). As an alternative paradigm, Self-Supervised Learning (SSL) has garnered increasing attention due to its remarkable efficacy in learning powerful data representations from unlabelled data (Jing et al., 2023). The fundamental principle of SSL is to generate supervision signals directly from the input data itself, enabling models to learn meaningful representations without explicit human annotations. This learning process typically involves defining and solving a "pretext task" designed to capture inherent data properties. Given its ability to mitigate the common challenge of data scarcity, SSL's applications are rapidly expanding across various domains, including computer vision and graph learning, which has profoundly inspired its adoption in recommender systems (Ren et al., 2024).

2.3.1 Preliminaries

SSL's core idea is to learn general-purpose representations by performing auxiliary tasks (pretext tasks) where labels are automatically derived from the data. These learned representations can then be transferred to downstream tasks (e.g., classification, recommendation). Common pretext task designs often involve:

- Generative Tasks (Hou et al., 2022; Liu et al., 2021b): Reconstructing missing or corrupted parts of the input (e.g., masked auto-encoders).

- Contrastive Tasks (Liu et al., 2021b; Ren et al., 2025): Maximising agreement between different views (augmentations) of the same data instance while pushing away views of different instances.
- Predictive Tasks (Lee et al., 2021b; Wu et al., 2021a): Predicting specific attributes or properties derived from the input (e.g., relative position).

The effectiveness of SSL heavily relies on data augmentation, which generates different "views" or corrupted versions of the original data. These augmented views then serve as the positive and negative samples within various pretext tasks, allowing the model to learn invariant and effective features during model training.

2.3.2 Self-Supervised Learning in Graph Learning

Graph-structured data, characterised by its non-Euclidean nature, presents unique challenges for augmentation compared to Euclidean data like images (Asif et al., 2021). Unlike geometric transformations applicable to images, graph data augmentation typically involves *structure-level perturbations* or *feature-level augmentations*. This close conceptual connection between graph learning and recommender systems, where recommendation tasks often map to link prediction or node classification on graphs, makes SSL in graph domains particularly relevant to this thesis.

2.3.2.1 Graph-Specific Data Augmentations

Graph data augmentations aim to create diverse views of a graph while preserving its fundamental semantic meaning, thereby serving as crucial inputs for self-supervised pretext tasks. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with an adjacency matrix $\mathbf{A}$ and a node feature matrix $\mathbf{X}$, common augmentations include:

- **Structure-level Augmentations**: These modify the graph's topology and can be categorised into:
  - *Node/Edge Dropout*: Randomly removing a proportion ρ of nodes from $\mathcal{V}$ (along with their incident edges) or edges from $\mathcal{E}$ (Zhou et al., 2020a). This can be formally expressed as generating $\tilde{\mathbf{A}}$ from $\mathbf{A}$ by setting $A_{ij} = 0$ with probability ρ.
  - *Subgraph Sampling*: Extracting a subgraph $\tilde{\mathcal{G}} \subset \mathcal{G}$ by selecting a subset of nodes and their induced edges based on various rules (e.g., random walks, ego-networks) (Veličković et al., 2018).
  - *Graph Diffusion/Rewiring*: Modifying existing edges or adding new ones based on certain heuristics or learnable mechanisms (You et al., 2020; Zhu et al., 2020). This can involve computing a diffused adjacency matrix $\tilde{\mathbf{A}} = (1 - \alpha)\mathbf{A} + \alpha\mathbf{S}$, where $\mathbf{S}$ is a similarity matrix.

- **Feature-level Augmentations**: These operate on the node feature matrix $\mathbf{X}$, and can be categorised into:
  - *Feature Masking/Dropout*: Randomly masking or setting to zero a portion of features in $\mathbf{X}$ (Thakoor et al., 2021).
  - *Feature Shuffling/Corruption*: Permuting or corrupting feature values within $\mathbf{X}$ to create varied input representations (Hassani and Khasahmadi, 2020).
  - *Feature Mixing*: Interpolating features from different nodes or existing feature distributions to synthesize new feature vectors for nodes (Wu et al., 2021b). Formally, $\tilde{\mathbf{x}}_v = \alpha \mathbf{x}_v + (1-\alpha)\mathbf{x}_u$ for some $\alpha \in [0,1]$.

It is important to note that indiscriminate application of these augmentations can inadvertently change essential semantic patterns of the original graph, especially when dropping important nodes or edges (Ding et al., 2022). This highlights the importance of adaptive or learnable augmentation strategies that consider both topological and semantic priors (Suresh et al., 2021; Zhu et al., 2020). Consequently, this highlights the need for task-aware augmentation methods that align with the objectives of specific learning tasks.

2.3.2.2 Typical Pretext Tasks in Graph Learning

The design of effective pretext tasks is central to self-supervised graph learning, particularly given the dominance of GNNs in this domain. Different paradigms exist:

- **Contrastive Methods (Ma and Liu, 2023; Wu et al., 2021a)**: These are highly prevalent, aiming to learn discriminative node embeddings by contrasting different views of the same graph instance. The objective is to maximize the similarity between embeddings of augmented views of the same node/graph while minimizing similarity to embeddings of other nodes/graphs. For an encoder $f_{\boldsymbol{\Theta}}$ and two augmented views $\tilde{\mathcal{D}}_1, \tilde{\mathcal{D}}_2$ of an original graph $\mathcal{D}$, the contrastive loss $\mathcal{L}_{ssl}$ typically involves maximizing a similarity metric (e.g., cosine similarity) between positive pairs and minimizing it for negative pairs. A common form is shown as follows:

$$\mathcal{L}_{ssl}^{\text{contrastive}} = -\frac{1}{|\mathcal{B}|} \sum_{k \in \mathcal{B}} \log \frac{\exp(\text{sim}(\mathbf{h}_{k,1}, \mathbf{h}_{k,2})/\tau)}{\sum_{j \in \mathcal{B}, j \neq k} \exp(\text{sim}(\mathbf{h}_{k,1}, \mathbf{h}_{j,2})/\tau)} \tag{2.11}$$

  where $\mathbf{h}_{k,1} = f_{\boldsymbol{\Theta}}(\tilde{\mathcal{D}}_{k,1})$ and $\mathbf{h}_{k,2} = f_{\boldsymbol{\Theta}}(\tilde{\mathcal{D}}_{k,2})$ are embeddings of two augmented views of instance k, $\mathcal{B}$ is a mini-batch, sim is a similarity function, and τ is a temperature parameter (Hassani and Khasahmadi, 2020; You et al., 2020).

- **Generative Methods (Wu et al., 2021a; Yu et al., 2024)**: These tasks focus on reconstructing missing or corrupted information.

  - *Masked Attribute Prediction*: Predicting masked node features based on the unmasked features and graph structure. The loss typically measures the difference between predicted and original features:

$$\mathcal{L}_{ssl}^{\text{generative}} = \|\mathbf{M} \odot (f_{\mathbf{\Theta}}(\tilde{\mathbf{X}}, \mathbf{A}) - \mathbf{X})\|_F^2 \tag{2.12}$$

    where $\mathbf{M}$ is a mask matrix and $\odot$ is the Hadamard product (Hu et al., 2020).

  - *Adjacency Matrix Reconstruction*: Reconstructing the original graph's adjacency matrix from a corrupted version or learned node embeddings. The loss aims to match the reconstructed adjacency matrix $\hat{\mathbf{A}}$ with the true one $\mathbf{A}$ (Wang et al., 2020a). For example, a Bernoulli negative log-likelihood loss for reconstructed adjacency $\hat{\mathbf{A}}$ can be used as follows:

$$\mathcal{L}_{ssl}^{\text{reconstruct}} = \mathbb{E}_{\mathbf{A}\sim p(\mathbf{A})}[-\sum_{i,j}(\mathbf{A}_{ij}\log\hat{\mathbf{A}}_{ij} + (1-\mathbf{A}_{ij})\log(1-\hat{\mathbf{A}}_{ij}))] \tag{2.13}$$

- **Predictive Methods (Lee et al., 2021b; Liu et al., 2022b)**: These methods involve predicting properties or context derived from the graph. For instance, predicting node properties (e.g., node degree) or contextual relationships (e.g., pair distance) from node embeddings (Veličković et al., 2019). Unlike contrastive or generative approaches, predictive SSL tasks directly formulate supervision signals in a classification or regression manner. For instance, given a node embedding $h_v = f_\Theta(v)$ generated by the encoder, the model predicts a label distribution $\hat{y}_v = \text{softmax}(Wh_v)$, which is compared with the true pseudo-label y_v. The corresponding *cross-entropy loss*, commonly used for classification-style SSL pretext tasks, is defined as:

$$\mathcal{L}_{\text{CE}}^{\text{predictive}} = -\frac{1}{|V|}\sum_{v\in V}\sum_{c=1}^{C} y_{v,c}\log(\hat{y}_{v,c}) \tag{2.14}$$

  where C is the number of classes, $y_{v,c}$ denotes the one-hot encoded true label for class c, and $\hat{y}_{v,c}$ is the predicted probability for node v belonging to class c.

These advances in self-supervised graph learning provide a foundation and direct inspiration for designing effective self-supervised recommender systems by explicitly leveraging graph structures. In this thesis, we primarily focus on applying self-supervised graph learning methods to the top-K recommendation tasks. We select this top-K recommendation setting because it most directly reflects the core objective of real-world systems: identifying and ranking the most relevant items for each target user.

2.3.3 Self-supervised Learning in Recommender Systems

The recommendation community has increasingly adopted SSL, leveraging its demonstrated success in other fields to address unique challenges in personalised recommendation. A growing number of parallel studies (Ren et al., 2025; Yao et al., 2021) focus either on developing tailored data augmentation techniques for recommendation or on devising novel self-supervised tasks directly relevant to user preference learning. The recommendation community has been increasingly turning to SSL in light of its demonstrated success in other fields (Gui et al., 2024; Jing and Tian, 2020). We have observed a growing number of parallel studies being conducted over the same period, with a focus on data augmentation techniques for recommendations Indeed, effective data augmentation is crucial for learning high-quality and generalisable representations in self-supervised recommendation (Wu et al., 2021a). We categorise commonly used data augmentation approaches in this field into three main types: sequence-based, graph-based, and feature-based.

2.3.3.1 Sequence-Based Augmentation

Given a sequence of items $\mathcal{S} = [i_1, i_2, \ldots, i_k]$ representing a user's historical behaviours, common sequence-based augmentations $\tilde{\mathcal{S}} = \mathfrak{T}(\mathcal{S})$ include:

- **Item Masking**: Randomly masking a proportion γ of items, replacing them with special tokens (e.g., '[mask]') (Sun et al., 2019). This can be formalized as $\tilde{\mathcal{S}}_t = \mathcal{S}_t$ if $r_t > \gamma$, else $\tilde{\mathcal{S}}_t =$ mask token, where $r_t \sim U(0, 1)$. The underlying idea is that a user's intention is relatively stable during a period of time. Thus, even with partial masking, the primary intent information is largely retained.

- **Item Cropping**: Randomly selecting a continuous sub-sequence of length $L_c = \lfloor \eta \cdot |\mathcal{S}| \rfloor$ (where $\eta \in (0, 1)$ adjusts the length) (Sun et al., 2019). This method provides a local view of the user's historical sequence. Through the self-supervised task, the selected sub-sequence enables the model to learn generalised representations without relying on a comprehensive user profile.

- **Item Reordering**: Shuffling a continuous sub-sequence to create augmentations, fostering robustness to order variations (Xie et al., 2022b).

- **Item Substitution**: Substituting items in short sequences with highly correlated items to inject less corruption (Liu et al., 2021a). A correlated item is obtained by calculating the correlation score which is based on the item co-occurrence or the similarity of the corresponding representations.

- **Item Insertion**: Inserting correlated items into short sequences to provide richer sequence representations capturing user dynamics (Liu et al., 2021a). This operation results in an augmented sequence of length $|\mathcal{S}| + k_{ins}$, where k_{ins} is the number of inserted items.

2.3.3.2 Graph-Based Augmentation

Given the user-item graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with adjacency matrix $\mathbf{A}$ (or other relevant graphs), the graph augmentation approaches from Section 2.2 can be directly applied to user-item interaction graphs, focusing on their relevance to user-item interaction graphs.

2.3.3.3 Feature-Based Augmentation

Feature-based augmentations operate on user or item attributes, or their learned embeddings, which we collectively denote as the feature matrix $\mathbf{X}$. These methods generate diverse views of user and item features by applying various transformations in the attribute or embedding space. Common feature-based augmentation techniques include:

- **Feature Dropout/Masking**: This method randomly masks or sets to zero a portion of the features within $\mathbf{X}$. It is analogous to item masking in sequences or edge dropout in graphs, fostering robustness by forcing the model to learn from incomplete information (Wu et al., 2021a; Xie et al., 2022b; Zhou et al., 2020a). Formally, for a feature matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$ (where N is the number of entities and D is the feature dimension), the augmented feature matrix $\tilde{\mathbf{X}}$ can be obtained as follows:

$$\tilde{\mathbf{X}}_{ij} = \mathbf{X}_{ij} \cdot m_{ij}, \quad \text{where } m_{ij} \sim \text{Bernoulli}(1 - \rho) \tag{2.15}$$

  Here, m_{ij} is a binary mask element, and ρ is the masking probability.

- **Feature Shuffling**: This technique involves permuting rows or columns within the feature matrix $\mathbf{X}$, effectively randomising the contextual information associated with features to create augmented views (Hassani and Khasahmadi, 2020; Li et al., 2020a). For example, shuffling features across a set of items can be conceptualised through multiplication with a permutation matrix $\mathbf{P}$:

$$\tilde{\mathbf{X}} = \mathbf{X}\mathbf{P} \tag{2.16}$$

  where $\mathbf{P}$ is a randomly generated permutation matrix.

- **Feature Clustering**: This augmentation method combines clustering with contrastive learning, ensuring that user and item representations should be close to their respective cluster prototypes in the feature space. (Fan et al., 2022; Luo and Luo, 2020; Xie et al., 2020a). The augmented prototype representations are typically learned via clustering within an expectation-maximisation (EM) framework, effectively creating refined "views" of feature groups.

- **Feature Mixing**: This approach generates new feature vectors by interpolating between original user/item features and features from other samples or previous versions of the same features (Wu et al., 2021a; Xie et al., 2022b). It is often used to synthesise informative negative or positive examples. For two feature vectors $\mathbf{x}_a$ and $\mathbf{x}_b$, the mixed feature vector $\tilde{\mathrm{x}}$ can be generated as follows:

$$\tilde{\mathbf{x}} = \alpha \mathbf{x}_a + (1 - \alpha)\mathbf{x}_b \tag{2.17}$$

where $\alpha \in [0, 1]$ is a mixing coefficient.

While many existing self-supervised recommendation methods adapt augmentation techniques from computer vision, natural language processing, and general graph learning fields, these approaches are not always seamlessly transferable. User behaviour data is often scenario-specific, noisy, and random. Moreover, most current augmentation methods rely on heuristics, necessitating trial-and-error for an optimal configuration. This highlights that effective data augmentation for recommendation is not only about generating views, but about how these views contribute to the broader self-supervised learning process. Therefore, we next delve into the typical learning paradigms that leverage these augmented data, followed by a discussion of the various training modes that integrate these components into a cohesive recommender system.

2.3.3.4 Training Modes for Self-Supervised Recommender Systems

The general training pipeline for self-supervised recommender systems involves three phases: (1) data augmentation, (2) encoding (using $f_{\mathbf{\Theta}}$), and (3) optimisation. The coordination between the primary recommendation task and the self-supervised pretext task can vary, leading to different training modes. We summarise three typical modes: Joint Learning (JL), Pre-training and Fine-tuning (PF), and Integrated Learning (IL).

- **Joint Learning (Huang et al., 2022; Yu et al., 2023b):** Joint learning optimises the pretext task and the primary recommendation task simultaneously through a shared encoder $f_{\mathbf{\Theta}}$. A trade-off between the two objectives, $\mathcal{L}_{ssl}$ and $\mathcal{L}_{rec}$, is achieved by a hyper-parameter α controlling the self-supervision magnitude. Here, the pretext task serves as an auxiliary regularisation mechanism, formulated as follows:

$$\mathbf{\Theta}^*, \phi_r^* = \underset{\mathbf{\Theta}, \phi_r}{\arg\min} \left[\mathcal{L}_{rec}\left(g_{\phi_r}\left(f_{\mathbf{\Theta}}(\mathcal{D})\right)\right) + \alpha \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_{\mathbf{\Theta}}(\tilde{\mathcal{D}})\right)\right)\right] \tag{2.18}$$

  where g_{ϕ_s} is the projection head for the SSL task and g_{ϕ_r} for the recommendation task.

- **Pre-training and Fine-tuning (Xu et al., 2023; Yu et al., 2023b):** Pre-training and fine-tuning is a standard two-stage training technique. In the first stage (pre-training), the encoder $f_{\mathbf{\Theta}}$ is trained with pretext tasks on augmented data $\tilde{\mathcal{D}}$ to obtain robust initial parameters $\boldsymbol{\Theta}_{\text{init}}$. In the second stage (fine-tuning), the pre-trained encoder $f_{\boldsymbol{\Theta}_{\text{init}}}$ is then fine-tuned on the original data $\mathcal{D}$ for the recommendation task:

$$\boldsymbol{\Theta}_{\text{init}}, \phi_s^* = \underset{\boldsymbol{\Theta}, \phi_s}{\arg\min} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_{\boldsymbol{\Theta}}(\tilde{\mathcal{D}})\right), \mathcal{D}_{\text{target}}\right), \tag{2.19}$$

$$\boldsymbol{\Theta}^*, \phi_r^* = \underset{\boldsymbol{\Theta}_{\text{init}}, \phi_r}{\arg\min} \mathcal{L}_{rec}\left(g_{\phi_r}\left(f_{\boldsymbol{\Theta}_{\text{init}}}(\mathcal{D})\right)\right). \tag{2.20}$$

- **Integrated Learning (Jing et al., 2023; Ren et al., 2025; Yu et al., 2023b):** Integrated learning unifies the pretext task and the recommendation task into a single, integrated objective. The overall loss $\mathcal{L}$ directly measures the difference or mutual information between the two outputs of the recommendation prediction and the self-supervised task:

$$\boldsymbol{\Theta}^*, \phi_r^*, \phi_s^* = \underset{\boldsymbol{\Theta}, \phi_r, \phi_s}{\arg\min} \mathcal{L}\left(g_{\phi_r}\left(f_{\boldsymbol{\Theta}}(\mathcal{D})\right), g_{\phi_s}\left(f_{\boldsymbol{\Theta}}(\tilde{\mathcal{D}})\right)\right) \tag{2.21}$$

Building upon the foundational concepts of graph learning and self-supervised learning discussed in this section, the next section introduces the evaluation metrics. These metrics are essential for precisely evaluating the performance of general, graph-based, and SSL-based recommender systems examined in this thesis.

2.4 Evaluation Methodology and Metrics

Recommender systems operate on various types of inputs, yet the outputs are usually of one form: a ranked list of recommended items, with the most recommended items at the top. Thus, the recommender systems are commonly evaluated with a set of ranking-based metrics. These metrics are usually computed on the top-K items retrieved in the ranking, with K varying from 1 to 100, depending on the need (e.g., number of recommended items that can be displayed on a website). Thus, recommender systems are evaluated using metrics suited to their tasks:

RMSE (Hodson, 2022): Root Mean Square Error (RMSE) is a simple and widely used metric to evaluate the accuracy of a model that predicts the value of a continuous variable, such as explicit feedback (e.g. a rating a user would give to a movie), for known user-item pairs:

$$\text{RMSE}(r, \hat{r}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i - \hat{r}_i)^2}$$

Precision@K and Recall@K (Torgo and Ribeiro, 2009): We can evaluate a model that outputs continuous values using Root Mean Squared Error. However, in cases where we wish to evaluate a model that produces a binary output (e.g. relevant/non-relevant or clicked/not clicked), we need a change of metric to be more effective. Moreover, since we are dealing with a recommendation task, we are mostly interested in recommending top-N items to a user, instead of the entire item list in the system. Thus, the metric employ should be parameterised by an integer K, which tells us how many items to take into consideration when computing the metric. Given a set of top-K recommended items, Precision @ K is the proportion of the recommended items that are relevant:

$$\mathrm{P}_u @ \mathrm{~K} = \frac{\mathrm{TP}_u}{\mathrm{TP}_u + \mathrm{FP}_u} = \frac{\sum_{i=1}^{K} \mathrm{rel}_{ui}}{K}$$

$$\text{Precision@K} = \frac{\sum_{u=1}^{U} \mathrm{P}_u @ K}{K}$$

We consider an item to be relevant if the item appears in the ground-truth interactions for the corresponding user. On the other hand, Recall @ K is the proportion of relevant items found in the top-K recommendations:

$$\mathrm{R}_u @ \mathrm{~K} = \frac{\mathrm{TP}_u}{\mathrm{TP}_u + \mathrm{TN}_u} = \frac{\sum_{i=1}^{K} \mathrm{rel}_{ui}}{|\mathrm{rel}_u|}$$

$$\text{Recall} @ \mathrm{~K} = \frac{\sum_{u=1}^{U} \mathrm{R}_u @ K}{K}$$

where $|\mathrm{rel}_u|$ is the total number of relevant items for user u.

MAP@K (Buttcher et al., 2016): Mean Average Precision (MAP) is another metric that is used to evaluate recommender systems that work on implicit feedback/binary output that takes the order of the results list into consideration. In order to compute it, we first define the concept of average precision. Given a value K, the average precision @ K (AP@K) metric represents the average of precision values for relevant items from 1 to K :

$$\mathrm{AP}_u @ \mathrm{~K} = \frac{\sum_{i=1}^{K} \mathrm{rel}_{ui} \, \mathrm{P}_u @ i}{\sum_{i=1}^{K} \mathrm{rel}_{ui}}$$

The value of the metric is higher if the recommended items are both relevant and presented in higher positions of the results list. The mean average precision at $\mathrm{K}(\mathrm{MAP} @ \mathrm{~K})$ is given as the mean of $\mathrm{AP}_u @ \mathrm{~K}$ across all the users in the dataset:

$$\mathrm{MAP} @ \mathrm{~K} = \frac{\sum_{u=1}^{U} \mathrm{AP}_u @ \mathrm{~K}}{U}$$

NDCG@K (Borko, 1962): Normalised Discounted Cumulative Gain (NDCG) is a metric used to evaluate recommender systems with explicit feedback that allows to quantify the quality of the current ordering of top-K relevant items in relation to a perfect ordering of the top-K most relevant items. In order to compute NDCG, we first need to define the concept of cumulative gain, which corresponds to the sum of all the relevance scores (i.e. ratings given to items) up to

K, of a top-K recommendations list:

$$\mathrm{CG@\,K} = \sum_{i=1}^{K} rel_i$$

Cumulative gain on its own does not take into consideration the ordering of the resulting items. An ordering discount term can be introduced, as items at distant positions in a ranking are generally less influential than those at earlier positions. This results in the Discounted Cumulative Gain (DCG) metric as follows:

$$\mathrm{DCG@\,K} = \sum_{i=1}^{K} \frac{rel_i}{\log_2(i+1)}$$

Finally, in order to compare recommenders that potentially return a different number of results, we introduce a normalisation termm which is set to the discounted cumulative gain of a perfect order of top-K items, by relevance:

$$\mathrm{IDCG@\,K} = \sum_{i=1}^{K} \frac{rel_i^{\text{ideal}}}{\log_2(i+1)}$$

Taking into consideration all of the previously defined terms, we can compute the normalised discounted cumulative gain as follows:

$$\mathrm{NDCG@\,K} = \frac{\mathrm{DCG@\,K}}{\mathrm{IDCG@\,K}}$$

MRR@K (Gupta et al., 2019): Mean Reciprocal Rank (MRR) is a measure used to evaluate the quality of sequential recommenders, or generally the quality of a ranking system, based on the position of the correctly recommended items in a ranked list. A large MRR value indicates that correct recommendations are at the top of the ranking list. The reciprocal rank (RR) of a list of recommendations provided to a user u, is defined as the inverse position (called rank) of the first relevant item among the first K items in the list:

$$\mathrm{RR}_u\mathrm{@\,K} = \frac{1}{\mathrm{rank}_{u,1}}$$

For example, if we are considering a list of $\mathrm{K} = 5$ recommendations, assuming that there is relevant item within the K recommendations, then depending on its position, the reciprocal rank can take the following values:

$$1/1, 1/2, 1/3, 1/4, 1/5.$$

MRR@K is the average of reciprocal ranks of the correctly-recommended items across all the users in the dataset:

$$\mathrm{MRR@\ K} = \frac{\sum_{u=1}^{U} \mathrm{RR}_u\mathrm{@\ K}}{U}$$

However, MRR has known limitations as a single evaluation metric since it emphasises only the first relevant item and ignores the rest of the ranked list, which can misrepresent overall ranking quality. This issue has been discussed in the literature (Fuhr, 2018). Therefore, we report MRR primarily for comparison with certain baselines and prior work that use the same metric, and we rely on additional metrics, such as Recall@K and NDCG@K, for a more complete assessment.

2.5 Conclusions

In this chapter, we have introduced the important concepts and preliminaries of the top-k recommender systems. We began by categorising different recommender system taxonomies, focusing specifically on collaborative filtering-based and content-based approaches. Next, we explored graph representation learning, highlighting its advantages and elucidating its critical role in modelling user-item interactions for the top-k recommendation task. This included presenting the necessary preliminaries, such as graph definitions, Laplacian matrices, and the fundamental architecture of Graph Neural Networks (GNNs). Furthermore, we introduced the self-supervised learning (SSL) paradigm, discussing its benefits and its intrinsic connections to both graph-based methods and the recommendation tasks. We also described various augmentation techniques and training modes for applying SSL within recommender systems. Finally, we outlined the evaluation methods and metrics commonly employed for recommender systems, which will be consistently used throughout this thesis.

The comprehensive background provided in this chapter lays the groundwork for understanding the inherent limitations of current graph-based recommender systems. As first introduced in Chapter 1, these limitations primarily pertain to less expressive architectures susceptible to over-smoothing and noisy implicit interactions, as well as issues with insufficient modality encoding and limited transfer capabilities. To effectively address these challenges and contextualise our novel contributions, the next Chapter – *Related Work* – provides an in-depth review of existing related work in the literature, specifically focusing on graph architectures, multi-modal, and multi-domain graph-based recommendation approaches, which serve as the three core building blocks for the methodologies presented in this thesis.

Chapter 3

Related Work

As we previously discussed in Chapter 1, in this thesis, we aim to develop expressive graph neural architectures that can more effectively incorporate complex and structured information in top-K recommendation. However, recall that in Section 1.4, different from the recommendation approaches in the literature, the main contributions of this thesis are not only enhancing existing graph neural architectures but also obtaining more supervision signals, sourced from diverse domains and modalities, in order to learn more effective user/item representations in a self-supervised learning (SSL) manner. Therefore, to distinguish our main contributions from the existing work, in this chapter, we explore graph-based recommender systems and highlight their limitations in the literature, focusing on graph neural architectures designed to model the relationships between users and items on user–item interaction graphs (Section 3.1). Next, since we also aim to use multi-modal data and multi-domain interactions between users and items, we discuss, in Sections 3.2 and 3.3, existing approaches that leverage various multi-modal and multi-domain modelling techniques in the top-K recommendation task, and discuss their corresponding limitations. In particular, we focus on discussing the modality fusion and cross-domain transfer along with key baselines used in this thesis, which are typical and commonly used concepts in various graph-based recommendation scenarios. Section 3.4 gives the concluding remarks for this chapter.

3.1 Graph-based Recommenders

In this thesis, we aim to enhance graph-based architectures for the top-K recommendation task. The main objective of applying a graph-based recommender system is to recommend items that the users might be interested in, based on observed similar behaviours from a near-neighbouring user on the user-item interaction graph. As we previously discussed in Section 2.2, Graph Neural Networks (GNNs), such as NGCF (Wang et al., 2019b) and LightGCN (He et al., 2020), have emerged as a dominant method for learning effective *ID representations* in graph-based recommender systems. By aggregating initialised and learned ID embeddings from neighbouring

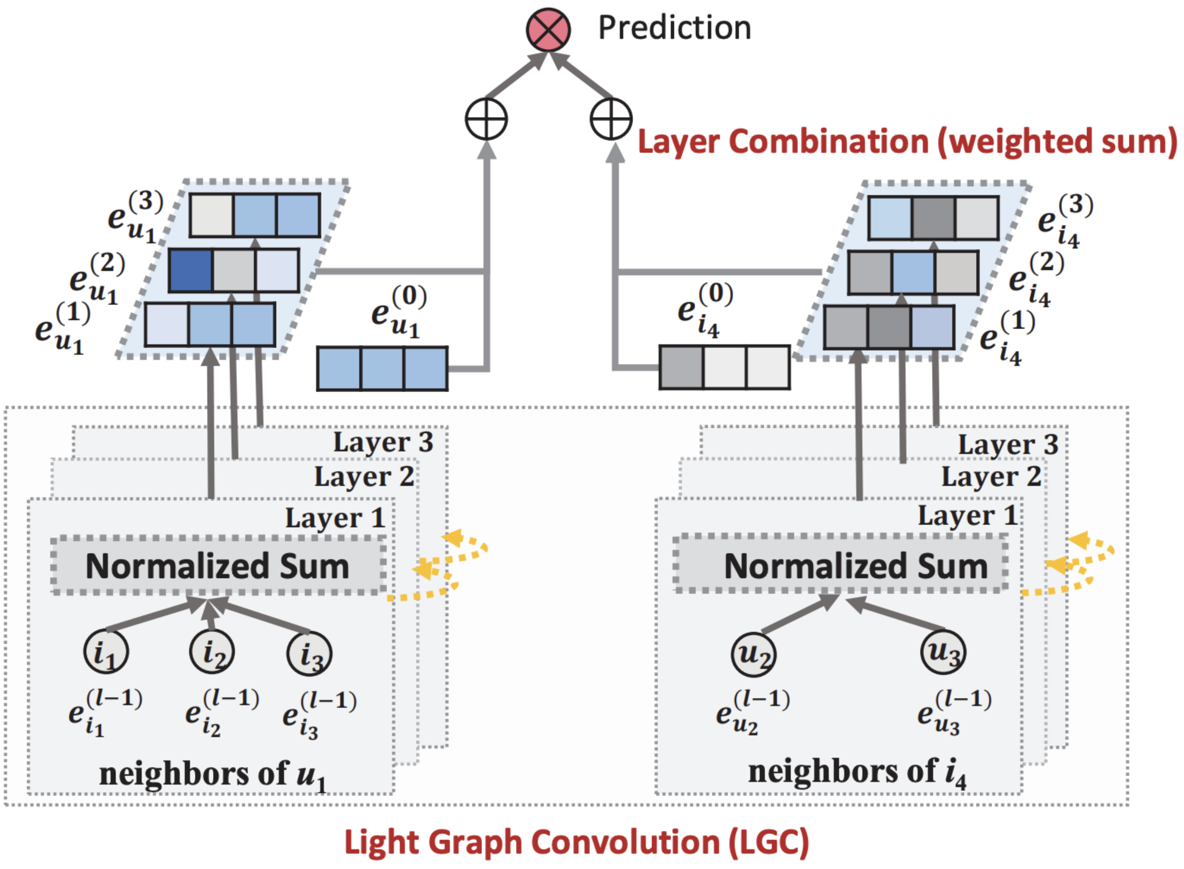


Figure 3.1: The LightGCN model architecture

nodes on the user-item interaction graph, GNN-based models can capture high-order connectivity and collaborative patterns (Zhang et al., 2022b). One prominent example is LightGCN (He et al., 2020), which simplifies the traditional graph neural architecture by removing non-linear activation functions and feature transformation matrices from NGCF, focusing purely on linear propagation over the user-item graph structure. Following the introduction of LightGCN, later approaches (e.g., SGL, SimGCL), particularly those using Self-Supervised Learning (SSL), have adopted this simplified architecture as a critical backbone, demonstrating both its strengths in effectiveness and its inherent limitations in expressiveness. In the next sections, we introduce both conventional graph-based recommender systems and SSL-enhanced graph-based recommender systems in the literature.

3.1.1 Conventional Graph Neural Architecture in Recommender Systems

To illustrate how a typical graph-based recommender system obtains ID user/item embeddings, we use LightGCN (He et al., 2020) as a representative example (as shown in Figure 3.1). Specifically, LightGCN propagates the ID embeddings of the users and items over the user-item interaction graph. The layer-wise propagation function of the ID embeddings is defined as follows:

$$x_{i-id}^{(l_g)} = \sum_{i \in \mathcal{N}_u} \frac{x_{i-id}^{(l_g-1)}}{\sqrt{|\mathcal{N}_i|\,|\mathcal{N}_u|}}, \tag{3.1}$$

where $\mathcal{N}_i$ and $\mathcal{N}_u$ denote the set of neighbours for user u and item i, respectively, while $|\mathcal{N}_u|$ and $|\mathcal{N}_i|$ represent the size of $\mathcal{N}_u$ and $\mathcal{N}_i$, and l_g is the layer number of GNN. The final embeddings are typically obtained by combining representations across layers. LightGCN is characterised by its minimalist design, which omits learned weight matrices from conventional GNNs (c.f. Equation (2.10)) and simplifies the design in the feature propagation component by removing the non-linear activation and bias terms. Indeed, most existing graph-based recommender systems (Wu et al., 2021a; Yu et al., 2023b) use LightGCN as a baseline and the backbone model in the top-K recommendation task (Yang and Toni, 2018). In particular, LightGCN is optimised with the pair-wise BPR (Rendle et al., 2009) loss, as follows:

$$\mathcal{L}_{BPR} = \sum_{(u,i,j) \in D_s} -\log\sigma(\boldsymbol{e_u}^\top(\boldsymbol{e_i} - \boldsymbol{e_j})) \tag{3.2}$$

where $\boldsymbol{e_u}$ and $\boldsymbol{e_i}$ denote user and item embeddings, respectively.

Despite LightGCN's effectiveness comes from its simplicity, this simplicity also exposes its limitations. Since LightGCN only consists of the aggregation component, the only trainable parameters are the initial user/item ID embeddings (i.e., $\Theta = \{\mathrm{E}^{(0)}\}$), as indicated in Equation (3.2). This means that the model complexity is equivalent to standard Matrix Factorisation (MF). This indicates LightGCN, as the most widely used graph-based recommender system, is not capable of modelling more complex relationships that allow for a fine-grained distinction between users and items within the user-item interaction graph, as it lacks learned weight matrices to dynamically transform features during the propagation step. Since the model relies purely on the raw connectivity structure to define relationships, users or items with similar high-order neighbours are often forced into similar latent spaces, even if their underlying preference behaviours differ. Furthermore, repeated linear graph aggregation operations between connected nodes can result in indistinguishable user and item embeddings, leading to the well-known *over-smoothing* problem in GNNs. In this thesis, we argue that a more expressive graph neural architecture is capable of capturing fine-grained differences between users and items by mitigating the over-smoothing problem.

Various studies (Dwivedi and Bresson, 2020; Dwivedi et al., 2022; Kreuzer et al., 2021; Lim et al., 2022; Mialon et al., 2021; Wang et al., 2021e; Ying et al., 2021) have exploited positional encodings on graphs to improve the expressiveness of GNNs. Many earlier studies (Loukas, 2020; Murphy et al., 2019) used index positional encoding to enhance conventional GNNs in terms of their associated model expressiveness. For example, GRP (Murphy et al., 2019) devised a positional encoding by assigning to each node an identifier that depends on the index ordering. This approach can be computationally expensive as it needs to account for all $n!$ node permutations to guarantee a higher expressiveness. Therefore, some prior studies (e.g., (Li et al., 2020b; You et al., 2019)) have applied a more efficient distance positional encoding to enhance the model expressiveness of GNNs. For example, P-GNN (You et al., 2019) enhanced the model expressiveness by projecting the distances between a target node and randomly sampled nodes into a position-aware embedding. However, a large number of sampled nodes will include most of the nodes on the graph, thus leading to insufficient positional embeddings. DEGNN (Li et al., 2020b) modelled a distance positional encoding by capturing distances between nodes using landing probabilities of random walks. However, this approach cannot scale to large-scale graphs because of the cost of computing the power matrices. Alternatively, Laplacian eigenvectors (Belkin and Niyogi, 2003) have been shown to be good candidates for graph positional encoding, since Laplacian eigenvectors form a meaningful local coordinate system while preserving the global graph structure. In particular, we can pre-compute the Laplacian eigenvectors/eigenvalues and provide a unique ID for each node, which solves the scalability issue on the user-item graph in a recommender system and further enhances the pre-existing node features by merging Laplacian eigenvectors/eigenvalues. Another alternative approach used by APPNP (Gasteiger et al., 2018) provided an improved graph feature propagation scheme with Personalised PageRank (Haveliwala, 2002), which particularly addresses the over-smoothing problem in a random walk manner. Therefore, the *Laplacian eigenvectors* and the *random walk operator* are expected to define a new relative positional encoding with a *separate message passing function* (c.f. Section 2.2) in graph-based recommender systems. In this thesis, we aim to investigate the usefulness of these graph positional encoding techniques in graph-based recommender systems for top-K recommendation.

On the other hand, some existing works (Jin and Zhu, 2025; Jung and Park, 2025; Zhang et al., 2024b) suggest that self-supervised learning can further enhance the expressiveness of graph-based models. In the following section, we introduce SSL-based graph recommender methods and discuss potential strategies to enhance the graph neural architecture.

3.1.2 SSL-based Graph Neural Architecture in Recommender Systems

As discussed in Section 2.3, from an architectural perspective, Self-Supervised Learning (SSL) in graph-based recommendation does not directly modify the graph neural network structure, but provides auxiliary supervision signals that enable the learning of more expressive user/item embeddings. SSL has been extensively used in graph-based recommendation to address the sparsity issue within the top-K recommendation task (Yu et al., 2023b). In particular, SSL approaches are able to generate additional supervision signals by maximising the agreement between different representations (views) of the same user/item node, thereby enhancing robustness and representation distinctiveness. Two notable examples are Self-supervised Graph Learning (SGL) (Wu et al., 2021a) and Simplifying Graph Contrastive Learning (SimGCL) (Yu et al., 2023b), both of which build upon the LightGCN architecture:

SGL (Wu et al., 2021a): This SSL approach was introduced to improve the accuracy of graph-based recommenders against noisy interactions by supplementing the classical supervised BPR task with an auxiliary self-supervised learning task (Wu et al., 2021a). This SSL task reinforces user/item node representation learning via self-discrimination by maximising the agreement between multiple views of the same node compared to that of other nodes. As discussed in Section 2.3.3, graph-based augmentations typically manipulate graph structure to obtain more effective views for SSL. Indeed, SGL proposes to augment the user-item graph structure by creating multiple views of the graph through various augmentation techniques, specifically devising two operators: node dropout and edge dropout. Given the user-item graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with adjacency matrix $\mathbf{A}$, the augmented graph $\tilde{\mathcal{G}}$ is generated by directly perturbing $\mathcal{G}$:

- **Edge Dropout:** The augmented graph $\tilde{\mathcal{G}}$ is produced by removing each edge in $\mathcal{E}$ with a probability p, endowing representations with robustness against noisy interactions.

$$\tilde{\mathcal{G}}, \tilde{\mathbf{A}} = \mathcal{T}_{\text{E-dropout}}(\mathcal{G}) = (\mathcal{V}, \mathbf{m} \odot \mathcal{E}) \tag{3.3}$$

  where $\mathbf{m} \in (0,1)^{|\mathcal{E}|}$ is the masking vector on the edge set.

- **Node Dropout:** Similarly, each node (and its associated edges) can be dropped with probability p to identify influential nodes from differently augmented views.

$$\tilde{\mathcal{G}}, \tilde{\mathbf{A}} = \mathcal{T}_{\text{N-dropout}}(\mathcal{G}) = (\mathcal{V} \odot \mathbf{m}, \mathcal{E} \odot \mathbf{m}') \tag{3.4}$$

  where $\mathbf{m} \in (0,1)^{|\mathcal{V}|}$ is the masking vector on the node set and $\mathbf{m}'$ is the vector masking the associated edges.

SGL then uses the LightGCN propagation function (c.f. Equation 3.1) to obtain two different representations, $\mathbf{e}_u^{(1)}$ and $\mathbf{e}_u^{(2)}$, for the same user u using these views. The self-supervised objective, typically the InfoNCE loss (as introduced in Equation 2.11), aims to maximise the consistency between these positive pairs:

$$\mathcal{L}_{SSL} = \sum_{u \in \mathcal{U}} -\log \frac{\exp(\text{sim}(\mathbf{e}_u^{(1)}, \mathbf{e}_u^{(2)})/\tau)}{\sum_{v \in \mathcal{U}} \exp(\text{sim}(\mathbf{e}_u^{(1)}, \mathbf{e}_v^{(2)})/\tau)}, \tag{3.5}$$

where $\text{sim}(\cdot,\cdot)$ is the cosine similarity function and τ is the temperature hyper-parameter. The final objective for SGL is a linear combination of the standard BPR loss ($\mathcal{L}_{BPR}$) and the InfoNCE loss ($\mathcal{L}_{SSL}$). By using this contrastive approach, SGL effectively improves the quality of the learned representations, particularly for long-tail items.

SimGCL (Yu et al., 2022): While SGL relies on explicitly modifying the graph structure, Yu et al. (2022) argued that complex graph augmentation is often computationally intensive, difficult to optimise, and can potentially introduce unnecessary noise. Instead, SimGCL simplifies the self-supervised learning process by introducing representation perturbation using specifically uniform noise addition, as the primary means of generating augmented views. For a user u, SimGCL generates two independent views, $\mathbf{e}_u^{(1)}$ and $\mathbf{e}_u^{(2)}$, by adding uniform noise ($\delta_{u,1}$ and $\delta_{u,2}$) to the final embeddings $\mathbf{e}_u$ derived from the LightGCN encoder:

$$\mathbf{e}_u^{(1)} = \mathbf{e}_u + \delta_{u,1}, \quad \mathbf{e}_u^{(2)} = \mathbf{e}_u + \delta_{u,2}, \tag{3.6}$$

where $\delta_{u,k} \sim \mathcal{U}(-\epsilon, \epsilon)$. These perturbed embeddings are then subjected to the same InfoNCE loss (c.f. Equation (3.5)). SimGCL demonstrated that this simple technique achieves comparable or better performance than complex augmentation methods on the structure of user-item interaction graphs (Yu et al., 2022). This strongly suggests that the core power behind the performance gains lies in SSL to enforce alignment and uniformity in the latent space, rather than the complexity of the view generation mechanism itself.

Both SGL and SimGCL rely fundamentally on LightGCN as their architectural encoder. In particular, they use the efficient, linear message-passing rule defined in Equation (3.1) for their primary propagation step. Since these models maintain the simplicity of LightGCN's propagation, the *only trainable parameters* in these models remain the initial user/item ID embeddings, $\mathbf{E}^{(0)}$. The enhanced performance observed in these SSL-enhanced graph-based recommenders is primarily attributable to the improved quality of the supervision signals generated by the contrastive objective, which effectively enforces representations apart (uniformity) or close together (alignment). SSL acts orthogonally to the expressiveness of the underlying architecture. While SSL improves representation improvements and mitigates the over-smoothing problem by forcing distinctiveness in the final latent space, the underlying graph-based system still lacks learned weight matrices or other components to model complex dependency and relationships during propagation (Li et al., 2024). The architectural incapability to apply transformation weights per propagation step prevents the model from learning to filter or prioritise certain neighbouring signals, making complex dependency modelling impossible. This weakness represents a clear research opportunity to enhance the architecture itself, rather than relying solely on the loss function for model improvement. As such, we discuss a more advanced graph neural architecture with promising expressiveness, the graph transformer architecture, in the following section.

3.1.3 Graph Transformer

The incorporation of the transformer architecture into the graph neural architecture has been proposed as a potential solution in improving graph-based models in graph-related tasks, such as node classification and link prediction (Feng et al., 2023; Hua et al., 2023; Wu et al., 2023). Unlike conventional GNN propagation, *graph transformers* introduce additional learnable feature transformation layers and attention-based neighbour weighting, which enable adaptive information aggregation, long-range dependency modelling, and richer feature interactions across nodes (Yun et al., 2019). This architectural expressiveness allows the model to selectively emphasise informative signals, suppress noise, and capture higher-order structural semantics, rather than treating all aggregated neighbour information uniformly (Yuan et al., 2025).

In the top-K recommendation setting, such adaptive neighbour weighting and learnable feature transformation are particularly beneficial, as implicit user–item interactions are inherently noisy and often contain false positives and false negatives. Traditional graph-based recommender systems propagate information directly along these noisy interaction edges, causing users or items with ambiguous neighbours to be aggregated into similar latent spaces. This capability of selectively denoising and transforming relational information during propagation analogously mirrors the difficulty of denoising the non-relevant information and ambiguities in natural language processing (Zou et al., 2023). A graph transformer architecture offers a more selective denoising mechanism, analogous to disambiguating noisy signals in natural language processing (Zou et al., 2023), by filtering out non-relevant interactions and preserving user-specific preference patterns. For example, GFormer (Li et al., 2023a) is a recently proposed graph transformer model to model implicit interactions, which contrasts the user/item embeddings of a graph encoder and a separate transformer module by selectively dropping out the low-weight edges. Specifically, GFormer uses an attention mechanism to compute node-to-node attention scores that represent the strength or importance of a user-item edge:

$$p((v_k, v_{k'})|\mathcal{G}) = \frac{\bar{\alpha}_{k,k'}}{\sum_{(v_k,v_{k'})\in\mathcal{E}} \bar{\alpha}_{k,k'}}$$

where $\bar{\alpha}$ is the multi-head attention score. As such, a sampled "rationale" graph $\mathcal{G}_R$ (representing the most important interactions) is obtained as follows:

$$\mathcal{G}_R, \mathbf{A}_R = \mathcal{T}_{\text{Rationale}}(\mathcal{G})$$

where $\mathbf{A}_R$ is the adjacency matrix sampled from $p(\cdot|\mathcal{G})$. This forces the transformer to learn a rationale that preserves all necessary information for accurate top-K ranking.

Despite these advantages, the use of a graph transformer architecture for denoising within a diffusion process in the top-K recommendation scenario has not yet been investigated in the literature. A diffusion process first corrupts implicit interactions through a forward noising phase and then iteratively refines user preference representations via a reverse denoising process. In this chapter, we aim to investigate the use of a more effective graph transformer architecture along with the diffusion process to facilitate top-K recommendations.

In summary, the limited expressiveness of current graph-based recommender system architectures, such as LightGCN and its SSL-enhanced variants, fails to distinguish fine-grained differences between user and item nodes within the user-item interaction graph. This shared architectural constraint motivates the necessity for more expressive graph neural architectures. Positional encoding techniques offer a promising direction to enhance the expressiveness of graph-based recommenders by enriching the structural information in a new message passing function. Moreover, graph transformer architectures, with their long-term adaptive neighbour weighting capabilities and learnable feature transformations, are well-suited to noisy or ambiguous implicit user-item interactions. In Chapter 4, we present new graph neural architectures for top-K recommender systems. We incorporate positional encoding within the GNN message passing function to enhance propagation, and introduce a diffusion technique in a graph transformer architecture to evolve the overall model design of graph-based recommender systems along with SSL for top-K recommendation.

3.2 Multi-modal Graph-based Recommenders

In this section, we present and discuss the challenges in multi-modal graph-based recommender systems, namely the equal modality importance in modality fusion, the shallow alignment in modality fusion and the isolation problems in multi-modal recommendation pipeline.

3.2.1 Equal Modality Importance in Modality Fusion

In the multi-modal graph-based recommendation literature, the item images and descriptions encapsulate rich semantic information, such as visual features of an e-commerce product and textual attributes that describe its functionality, brand, and material (Li, 2024). Therefore, many graph-based recommender systems leveraged these multiple modalities from items to create user/item representations and to address the top-K recommendation task. VBPR (He and McAuley, 2016b) is one of the first models to incorporate visual features into recommender systems by concatenating visual embeddings with ID embeddings in the item representation. MMGCN (Wei et al., 2019) further advances this approach by injecting high-order semantics into user/item representation learning through several graph convolutional layers. This method generates aggregated representations for each modality and combines them using either mean or sum operations, resulting in the final fused representations. These models devise augmentations on modality-specific user-item graphs to enhance multi-modal *feature alignment*, enabling to synthesise information across different modalities for a more coherent representation. Another line of approaches effectively mines item-item structures to enhance item representation learning by capturing the underlying relationships and similarities between items. For instance, LATTICE (Zhang Jinghao et al., 2021) constructs item-item graphs for each modality based on the user-item bipartite graphs, performing graph convolutional operations several times on both item-item graphs and user-item interaction graphs to obtain more comprehensive and informative user/item representations, which reflects the complex interactions and dependencies among items and users. This process contributes to aligning multi-modal features by uncovering latent item-item relationships and associating items with similar modality features. However, we argue that multi-modal graph-based recommender systems treat each modality equally (e.g., they

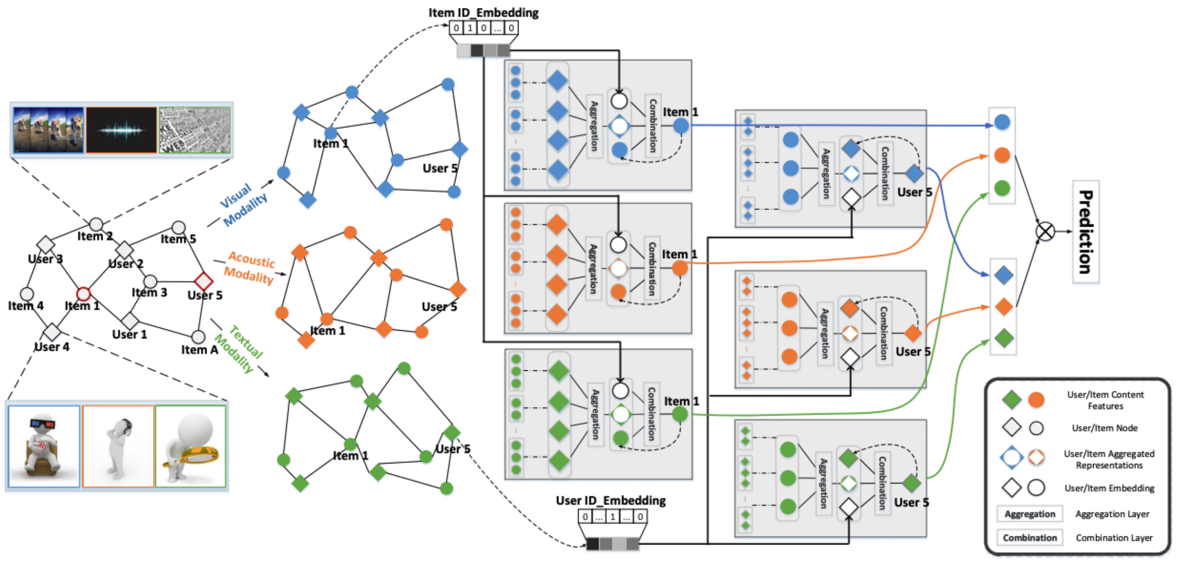


Figure 3.2: An illustration of the MMGCN model architecture.

simply concatenate user/item embeddings from each modality). Recall the discussion of GNN's incapability in fusing diverse features from different modalities in Section 1.1 and Section 2.2, graph-based recommenders lack dedicated methods to effectively fuse multi-modal features for learning richer user/item representations. Consequently, different from the existing recommendation techniques, we introduce a more effective fusion in graph-based recommenders for an improved recommendation performance. In the following, we discuss typical and representative multi-modal graph-based recommenders to show how they encode multi-modal data, so as to learn the users' preferences and the items' attributes for recommendations.

MMGCN: To illustrate the use of multi-modal data in graph-based recommender systems, we use MMGCN (Wei et al., 2019) as an example, which is a representative graph-based recommendation model in the literature. We also consider MMGCN as the baseline to compare to our proposed model, so as to assess the performance of our proposed graph-based recommendation approaches. MMGCN constructed modal-specific user-item interaction graphs using the item images, descriptions and acoustic features, respectively:

$$\mathcal{G}_m = (\mathcal{U} \cup \mathcal{I}, \mathcal{E}, \mathbf{F}_m) \tag{3.7}$$

where $\mathcal{E}$ are the observed interactions, and $\mathbf{F}_m$ is the matrix of content features specific to modality m. We denote $m \in \mathcal{M} = \{v, a, t\}$ as the modality indicator, where v, a, and t correspond to the visual, acoustic, and textual modalities, respectively. Figure 3.2 presents the architecture of the MMGCN model. Given pre-extracted visual, textual and acoustic item features using different feature extractors, MMGCN uses GNNs to aggregate the features of a user's interacted items for each modality. It then concatenates all the user features from each modality along with ID embeddings as the combined user representation. Then, MMGCN trains each modal-specific GNN with a BPR loss (c.f. Equation (2.4) in Section 2.1.2) to learn the user preference to rank the top-K most relevant items of the target user. This confirms that MMGCN is trained by treating each modality with equal importance, and it fails to leverage the correlation between multiple modalities. This might lead to suboptimal representations by overemphasising a particular modality in the learned representations. Moreover, as discussed in Section 3.1.2, SSL typically provide additional supervision signals by generating multiple views by perturbing the user-item bipartite graphs. However, existing SSL approaches based on graphs (e.g., SGL (Wu et al., 2021a)) cannot be directly applied to multi-modal recommendations. We argue that these ID-based approaches are not able to encode item representations with multiple modalities, since the used methods cannot fully disentangle the users' tastes on different modalities. As such, there is a pressing need to explore the correlation between modalities to enhance the multi-modal representation learning via SSL in the top-K recommendation task. We can propose modal-specific graph augmentations to construct effective views within SSL so as to enable the contribution of each modality to user/item node representations during training.

3.2.2 Shallow Alignment in Modality Fusion

On the other hand, feature extraction is crucial in multi-modal recommendation because it enables the identification of meaningful and discriminative information from various modalities, such as textual, visual, and auditory data (Zhou et al., 2023a). By effectively capturing the intrinsic properties of each modality and their relationships, the recommendation models can better comprehend and represent the items and users (Tao et al., 2022). In the multi-modal recommendation literature, various approaches employ different modality-specific encoders to extract features from raw data. VECF (Chen et al., 2019) uses the VGG-19 model (Simonyan and Zisserman, 2015) for the pre-segmentation of images, capturing users' attention on different image regions. VBPR (He and McAuley, 2016b) extracts visual features from item images using a pre-trained Deep CNN (Donahue et al., 2014). MMGCN (Wei et al., 2019) employs the ResNet50 (He et al., 2016) model for visual feature extraction, Sentence2Vector (Le and Mikolov, 2014) for deriving textual features from micro video descriptions, and VGGish (Gemmeke et al., 2017) for learning acoustic features. LATTICE (Zhang Jinghao et al., 2021) uses Deep CNN (Donahue et al., 2014) and Sentence-Transformer (Reimers and Gurevych, 2019) for visual and textual feature extraction, respectively. It then constructs item-item graphs for each modality based on the user-item bipartite graphs, performing graph convolutional operations several times on both item-item graphs and user-item interaction graphs to obtain more comprehensive and informative user/item representations, which reflects the complex interactions and dependencies among items and users. Specifically, LATTICE constructs the modality-specific

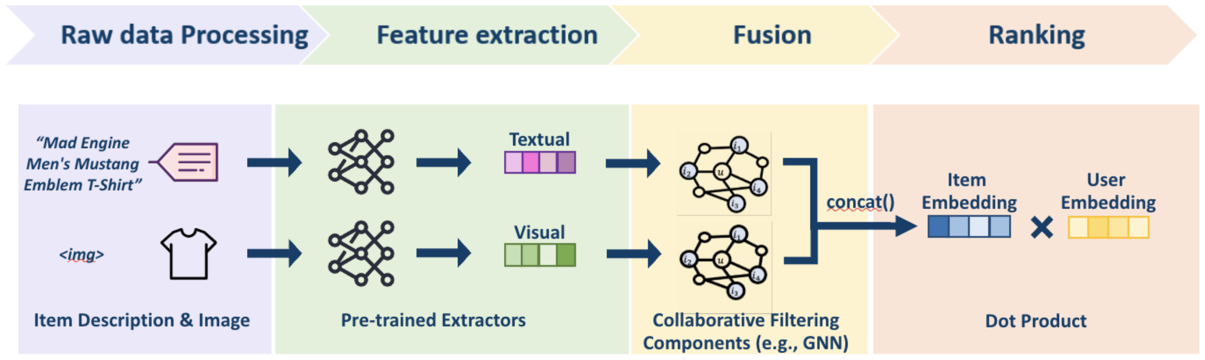


Figure 3.3: A typical working flow in existing multi-modal graph-based recommender systems.

item-item graphs $\tilde{\mathbf{S}}^m$ by calculating feature similarity and applying KNN sparsification:

$$\tilde{\mathbf{S}}_{ij}^m = \begin{cases} \frac{(\mathbf{e}_i^m)^\top \mathbf{e}_j^m}{\|\mathbf{e}_i^m\| \|\mathbf{e}_j^m\|}, & \text{if } \mathbf{S}_{ij}^m \in \text{top-}k(\mathbf{S}_i^m) \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

The resulting latent item-item adjacency matrix $\mathbf{A}$ is then obtained by an adaptive fusion of these adjacency matrices $\mathbf{A}^m$ (the normalised $\tilde{\mathbf{S}}^m$ combined with a learned component) using learnable weights α_m:

$$\mathbf{A} = \sum_{m \in \mathcal{M}} \alpha_m \mathbf{A}^m \tag{3.9}$$

This process contributes to aligning multi-modal features by uncovering latent item-item relationships and associating items with similar modality features. FREEDOM (Zhou and Shen, 2023) further improves LATTICE by simply freezing the item-item graph before training and denoising the user-item interaction graph using a degree-sensitive edge pruning technique from false-positive interactions. However, existing methods primarily rely on the concatenation or combination of static and extracted representations from each modality, thereby only performing a *shallow alignment* for multi-modal fusion, which cannot deeply capture the interrelations among the modalities. To the best of our knowledge, there are no existing approaches that perform deep feature alignment for each modality in top-K multi-modal recommendations, which necessitates comprehensively learning the relationships between modalities to achieve a more effective representation of the multi-modal data. As such, integrating deep alignment methods like CLIP (Radford et al., 2021), VLMo (Bao et al., 2022) or BEiT-3 (Wang et al., 2023c) as a supplementary component into existing multi-modal graph-based recommender systems is worth investigating for more effective modality fusion in top-K multi-modal recommendation.

3.2.3 The Isolation Problems in Multi-modal Recommendation Pipeline

As discussed in Section 3.2.2, existing graph-based recommender systems, such as LATTICE and FREEDOM, used Deeper CNN (Donahue et al., 2014) and Sentence-Transformer (Reimers and Gurevych, 2019) for visual and textual feature extraction, respectively. As shown in Figure 3.3, each multi-modal recommender system leverages features extracted from pre-trained extraction models and employs specific strategies like Graph Neural Network (GNN) for multi-modal fusion. These recommendation models primarily rely on pre-trained extraction models for feature extraction. As a result, the use of pre-trained models for extraction isolates the extraction step from the entire process of top-K multi-modal recommendation. This isolation occurs because the extraction models are not tailored or fine-tuned for the top-K multi-modal recommendation task, thus potentially incorporating non-relevant information that could harm the item representations in a multi-modal recommendation context. We refer this as the *isolated extraction process* in top-K multi-modal recommendation. This isolated extraction process potentially leads to the incorporation of non-relevant information into the subsequent fusion component, thereby impeding the extraction of more effective multi-modal features. Moreover, as illustrated in Figure 3.3, the existing recommender systems apply multiple components (e.g., collaborative filtering components) to process each modality separately before finally fusing them using simple concatenations. We refer this as *isolated modality encoding process* that prevents any interaction between modalities in the working flow of all existing multi-modal graph-based recommender systems. As a result, this separation misses opportunities to jointly optimise the user/item embeddings resulting from the different modalities, thereby potentially reducing the overall recommendation effectiveness. In contrast to aforementioned approaches that use isolated processes for both feature extraction and modality encoding, it is worth investigating an *end-to-end* architecture that simultaneously integrates feature extraction and multi-modal fusion into a unified, end-to-end trainable architecture for the top-K recommendation task.

We also discuss several other representative multi-modal graph-based recommender systems that employ variations on this isolated workflow:

- Similar to SSL methods used in unimodal graph recommenders (like SimGCL), SLMRec (Tao et al., 2022) leverages self-supervised learning with pre-extracted features to generate supervised signals. The model contrasts fine-grained and coarse-grained item embeddings across each modality. The fine-grained embeddings are typically the modality-specific feature representations, while the coarse-grained embeddings are aggregated representations, and the contrastive loss (similar to $\mathcal{L}_{\text{InfoNCE}}$ in Equation (2.11)) is used to ensure consistency between these representations, thereby generating richer self-supervision signals from the multi-modal data. The architecture remains centred on the GNN aggregation component, focusing on enriching the input features via SSL rather than dedicated fusion blocks;

- BM3 (Zhou et al., 2023b) aims to enhance the recommendation performance by improving the latent multi-modal user/item representations through contrastive learning. Specifically, BM3 bootstraps these latent representations by reconstructing the user-item interaction graph. This process ensures that the multi-modal representations learned are highly predictive of the original interaction structure, effectively using the contrastive signal to implicitly fuse and refine the modality-specific embeddings.

In summary, all multi-modal graph-based recommendation approaches involve three main components: 1) feature extraction (e.g. the use of pre-trained extractors); 2) the representation learning of users/items across each modality; 3) the fusion of the aggregated user/item node features. Many studies have proposed developing practical multi-modal graph-based recommendation approaches by varying the implementation of the above three components. However, the equal importance in modality fusion (c.f. Section 3.2.1), shallow alignment in modality fusion (c.f. Section 3.2.2) and the isolated extraction and fusion pipeline (c.f. Section 3.2.3) leads to an incapability of effectively modelling multi-modal semantics of graph-based recommender systems into the top-K recommendation task. In Chapter 5 and 6, we propose new SSL-based multi-modal recommender systems to address the insufficient modality fusion and isolation problems in the multi-modal graph-based recommender systems.

3.3 Cross-domain and Multi-domain Graph-based Recommenders

In this section, we present the main bodies of related work for cross-domain recommendation and multi-domain recommendation – the two primary paradigms for knowledge transfer – with a particular focus on graph-based recommender systems.

3.3.1 Cross-domain Recommendation

In general, cross-domain recommendation (CDR) is an application of transfer learning to recommendation scenarios involving two domains, namely a *source* domain (which provides us with additional useful knowledge, in order to reduce data sparsity and therefore improve recommendations on a separate *target* domain (Yuan et al., 2019; Zeng et al., 2021). The most common paradigm is based on shared users, where user profiles on the source domain are used to permit improved personalisation of recommendations in the target domain. For example, CMF (Tang et al., 2016) – a classical CDR approach – jointly factorises the rating matrices from two domains with a shared global user embedding matrix. Different from CMF, CDMF (Loni et al., 2014) leverages factorisation machines to learns shared latent factors with the interactions between users and items across different domains, so as to estimate user preferences in the target domain. Another approach, CoNet (Hu et al., 2018), introduces a cross-connection unit to transfer the user-item interaction features between two domains. With the rise of Graph Neural Networks (GNNs) in recommender systems, PPGN (Zhao et al., 2019) adopts the GNN model to model the interactions of different domains ($\mathcal{D}_s$ and $\mathcal{D}_t$) as a single, joint interaction graph. This allows PPGN to model the high-order connectivity between users and items on a cross-domain adjacency matrix $\hat{\mathbf{A}}$ in both domains, thereby enabling knowledge transfer with shared

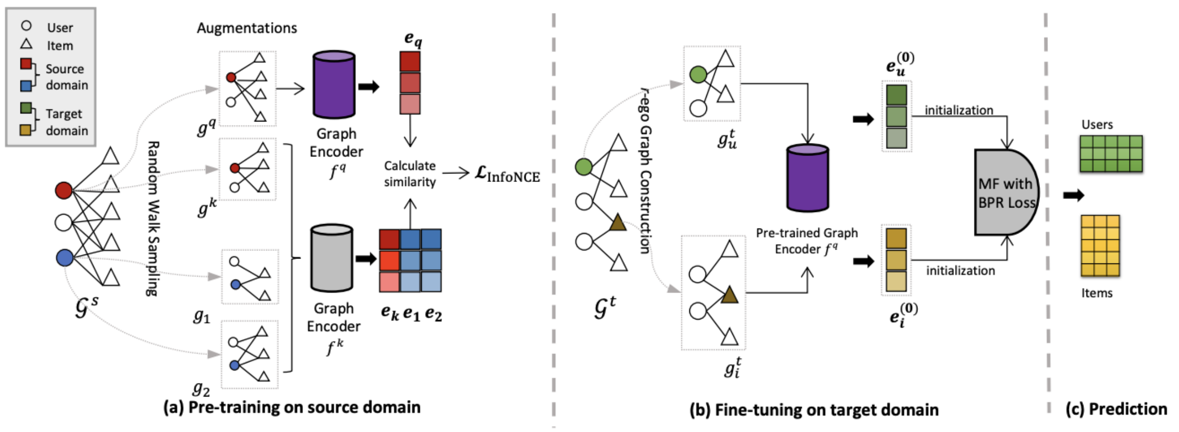


Figure 3.4: A typical first pre-training then fine-tuning in cross-domain graph-based recommender systems.

user features. Specifically, the model propagates information across the combined user/item embeddings $\mathbf{E}^{l-1}$ at layer $l-1$ contains the embedding for all items $\mathbf{E}_{i_s}$ in $\mathcal{D}_s$, all users $\mathbf{E}_u$ of both domains, and all items $\mathbf{E}_{i_t}$ in $\mathcal{D}_s$, where $\mathbf{E}^{l-1} = [\mathbf{E}_{i_s}, \mathbf{E}_u, \mathbf{E}_{i_t}]^\top$. The Graph Convolution and Propagation layer then updates these embeddings for the next layer $\mathbf{E}^l$ using the normalised, unified adjacency matrix $\hat{\mathbf{A}}$ across domains:

$$\mathbf{E}^l = \sigma\left(\hat{\mathbf{A}}\mathbf{E}^{l-1}\mathbf{W}^l + \mathbf{b}^l\right) \tag{3.10}$$

where $\mathbf{W}^l$ and $\mathbf{b}^l$ are trainable parameters. Similarly, BiTGCF (Liu et al., 2020a) is also a joint-learning graph approach that extends LightGCN for cross-domain recommendation by using dual linear graph encoders to generate user and item representations in each domain. A feature transfer layer is then used to fuse user representations across domains, effectively capturing the underlying relationships and facilitating top-K recommendations in the target domain.

Among the CDR approaches, those in the literature (Man et al., 2017; Zhao et al., 2019; Zhu et al., 2022) have focused on pre-training in source domains and fine-tuning in the target domain. For instance, as shown in Figure 3.4, PCRec (Zhao et al., 2019) first pre-trains a graph encoder, which is typically a GNN, on the source domain's interaction data. It then transfers the learned user features to the target domain for an improved performance after fine-tuning the resulting recommendation model. As shown in Figure 3.4, these sub-graphs are then fed into two graph encoders f^{q} and f^{k}. Correspondingly, low-dimensional representative vectors e^{q} and e^{k} are obtained for the positive pair g^{q} and g^{k}, respectively. The graph encoder is then self-supervisedly optimised using the InfoNCE loss (c.f., Equation 2.11). The resulting pre-trained user embeddings are then transferred and integrated into a Matrix Factorisation (MF) model, serving to estimate user preference during the fine-tuning phase in the target domain. However, the existing approaches rely on the pre-trained embeddings and train multiple models in multiple times, resulting in significant time and computational costs associated with the model's parameters.

We argue that there is an effective yet efficient approach that can effectively transfer structural knowledge from the source domain to the target domain in graph-based recommender systems. As a result, a personalised graph prompt-tuning technique can adapt the users' preferences in the target domain using an SSL-based pre-training then prompt-tuning paradigm. In Chapter 7, we address this limitation of insufficient knowledge transfer within the graph-based recommender systems by leveraging SSL-enriched structural information (neighbouring information as graph prompts) across domains.

3.3.2 Multi-domain Recommendation

Shifting from the concept of Cross-Domain Recommendation (CDR), which focuses on transferring knowledge from a data-rich source domain to enhance a target domain, Multi-Domain Recommendation (MDR) systems optimise performance across various domains using a unified model (Zhu et al., 2019). This shift motivates the need for a universal recommendation model that effectively generalises across multiple target domains, thereby simplifying the integration of domain-specific knowledge without the necessity for separate models for each domain (Zhu et al., 2021). For example, MGFN (Zhang et al., 2022a) uses Graph Attention Networks to learn both intra-domain (local) and inter-domain (transferable) knowledge by first refining initial user ID embeddings into context-aware user embedding vectors, $\mathbf{E}_u^k$, for each domain D_k. For a user u in a specific target domain D_{target} (denoted by index t), the final, fused embedding $\mathbf{E}_u'^t$ is computed as:

$$\mathbf{E}_u'^t = \mathbf{E}_u^t + \sum_{j \neq t} \alpha_{t,j} \mathbf{E}_u^j \tag{3.11}$$

where $\mathbf{E}_u^t$ is the user embedding derived from the intra-domain GNN in the target domain. The summation iterates over all source domains $j \in \mathcal{D}$ where $j \neq t$. Each contribution $\mathbf{E}_u^j$ is weighted by an attention weight $\alpha_{t,j}$, quantifying the importance of knowledge transferred from source domains j to the target domain t. Similarly to MGFN, other non-GNN-based MDR approaches, such as MMOE (Ma et al., 2018) and PLE (Tang et al., 2020), also incorporate shared ID-based features across multiple domains and task-specific components for domain adaptation. However, these MDR approaches are affected by domain conflicts where overlapping or contradictory domain characteristics can disrupt the model's learning process. They are also prone to generalisation issues, especially in sparse domains where limited data leads to overfitting (Ning et al., 2023). This limitation is caused by the models' primary reliance on user/item ID embeddings. These embeddings are abstract representations learned solely from sparse interaction data, giving them limited semantic meaning and high domain dependence, which restricts their transferability compared to multi-modal or semantic-based features. To overcome the existing MDR models' inherent limitations and better mine effective supervision signals, we propose a paradigm shift from ID-based to semantic-based universality. We argue that multi-modal large language models (MLLMs) can enable better generalisation by generating universal representations for the multi-modal items' contents across multiple domains. In Chapter 8, we pre-train MLLMs using SSL to generate more generalisable user/item representations and effective model configurations from source domains, and leverage a graph adaptor to further refine and contextualise these embeddings by integrating domain-specific knowledge in target domains.

3.4 Conclusions

In this chapter, we first discussed the widely-used graph-based recommender systems, such as LightGCN, SGL and SimGCL, according to their architectural differences in the top-K recommendation task. In particular, we concluded that, according to the discussion of all graph-based recommender systems, we identified the limited expressiveness of current graph neural architectures that fail to distinguish fine-grained differences between user and item nodes within the user-item interaction graph. Next, we discussed the multi-modal graph-based recommender systems (e.g., MMGCN, SLMRec, LATTICE) and their limitations in modality fusion and the isolated multi-modal recommendation pipeline in top-K multi-modal recommendation. Specifically, we found that existing multi-modal approaches model each modality equally, perform shallow cross-modal modelling and perform isolated feature extraction and fusion processes in the workflow. Moreover, we discussed the cross-domain graph-based recommender systems (e.g., PCRec, PPGN, BiTGCF) and identified that these approaches do not fully exploit the graph structure within user-item interaction graphs across domains. Furthermore, we expanded the discussion from cross-domain recommendation to multi-domain recommendation that requires effective mining of universal representations from generalisable semantic-based features rather than ID-based modelling in top-K multi-domain recommendation. Based on this comprehensive discussion, we identified three key limitations in modern graph-based recommender systems: the limited expressiveness of current graph neural architectures, insufficient modality encoding, and insufficient domain transfer capability. In the next chapter, we begin with our proposed graph-based recommender systems that leverage more expressive and advanced graph neural architectures. Specifically, we investigate how graph positional encodings and a graph transformer architecture can enable more effective graph neural architectures for the top-K recommendation task.

Chapter 4

Self-supervised Graph Architectures

In our thesis statement in Section 1.2, we hypothesised that we can enhance the graph neural architecture by addressing key limitations of conventional graph-based recommender systems, such as the over-smoothing problem and incapability in denoising noisy implicit interactions. As discussed in Section 3.1, current graph neural architectures face two distinct structural challenges. First, the repeated graph aggregation operations between connected user/item nodes lead to the over-smoothing problem, where user and item embeddings become indistinguishable. This issue stems from the conventional graph message-passing function, which lacks effective mechanisms to preserve representation distinction during propagation. Second, while implicitly collected user interactions are abundant, they are inherently prone to interaction noise, often including false-positive and false-negative interactions. Existing architectures typically lack dedicated components to identify and filter this noise, resulting in inaccurate user and item representations. We argue that addressing these limitations through a new message-passing function and a denoising component within the architecture can lead to more expressive graph neural architectures. However, to the best of our knowledge, no study in the literature addressed the over-smooth problem and denoised implicit interactions with graph-based recommender systems. Therefore, in this chapter, we propose two graph-based recommender systems with new graph architectures that address the limitations identified in Section 3.1. The first proposed graph neural architecture includes graph positional encoding techniques (e.g., Laplacian eigenvector) with a new message-passing function to obtain a more expressive graph recommender, thereby also alleviating the over-smoothing problem in an SSL paradigm. The second proposed model leverages a graph transformer architecture alongside a directional diffusion technique to explicitly denoise noisy implicit interactions using SSL, resulting in improved top-K recommendation performance. The remainder of this chapter is structured as follows:

- Section 4.1 presents the first model with a new graph message-passing function, which investigates whether the graph positional encoding techniques can result in a more expressive graph neural architecture in distinguishing the users/items with the same neighbours after propagating several graph convolution layers in the top-K recommendation task;

- Section 4.2 provides details of the second model with a new graph transformer architecture, which leverages a diffusion process that includes a forward phase for gradually introducing noise to implicit interactions, followed by a reverse process to iteratively refine the representations of the users' hidden preferences (i.e., a denoising process);
- Section 4.3 summarises insights gained from our proposed new graph neural architectures in this chapter.

4.1 Self-supervised Graph Positional Representations

As introduced in Section 2.2, Graph Neural Networks (GNNs) provided a strong and fundamental opportunity to develop effective top-K personalised recommendations (Wang et al., 2019b). Specifically, GNNs adopt embedding propagation to aggregate neighbourhood embeddings iteratively through connectivities on a bipartite user-item graph. By stacking the multiple propagation layers, each node on the graph can access high-order neighbours' information through the message-passing scheme (Welling and Kipf, 2016), rather than only modelling the direct interactions between users and items. With their advantages in handling structural data and exploring structural information on a graph, graph-based recommender systems have attained a state-of-the-art recommendation performance (Gao et al., 2022).

Despite the success of graph-based recommender systems, the current graph feature propagation function only repeatedly aggregates neighbourhood embeddings that are adjacent to the target node. As a consequence, the conventional message-passing scheme of the graph-based recommender systems typically fails to differentiate two users with the same interacted items, hence all user representations converge to a constant after propagating several graph convolution layers (Chen et al., 2020). As introduced in Section 3.1, this problem refers to the well-known over-smoothing problem (Li et al., 2018). Indeed, this limitation is now well understood in the context of the equivalence of GNNs with the Weisfeiler-Leman (WL) test (Weisfeiler and Leman, 1968) for graph isomorphism (Morris et al., 2019; Xu et al., 2018), which further confirms the limited expressive power of the current graph-based recommender systems.

Inspired by recent studies (Lee et al., 2021a; Wu et al., 2021a; Xie et al., 2022a; Yu et al., 2021), which have shown the superior ability of Contrastive Learning (CL) to construct supervised signals from correlations within raw data, we also investigate the possibility of leveraging CL to explore the correlations among learned graph positional encodings (c.f. Section 3.1) and address the limited expressive power problem in graph-based recommender systems. A typical approach (Wu et al., 2021a; Yu et al., 2022) to apply CL to recommendations on graphs is to first augment the user-item bipartite graph with noise or structure perturbations, and then to maximise the agreement of the augmented user/item embeddings via a graph encoder. To address the limited expressive power of graph-based recommender systems, we propose a novel recommender system named Positional Graph Contrastive Learning (PGCL) for top-K recommendation, which aims to use graph positional encoding techniques to improve the expressive power of graph-based recommender systems. PGCL aims to enhance the integrated user/item representations through a noise-based augmentation method. To be more specific, PGCL provides additional positional information to existing graph-based recommender systems by injecting the learned graph positional encoding into each feature propagation layer. Furthermore, in order to prevent distorting the users' intents, we apply a noise-based augmentation technique to such position-enhanced user/item embeddings. The latter aims to maintain the users' intent unchanged while adding distance properties to the learned user/item representations. To summarise, we argue that graph-based recommender systems enriched with our proposed graph positional encoding can effectively improve their expressive power using SSL while alleviating the over-smoothing problem. Our PGCL model enforces the divergence of the learned user/item representations in order to improve recommendation performance.

The remainder of this section is organised as follows: In Section 4.1.1, we describe existing graph positonal encoding techniques in general domains. Next, in Section 4.1.2, we introduce the first proposed model architecture, PGCL, which directly addresses the thesis objective of enhancing graph neural architectures for top-K recommendation. We then describe the experimental setup for our proposed model and discuss the obtained experimental results to answer the research questions in Section 4.1.3. This section therefore provides the first concrete step towards validating our thesis claim, before the following chapters extend this direction with more advanced graph architectures.

4.1.1 Graph Positional Encoding

The notion of positional encodings (PEs) in graphs is not a trivial concept, as there exists no canonical way of ordering nodes (Kreuzer et al., 2021). Various studies (Dwivedi and Bresson, 2020; Dwivedi et al., 2022; Kreuzer et al., 2021; Lim et al., 2022; Mialon et al., 2021; Wang et al., 2021e; Ying et al., 2021) have exploited positional encodings on graphs to improve the expressiveness of GNNs. Many earlier studies (Loukas, 2020; Murphy et al., 2019) used index positional encoding to enhance conventional GNNs in terms of their associated model expressiveness. For example, GRP (Murphy et al., 2019) proposed a positional encoding by assigning to each node an identifier that depends on the index ordering. This approach can be computationally expensive as it needs to account for all $n!$ node permutations to guarantee a higher expressiveness. Therefore, some prior approaches (e.g., (Li et al., 2020b; You et al., 2019)) have applied a more efficient distance positional encoding to enhance the model expressiveness of GNNs. For example, P-GNN (You et al., 2019) enhanced the model expressiveness by projecting the distances between a target node and randomly sampled nodes into a position-aware embedding. However, a large number of sampled nodes include most of the nodes on the graph, thus leading to insufficient positional embeddings. DEGNN (Li et al., 2020b) modelled a distance positional encoding by capturing distances between nodes using landing probabilities

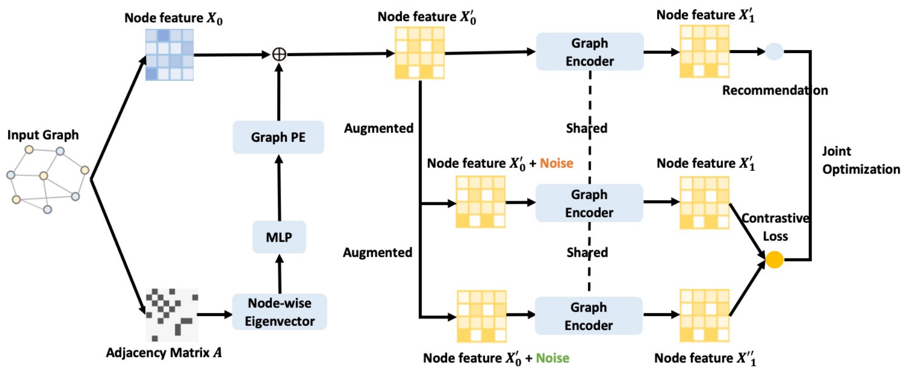


Figure 4.1: The architecture of our PGCL model

of random walks. However, this approach cannot scale to large graphs because of the cost of computing the power matrices. Alternatively, Laplacian eigenvectors (Belkin and Niyogi, 2003) have been shown to be good candidates for graph positional encoding, since Laplacian eigenvectors form a meaningful local coordinate system while preserving the global graph structure. In particular, we can pre-compute the Laplacian eigenvectors/eigenvalues and provide a unique representation for each node, which solves the scalability issue on the user-item graph in a recommender system and further enhances the pre-existing node features by merging Laplacian eigenvectors/eigenvalues. Another alternative approach used by APPNP (Gasteiger et al., 2018) provided an improved graph feature propagation scheme with Personalised PageRank (Haveliwala, 2002), which particularly addresses the over-smoothing problem in a random walk manner. In section, we leverage the Laplacian eigenvectors and the random walk operator to define a new relative positional encoding in recommender systems. Unlike the above prior works, we allocate the learned positional encodings to a separate message-passing function to generate the user/item positional embedding from the neighbours' positional information (c.f. Section 3.1).

4.1.2 Positional Graph Contrastive Learning

In this section, we describe our proposed PGCL model, the architecture of which is illustrated in Figure 4.1. Next, we define the graph positional encoding and illustrate the motivation of decoupling the positional encoding from the conventional message-passing function. We then apply SSL-based contrastive learning for effectively learning the positional encoding and optimise our PGCL model jointly with a pairwise ranking loss.

4.1.2.1 Task Definition

In this section, we focus on addressing the top-K recommendation task. Conceptually, we consider a recommender system with a user set $\mathcal{U}$ and an item set $\mathcal{I}$. In order to facilitate the description of graph-based recommenders, we use $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to denote an interaction graph, where the node set $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ includes all users and items. $\mathcal{E}$ is the set of edges. e_u denotes the user feature for user u and e_i denotes the item feature for item i. In addition, p_u and p_i denote the positional feature of the user and item, respectively. The layers are indexed by ℓ, where $\ell = 0$ denotes the input layer. For a given user u or item i, there is a positional feature p_u or p_i on an interaction graph $\mathcal{G}$. We aim to estimate the users' preferences through a graph encoder f, which can recommend the top-K items for a target user u.

4.1.2.2 Definition of Initial Graph Positional Encoding

As mentioned Section 4.1.1, we aim to use the Laplacian eigenvectors and the random walk operator to define the graph Positional Encoding (PE) in recommendation. In this section, we define the initial positional encoding of a given user u or item i.

Laplacian PE: Laplacian PE (LapPE) is a spectral technique that embeds graphs into an Euclidean space, and is defined via the factorisation of the graph's Laplacian $\Delta = \mathrm{I} - D^{-1/2}AD^{-1/2} = U^T\Lambda U$, where I is the identity matrix, A is the adjacency matrix, D is the degree matrix, and matrices Λ and U correspond to the Laplacian eigenvalues and Laplacian eigenvectors of a graph, respectively. In this work, we consider the Laplacian eigenvector as the initial graph positional encoding, which is defined as follows:

$$p_i^{\text{LapPE}} = [U_{i1}, U_{i2}, \cdots, U_{ik}] \in \mathbb{R}^k \tag{4.1}$$

As a consequence, LapPE is expected to provide a unique ID for each user/item representation and is distance-sensitive w.r.t. the Euclidean norm.

Random Walk PE: Apart from LapPE, we also investigate the use of a random walk-based method (Dwivedi et al., 2022) to generate the graph positional encoding. Hence, we use Random Walk PE (RWPE), which is a method based on the random walk diffusion process. Formally, RWPE is defined with k-steps of random walks as follows:

$$p_i^{\text{RWPE}} = \left[\mathrm{RW}_i, \mathrm{RW}_i^2, \cdots, \mathrm{RW}_i^k\right] \in \mathbb{R}^k \tag{4.2}$$

where $\mathrm{RW} = AD^{-1}$ is the random walk operator. As such, RWPE provides a unique node representation under the condition that each user/item has a unique k-hop topological neighbourhood (Li et al., 2020b) for a sufficiently large k.

Finally, the initial graph PE of the network is obtained by projecting LapPE or RWPE into a d-dimensional feature vector with a Multi-Layer Perceptron (MLP) network:

$$p_i^{\ell=0} = \mathrm{MLP}\left(p_i^{\mathrm{PE}}\right) = W^0 p_i^{\mathrm{PE}} + b^0 \in \mathbb{R}^d, \tag{4.3}$$

where $W^0 \in \mathbb{R}^{d\times k}$ and $b^0 \in \mathbb{R}^d$ are the learned parameters of the MLP network. As illustrated in Figure 4.1, we leverage graph PE (i.e., LapPE or RWPE) to generate the initial positional encoding through an MLP network.

4.1.2.3 Feature Propagation with Learned Positional Encoding

Figure 4.1 illustrates how PGCL concatenates the graph PE and the pre-existing node feature X' which is generated by user/item IDs. As discussed in Section 4.1.1, we aim to decouple the graph PE from the conventional message-passing function. Hence, we propose a message-passing function for the graph PEs. The layer update equation is defined as follows:

$$p_u^{\ell+1} = \text{Tanh}\left(\text{AGG}\left(p_u^\ell, \{p_i^\ell\}_{i\in\mathcal{N}_u}\right)\right), \tag{4.4}$$

where Tanh is the activation function, and AGG is an aggregation function that combines the positional information of the adjacent item nodes. Since the graph positional encoding is interpreted as a unique positional ID of a given node (see Section 4.1.1), we expect to use the message-passing scheme to exploit high-order positional information of the positional features through the aggregation operation. Next, we aim to integrate the graph PE p_i into user representations. Analogously, we can obtain the updated positional embeddings of the items.

As illustrated in Figure 4.1, we concatenate the graph PE – which is generated by Equation (4.4) – with the existing node feature X, similar to the Transformers (Vaswani et al., 2017) network structure:

$$e_u^{\ell+1} = \text{AGG}\left(\begin{bmatrix} e_u^\ell \\ p_u^\ell \end{bmatrix}, \left\{\begin{bmatrix} e_i^\ell \\ p_i^\ell \end{bmatrix}\right\}_{i\in\mathcal{N}_u}\right) \tag{4.5}$$

where e_u and e_i denote the representations of user u and item i, respectively, $\mathcal{N}_u$ is the neighbourhood of the user u, and p_u & p_i denote the position representations of user u and item i, respectively. We use LightGCN (c.f. Section 3.1) to aggregate the concatenated result of the pre-exiting item feature e_i and with the item positional feature p_i. Hence, the feature propagation equation is defined as follows:

$$\mathbf{e}_u^{\ell+1} = \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} \begin{bmatrix} e_i^\ell \\ p_i^\ell \end{bmatrix}, \tag{4.6}$$

$$\text{with } p_i^\ell = \text{Tanh}\left(W_1 p_i^{\ell-1} + \sum_{i\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u|\,|\mathcal{N}_i|}}\left(W_1 p_u^{\ell-1} + W_2\left(p_u^{\ell-1} \odot p_i^{\ell-1}\right)\right)\right) \tag{4.7}$$

where W_1 and W_2 are trainable weight matrices. The main difference of our feature propagation layer with the standard graph-based recommenders is a separated message-passing function for the graph PE, which injects the graph PE into each propagation layer. For this reason, we expect PGCL to provide less over-smoothed user/item embeddings, thereby alleviating the over-smoothing problem of the existing graph-based recommenders.

4.1.2.4 Self-augmented Learning

As discussed in Section 4.1.1, since a graph perturbation has the possibility to distort the user-item bipartite graph, applying a representation-level augmentation on the learned graph PE is more rational than perturbing the graph structure. Following Yu et al. (2022), we apply a noise-based augmentation on the representation level for both the integrated user and item embeddings. For example, given a user embedding e_u, which integrates its graph positional feature p_u, we can generate an augmented user representation by adding a noise vector Δ_u as follows:

$$\mathbf{e}_u' = \mathbf{e}_u + \Delta_u', \mathbf{e}_u'' = \mathbf{e}_u + \Delta_u'',\ e_u \in \mathbb{R}^d \tag{4.8}$$

$$\text{with } \Delta_u = e_x \odot \operatorname{sign}(\mathbf{e}_u) \odot \epsilon,\ e_x \in \mathbb{R}^d \sim U(0,1) \tag{4.9}$$

where $\mathbf{e}_u'$ and $\mathbf{e}_u''$ are two augmented user representations, e_x is a vector that is generated by random numbers from a uniform distribution, and ϵ is a hyper-parameter to control the strength of the user representation perturbation with a range in $[0,1]$. Goodfellow et al. (2015) have also shown that a linear perturbation in high-dimensional spaces can generate sufficient samples. As such, in addition to applying the noise-based augmentation, we also aim to enforce that the integrated user/item representations further spread out in the entire embedding space so as to fully exploit the expressive power of the embedding space. In the graph contrastive recommendation scenario (Wu et al., 2021a; Yu et al., 2022), the target is to generate a better user/item representation via data augmentations. Hence, we use InfoNCE (c.f. Equation (2.11)), to maximise the agreement of two augmented representations:

$$\mathcal{L}_{cl}^{user} = -\log \frac{\exp\left(\mathbf{e}_u'^{\top}\mathbf{e}_u''/\boldsymbol{\tau}\right)}{\sum_{i=1}^{n}\exp\left(\mathbf{e}_u'^{\top}\mathbf{e}_n/\boldsymbol{\tau}\right)} \tag{4.10}$$

where $\mathbf{e}_n$ is the embedding of a different user, and τ is a hyper-parameter that adjusts the dynamic range of the resulting loss value. Analogously, we can calculate the contrastive loss of a target item $\mathcal{L}_{cl}^{item}$. Therefore, we obtain a combined contrastive loss that acts as an auxiliary loss for a top-K recommendation task as follows: $\mathcal{L}_{cl} = \mathcal{L}_{cl}^{user} + \mathcal{L}_{cl}^{item}$. To better mine the user/item representations in recommendation, we adopt a multi-task training strategy to jointly optimise the widely used pair-wise ranking objective, namely BPR (see Equation (2.4) in Section 2.1.2), and the contrastive learning objective $\mathcal{L}_{cl}$.

$$\mathcal{L} = \lambda_1\mathcal{L}_{cl} + \lambda_2\mathcal{L}_{BPR} \tag{4.11}$$

where the second term is the BPR loss, while λ_1 and λ_2 are hyper-parameters to control the strengths of the contrastive learning and the BPR loss, respectively. Through propagating the integrated user/item representations in multiple feature propagation layers with Equation (4.6), we obtain multiple user/item embeddings from each layer, then we concatenate each user/item

Table 4.1: Statistics of the used datasets.

Dataset	Users	Items	Interactions	Density
Gowalla	39, 657	31, 211	1,072,325	0.087%
Yelp2018	28,361	43,142	1,481,472	0.121%
Amazon-Kindle	116,417	72,439	1,643,646	0.019%

embedding e_u^ℓ, so that the final embedding collectively contains information from each layer. Hence, we can estimate the relevant score between a user and item by minimising the multi-task learning loss in Equation (4.11).

4.1.3 Experiments

We now examine the performance of PGCL through experiments on three real-world datasets, in comparison to four existing state-of-the-art graph recommendation models. To demonstrate the effectiveness of PGCL, we conduct experiments to answer the following research questions:

- **RQ4.1**: How does the PGCL model perform in top-K recommendation compared with existing baselines?
- **RQ4.1**: How do different positional encodings and augmentation methods impact the recommendation performance?
- **RQ4.1**: Is our PGCL model more expressive than the conventional graph neural architecture LightGCN thereby alleviating the over-smoothing problem compared to the baselines based on LightGCN?

4.1.3.1 Datasets and Experimental Settings

We evaluate our PGCL model using three real-world datasets, namely *Yelp2018*[1], *Gowalla*[2] and *Amazon-Kindle*[3]. Table 4.1 shows the statistics of these datasets. Following He et al. (2020) and Wang et al. (2019b), we randomly split the above datasets into training, validation, and testing sets with a 7:1:2 ratio. We use two commonly used evaluation metrics: Recall@K and NDCG@K (c.f. Section 2.4) to evaluate the performance of top-K recommendation. We follow Yu et al. (2022) in setting K = 20 and report the average performance achieved for all users in the testing set. We use the Adam (Kingma and Ba, 2014) optimiser in both our PGCL model and the four baseline models. We apply early-stopping during training, terminating the training when the validation loss does not decrease for 50 epochs. To determine the hyper-parameters in both PGCL and the baseline models, we apply a grid search on the validation

[1] `https://www.yelp.com/dataset` [2] `https://snap.stanford.edu/data/loc-gowalla.html`
[3] `https://jmcauley.ucsd.edu/data/amazon/`

set. Specifically, we tune our PGCL model by varying the learning rate in $\{10^{-2}, 10^{-3}, 10^{-4}\}$. The learning rates of the baseline models are also tuned according to the suggested ranges in (He et al., 2020), for a fair comparison. Similarly, we also tune each of λ_1, λ_2 and ϵ within the range of $\{0, 0.1, 0.2, ..., 1.0\}$. A detailed analysis of the models' performance with different layer settings is shown in Section 4.1.3.5.

4.1.3.2 Baselines

We compare the effectiveness of PGCL[4] with four existing strong baselines. Specifically, two classical graph-based models, namely NFCG and LightGCN, are introduced in Section 3.1.1. In addition, SGL and SimGCL are SSL-enhanced graph-based recommendation baselines introduced in Section 3.1.2. In addition, to examine the effectiveness of the Laplacian positional encoding (see Equation (4.1)), we compare PGCL to a variant called PGCL$_{w/oCL}$. Different from PGCL, PGCL$_{w/oCL}$ only concatenates the positional encoding (from Equation (4.7)) with the pre-existing users/items' features from LightGCN, without applying contrastive learning (c.f. Equation (2.11)).

4.1.3.3 Performance Comparison with Baselines (RQ4.1)

Table 4.2 compares our proposed PGCL model with four used baselines: NGCF, LightGCN, SGL and SimGCL (as described in Section 3.1). From the table, we observe that for all three datasets, PGCL outperforms all the baseline models on all metrics, and statistically significantly in most cases according to the paired t-test with Holm-Bonferroni correction ($p < 0.01$). This result demonstrates the rationality and effectiveness of injecting graph PE to the graph feature propagation layer and incorporating the augmented positional and pre-existing node features (i.e. IDs). For a given GNN-based method (NGCF, LightGCN, PGCL$_{w/oCL}$, PGCL), we evaluate the usefulness of leveraging the graph PE in enriching the user/item representations. Comparing NGCF, LightGCN and PGCL$_{w/oCL}$, we observe that PGCL$_{w/oCL}$ performs generally better than both NGCF and LightGCN on all three used datasets. This result demonstrates the benefit of injecting the learned positional encoding into the pre-exiting users/items' features to estimate the users' preferences. On the other hand, as can be observed in Table 4.2, PGCL$_{w/oCL}$ performs worse than PGCL on all three used datasets. This result illustrates the importance of contrastive learning in providing additional supervised signals during training. For the contrastive learning method, we also evaluate the usefulness of different augmentations by comparing our PGCL model with SGL and SimGCL. Table 4.2 shows that the noise-based methods (SimGCL, PGCL) markedly outperform the dropout-based method (SGL) on both the Yelp2018 and Amazon-Kindle datasets, but perform comparably on the Gowalla dataset. This result shows the marginal effect of graph perturbation and the effectiveness of using noise-based

[4] Source code is available at: `https://github.com/zxy-ml84/PGCL`

Table 4.2: Experimental results for PGCL in comparison to other baselines. The best performance is highlighted in bold and the second-best result is highlighted with an underline. * denotes a significant difference compared to the result of PGCL using the paired t-test with the Holm-Bonferroni correction for $p < 0.01$.

Dataset	Yelp2018		Gowalla		Amazon-Kindle	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
NGCF	0.0502^*	0.0417^*	0.0889^*	0.0592^*	0.1893^*	0.1285^*
LightGCN	0.0542^*	0.0437^*	0.0996^*	0.0635^*	0.2230^*	0.1644^*
SGL	0.0563^*	0.0449^*	<u>0.1071</u>	<u>0.0671</u>	0.2331^*	0.1726^*
SimGCL	<u>0.0577</u>	<u>0.0466</u>	0.1068^*	0.0664^*	<u>0.2425</u>	<u>0.1801</u>
PGCL$_{w/oCL}$	0.0581^*	0.0477^*	0.1059^*	0.0675^*	0.2420^*	0.1803^*
PGCL	**0.0608**	**0.0501**	**0.1122**	**0.0699**	**0.2572**	**0.1934**
%Improv.	5.37%	7.51%	4.76%	4.17%	6.06%	7.38%

augmentation. Moreover, Table 4.2 also shows that PGCL outperforms SimGCL by a large margin on all metrics (significantly on Gowalla), which demonstrates that graph PE can enrich the user/item representations as an additional feature. Hence, in answer to RQ4.1, we conclude that our proposed PGCL model can effectively leverage both the graph positional feature and the augmented user/item representations, thereby enhancing the existing graph-based recommender models with significant performance improvements.

4.1.3.4 Ablation Study (RQ4.2)

To investigate the impact of each component of our PGCL model and different graph positional encodings (PE), Table 4.3 shows how the performance of PGCL changes when we start with LightGCN as the basic graph encoder and apply graph PEs (denoted as $LapPE$ and $RWPE$ in Section 4.1.2.2) on top of it so as to conclude on the effectiveness of graph PE and contrastive learning. Table 4.3 shows that the PGCL$_{LapPE}$ variant, which uses the Laplacian eigenvalue and a representation level augmentation achieves the best performance on all datasets. This promising result is due to the addition of the unique learned positional features of the users/items and an effective representation learning on the users/items' embeddings. Specifically, we observe from Table 4.3 that both LightGCN$_{RWPE}$ and LightGCN$_{LapPE}$ achieve a better effectiveness than LightGCN. This result demonstrates the effectiveness of graph PE. One possible reason is that the graph PE denotes a unique positional information to the user/item embedding in each feature propagation layer. For the PGCL$_{RWPE}$ and the LightGCN$_{RWPE}$ variants, which use a random walk operator in Table 4.3, there is a performance reduction compared with PGCL$_{LapPE}$ and LightGCN$_{LapPE}$, which indicates that a global ID (LapPE) is more beneficial than a local ID (RWPE) for the user-item interaction data in recommender systems. Moreover, we also observe that there is an effectiveness improvement from LightGCN$_{LapPE}$ to PGCL$_{LapPE}$. This suggests that both the PE and the pre-existing users/items' features provide an additional supervised signal through contrastive learning. Hence, in answer to RQ4.2, we conclude that PGCL

Table 4.3: PGCL performance in terms of Recall@20 and NDCG@20 on the used datasets. * denotes a significant difference compared to the result of PGCL using the paired t-test with the Holm-Bonferroni correction for $p < 0.01$.

Dataset	Yelp2018		Gowalla		Amazon-Kindle	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
LightGCN	0.0542^*	0.0437^*	0.0996^*	0.0635^*	0.2230^*	0.1644^*
LightGCN$_{RWPE}$	0.0556^*	0.0458^*	0.1014^*	0.0631^*	0.2303^*	0.1713^*
LightGCN$_{LapPE}$	0.0581^*	0.0477^*	0.1059^*	0.0675^*	0.2420^*	0.1803^*
PGCL$_{RWPE}$	0.0583^*	0.0486^*	0.1068^*	0.0674^*	0.2451^*	0.1815^*
PGCL$_{LapPE}$	**0.0608**	**0.0501**	**0.1122**	**0.0699**	**0.2572**	**0.1934**

successfully leverages graph positional encodings to learn effective user/item representations in a contrastive learning scheme.

4.1.3.5 The Over-smoothing Problem (RQ4.3)

After showing that PGCL is effective in improving LightGCN, we now study the characteristics of graph PE in terms of their usefulness against over-smoothing. In this section, we investigate the over-smoothing problem by comparing PGCL and LightGCN with different layer settings in Table 4.4. As shown in Table 4.4, both PGCL and LightGCN reach their best effectiveness within 5 graph layers. In addition, all PGCL variants outperform LightGCN under different layer settings on all used datasets. The largest improvements are observed on the Amazon-Kindle dataset where PGCL can remarkably improve LightGCN by 15.6% on Recall and 17.9% on NDCG with a 4-layer setting. Specifically, PGCL continues to reach a higher recommendation performance on the Gowalla and Amazon-Kindle datasets with more layers while LightGCN already reaches its peak performance at 3-layer. This result indicates that injecting the learned graph positional encoding (PE) can benefit the general message-passing scheme by encoding the graph PE as additional features and can improve the expressive power of LightGCN with an increased models' depth.

To further examine the effectiveness of the graph PE, we conduct a further analysis on the over-smoothness values for both the 2-layer PGCL and all 2-layer baselines. We use the over-smoothness of second-order embedding to evaluate the PGCL's capability of alleviating the over-smoothing problem. Following He et al. (2020), we calculate the users' over-smoothness that have an overlap on the interacted items. In particular, as in (He et al., 2020), we use a smoothness metric to evaluate the over-smoothness of the users/items. A higher value indicates less over-smoothing. Similarly, we can also obtain the over-smoothness for the item embeddings. Table 4.5 shows the over-smoothness values of PGCL and the various used baseline models. The results show that our PGCL model obtains the largest O-Smoothness$_u$ and O-Smoothness$_i$ values, which indicate that more effective user/item embeddings are generated with the learned graph PE. Comparing LightGCN and PGCL$_{w/oCL}$, we note that the graph positional encoding

Table 4.4: Performance comparison between PGCL and LightGCN at different layers. The peak performance for each method is highlighted in bold.

Dataset		Yelp2018		Gowalla		Amazon-Kindle	
Layers	Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
1 Layer	LightGCN	0.0531	0.0433	0.0982	0.0622	0.2214	0.1635
	PGCL	0.0570 (+7.3%)	0.0477 (+10.2%)	0.109 (+11.0%)	0.0677 (+8.8%)	0.2553 (+15.3%)	0.1917 (+17.2%)
2 Layers	LightGCN	0.0519	0.0421	0.0993	0.0630	0.2225	0.1641
	PGCL	0.0582 (+12.1%)	0.0493 (+17.1%)	0.1106 (+11.4%)	0.0686 (+8.9%)	0.2561 (+15.1%)	0.1921 (+17.1%)
3 Layers	LightGCN	0.0536	0.0435	**0.0996**	**0.0635**	**0.2230**	**0.1644**
	PGCL	0.0580 (+8.2%)	0.0488 (+12.2%)	0.1112 (+11.6%)	0.0692 (+9.0%)	0.2565 (+15.0%)	0.1927 (+17.2%)
4 Layers	LightGCN	**0.0542**	**0.0437**	0.0991	0.0632	0.2224	0.1640
	PGCL	0.0595 (+9.8%)	0.0496 (+13.5%)	0.1115 (+12.5%)	0.0697 (+10.3%)	**0.2572 (+15.6%)**	**0.1934 (+17.9%)**
5 Layers	LightGCN	0.0538	0.0427	0.0987	0.0630	0.2217	0.1637
	PGCL	**0.0608 (+13.0%)**	**0.0501 (+14.6%)**	**0.1122 (+13.7%)**	**0.0699 (+11.0%)**	0.2562 (+15.6%)	0.1925 (+17.6%)

exhibits a large gain on O-Smoothness$_u$ and O-Smoothness$_i$ while improving the recommendation performance at the same time. From in Table 4.5, PGCL outperforms PGCL$_{w/oCL}$ both in over-smoothness and recommendation performance by a large margin, which demonstrates the effectiveness of mining augmented user/item embeddings through contrastive learning. Hence, in answer to RQ4.3, we conclude that PGCL successfully alleviates the over-smoothing problem by injecting the learned graph positional encoding to each feature propagation layer. This further shows that a graph positional encoding learned with a separate message-passing function can lead to a more expressive graph-based recommender.

Table 4.5: Over-smoothness comparison of the 2-layer user/item embeddings between PGCL and the baselines. O-Smoothness$_u$ and O-Smoothness$_i$ represent the over-smoothness of users/items, respectively (He et al., 2020). A higher over-smoothness value indicates less over-smoothing (i.e. a higher value is better).

Dataset	Yelp2018			Gowalla		
Methods	O-Smoothness$_u$↑	O-Smoothness$_i$↑	Recall@20↑	O-Smoothness$_u$↑	O-Smoothness$_i$↑	Recall@20↑
LightGCN	10747.4	8318.5	0.0542	14634.6	6314.2	0.0996
SimGCL	12187.5	9932.3	0.0577	15043.1	6939.1	0.1068
PGCL$_{w/oCL}$	13317.8	10177.4	0.0581	15257.4	7192.5	0.1063
PGCL	13978.4	11748.1	**0.0608**	16462.3	7936.8	**0.1122**

4.1.4 Discussion

In this section, we have explored the effectiveness of the proposed Laplacian PE and Random Walk PE in enhancing the expressiveness of the graph neural architecture with an improved top-K recommendation performance. These findings provide the first empirical evidence of the architectural component of our thesis statement: the graph neural architecture can be strengthened by addressing key limitations of conventional graph-based recommender systems, such as the over-smoothing problem, in an SSL manner. In particular, the first part of this chapter confirms that integrating graph positional encodings into the message-passing function effectively mitigates the over-smoothing problem. When trained with an SSL loss, this architecture generates highly distinguishable user/item embeddings. This demonstrates that enhancing the foundations of graph propagation via SSL directly validates our thesis aim of developing more expressive graph neural architectures.

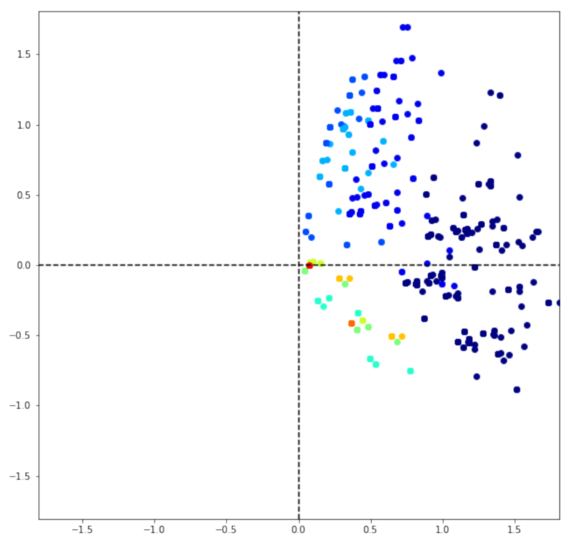
However, as discussed in Section 2.1, improvements at the message-passing level do not fully address the architectural limitations of graph-based recommender systems. Existing models still lack dedicated components to selectively filter or prioritise neighbour information, particularly in the presence of noisy implicit interactions. A new graph neural architecture at a higher architectural level, such as the integration of a transformer component, represents a complementary direction that extends beyond inherent message-passing enhancements while working toward the same thesis goal of expressive graph neural architectures for top-K recommendation. As a consequence, for the next section of this chapter, we propose a new graph neural architecture along with an SSL-enhanced diffusion technique that explicitly denoises the implicit interactions in the top-K recommendation task.

4.2 Self-supervised Graph Transformer

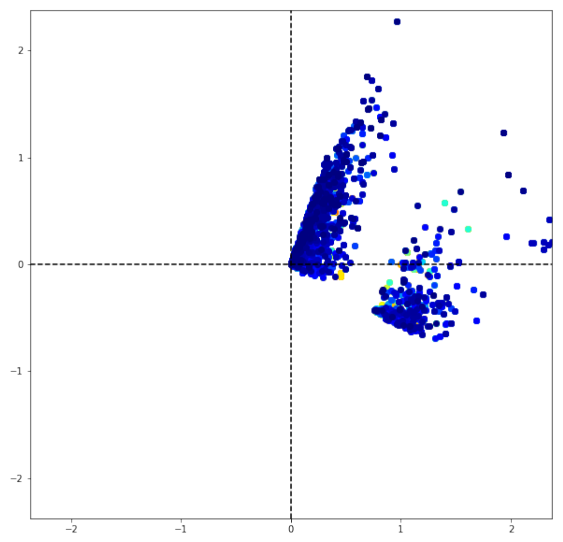
In this section, we propose a novel graph neural architecture where a transformer is paired with a graph encoder (i.e., a GNN) to explicitly denoise the noisy user/item embeddings derived from implicit interactions. As we discussed in Section 2.1.1, unlike explicit interactions, where users provide intentional input like ratings or reviews, implicit interactions represent the users' preferences in a one-dimensional, positive-only format (Wu et al., 2016). However, these signals often encompass noisy interactions, which makes the accurate interpretation of user intent significantly more challenging. For example, a high volume of clicks may not necessarily indicate an actual intent to purchase items. Similarly, a purchase might not always reflect satisfaction, as indicated by instances where purchases are followed by negative reviews. These discrepancies can frequently be attributed to biases in the presentation of items, such as the position bias, which considerably influences the users' initial impressions and their subsequent interactions with the items (Wang et al., 2021c). As discussed in Section 3.1, this notable gap between the user interactions and the actual user satisfaction raises a critical challenge: implicit interactions, while rich in collaborative information, often fail to fully capture the users' preferences due to the prevalence of noisy data, which in turn harms the recommendation accuracy. Consequently, uncovering the users' true preferences amidst this inherent noise becomes crucial.

Extracting more effective collaborative signals from implicit interactions is essential for enhancing the performance of the top-K recommendation task (Liang et al., 2018; Wang et al., 2021c, 2023e; Wu et al., 2016). The recent DiffRec model (Wang et al., 2023e) leveraged a diffusion model with an autoencoder architecture to denoise the implicit interactions. Conceptually, diffusion models gradually corrupt the user-item interactions in a tractable forward process and learn the reverse reconstruction iteratively in the direction of recovering the original interactions (Wang et al., 2023e). In particular, DiffRec leveraged continuous diffusion at the embedding level, gradually adding normal Gaussian noises – a type of *isotropic noise* (Tailor et al., 2022) – into the encoded user/item embeddings as noisy latent variables. Then, DiffRec refined the obtained noisy embeddings in a denoising manner. To understand the implications of the used noise, we conduct a preliminary analysis on two commonly used recommendation datasets, using 2D Singular Value Decomposition (SVD) to visualise the raw item features. Figure 4.2 (a) shows data instances in various colours representing each item as per the movie genre in the MovieLens-1M dataset, while Figure 4.2 (b) represents the item distribution according to the venue tags (examples of tags include "brunch" and "bar") in the Foursquare dataset. For example, the projected data from the Foursquare dataset exhibit strong anisotropic structures along only a few directions rather than spreading over the distribution space. This observation indicates that the recommendation data inherently features distinctive anisotropic and directional characteristics, which may not align with the assumptions underlying the use of isotropic Gaussian noise. Consequently, the integration of isotropic noise in existing diffusion-enhanced recommendation models may lead to less distinguishable user/item embeddings. These indistinct embeddings obscure the unique attributes of items, making it more challenging to preserve item heterogeneity. Motivated by the aforementioned observation – that the recommendation data inherently features distinctive anisotropic and directional characteristics – we argue that the use of *directional noise* is more suitable, as it aligns with the anisotropic structures typically observed in the recommendation data. The adoption of directional noise not only addresses the issue of distinguishable embeddings but also aligns with the underlying data patterns, thereby enhancing the model's ability to accurately capture the unique characteristics of each user/item.

Inspired by the remarkable success of diffusion models in different domains, such as image synthesis (Dhariwal and Nichol, 2021; Rombach et al., 2022), sequential recommendation (Li et al., 2023b; Liu et al., 2023; Wang et al., 2023e) and graph/node classification tasks (Gasteiger et al., 2019; Kong et al., 2023; Vignac et al., 2023), in this second part of this chapter, we propose an diffusion model tailored to leverage the inherent characteristics of recommendation data. As discussed in Section 3.1, conventional graph-based recommender systems are unable to denoise noisy implicit interactions. To date, diffusion techniques have not been applied within graph-based recommender systems to explicitly model user-item interactions from implicit feedback. However, the common practice of implementing diffusion in graph neural models is based on discrete diffusion, which necessitates the use of a transition matrix to update the binary states (true/false) of interactions at each diffusion step (Chen et al., 2023). This process updates the edges' states in the adjacency matrix according to the transition probabilities in the matrix, increasing the computational complexity of the model. Consequently, this discrete diffusion approach, while tailored to discrete data such as graph data, encounters a notable efficiency



(a) MovieLens-1M



(b) Foursquare

Figure 4.2: 2D visualisation of the data using SVD decomposition. (a) Visualisation of the item representations in the MovieLens-1M dataset, with different colours indicating the genres of movies. (b) Visualisation of all items' tags of venues in the Foursquare dataset.

limitation. In contrast to discrete diffusion, we leverage instead a continuous diffusion at the user/item node embedding level for effective and efficient graph neural recommendation. Considering the observed directional structure in the recommendation data (as shown in Figure 4.2), we aim to determine (1) the most effective diffusion methods (discrete vs. continuous) and (2) the most effective type of noise (normal vs. directional) in recommender systems. We introduce a new **Diff**usion **G**raph **T**ransformer (**DiffGT**) model, which aims for both an effective and efficient integration of a diffusion process tailored to recommender systems.

The remainder of Section 4.2 is organised as follows: We first present the complementary related work in Section 4.2.1, situating our approach within existing denoising recommendation methods discussed in Section 3.1. Next, we introduce the second proposed model architecture in this chapter, a directional diffusion-based graph transformer model for top-K recommendation, in Section 4.2.2. Section 4.2.3 details the experimental setup for evaluating our DiffGT model and discusses the results in relation to each research question. This section demonstrates a further step in advancing graph neural architectures: moving beyond the message-passing propagation level to an overall architectural level through the integration of a transformer, thereby providing additional empirical support for our thesis claim that more expressive graph architectures can improve recommendation performance.

4.2.1 Denoising in Recommendation

As introduced in Section 3.1, the presence of noise poses significant challenges to implicit interactions. The noise skewes the capture of the users' preferences and obscures their true behavioural patterns, complicating the accurate prediction of the users' interests (Zhang et al., 2023a). The challenge and the necessity of mitigating noise in implicit user-item interactions have led to the exploration of various denoising techniques for recommender systems. Traditional approaches (Li et al., 2023b; Walker et al., 2022; Wang et al., 2023e; Wu et al., 2016; Ye et al., 2023) have primarily focused on using advanced neural models. Auto-encoder-based approaches like CDAE (Wu et al., 2016) corrupted the implicit interactions of users with random noises and then reconstructed the original input with auto-encoders during training. Recent diffusion-enhanced recommendation models (Li et al., 2023b; Walker et al., 2022; Wang et al., 2023e) principally enabled the used auto-encoder architecture to effectively denoise implicit interactions at different noise levels and removed the noise progressively in a step-wise manner. For example, DiffRec (Wang et al., 2023e) applied the isotropic Gaussian noise on the user's interaction embeddings and used a shared Multilayer Perceptron (MLP) for multi-step prediction, while maintaining a step-by-step approach to predict the subsequent state of these user interaction embeddings. However, these denoising methods assume that the embeddings are uniformly distributed and incorporate isotropic Gaussian noise into the user/item embeddings (Ortiz-Jimenez et al., 2020). Instead of employing isotropic noise, our model uses anisotropic and directional noise in the diffusion process to allow a more effective and tailored interaction prediction from the noisy implicit interactions. We argue that the choice of anisotropic and directional noise better aligns with the inherent anisotropic structures observed in Figure 4.2. Furthermore, the existing diffusion recommender (i.e., DiffRec) essentially elaborates an unconditional generation paradigm. Hence, we integrate personalised information – such as the average embeddings derived from the items that a user has interacted with – to guide the diffusion process, enabling more effective user-item interaction predictions that better aligns with the users' interests.

4.2.2 A Directional Diffusion-based Graph Transformer

In this section, we first introduce how diffusion technique can be applied to recommender systems. Then, as discussed in Section 3.1, we present the challenges of incorporating the diffusion process into graph-based recommender systems. Next, we introduce the newly proposed graph neural architecture along with the injection of directional noise within the diffusion process into our proposed Diffusion Graph Transformer (DiffGT) model. We also present the optimisation strategy of our DiffGT model for the top-K recommendation task.

4.2.2.1 Preliminaries

A vanilla diffusion model involves two major components: the forward and reverse processes, which use latent variable modelling to enable the progressive generation of refined representations (Sohl-Dickstein et al., 2015).

- **Forward Process:** In the forward process of the diffusion model, a data point initially sampled from a real-world distribution, $\mathrm{x}_0 \sim q(\mathbf{x})$, is progressively corrupted through a Markov chain into a standard Gaussian noise, represented as $\mathbf{x}_T \sim \mathcal{N}(0, \mathrm{I})$. For each forward step $t \in [1, 2, \ldots, T]$, this corruption transforms the original representation $\mathbf{x}_0$ into a noisy representation $\mathbf{x}_t$ at each step t:

$$\begin{aligned} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) &= \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right), \\ &= \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathrm{I}) \end{aligned} \tag{4.12}$$

where $\mathcal{N}$ is a Gaussian distribution, and $\beta_t \in (0, 1)$ controls the level of the added noise at step t. This approach shows the flexibility of the direct sampling of $\mathbf{x}_t$ conditioned on the input $\mathbf{x}_{t-1}$ at an arbitrary diffusion step t from a random Gaussian noise ϵ.

- **Reverse Process:** The primary purpose of the diffusion models is to learn a denoising model capable of removing the added noise to the data and to gradually recover the initial distribution. Indeed, once a noisy embedding $\mathbf{x}_t$ is obtained, the reverse process aims to denoise this $\mathbf{x}_t$, with a trajectory towards the direction of $\mathbf{x}_0$ and to gradually recover the initial representation $\mathbf{x}_0$. The transition $(\mathbf{x}_t \rightarrow \mathbf{x}_{t-1} \rightarrow, \ldots, \rightarrow \mathbf{x}_0)$ is defined as follows:

$$\begin{aligned} p\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) &= \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_{\theta t}\left(\mathbf{x}_t, \mathbf{x}_0\right), \beta_t \mathbf{I}\right), \\ \mu_{\theta t}\left(\mathbf{x}_t, \mathbf{x}_0\right) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t, \\ \beta_t &= \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t, \end{aligned} \tag{4.13}$$

where μ_θ and σ_θ are the mean and variance of the user/item embeddings, respectively; $\alpha_t = 1 - \beta_t$ while $\bar{\alpha}_t = \prod_{t'=1}^{t} \alpha_t$. The learning of the mean μ_θ is based on a neural network f_θ parameterised by θ. As such, to effectively recover $\mathbf{x}_0$, the use of neural models, such as a Transformer (Vaswani et al., 2017) or a U-Net (Dhariwal and Nichol, 2021), is commonly used in practice (Ho et al., 2020).

- **Optimisation:** As described in Equation (4.13), the diffusion model parameterises both the mean μ_θ and variance σ_θ and learns to approximate the real data distribution during the reverse process. The model parameter θ is updated by minimising the difference between the Gaussian noise ϵ and the predicted noise, as follows:

$$\mathcal{L}_{\text{diffusion}} = \left\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right)\right\|^2, \tag{4.14}$$

where $\epsilon_\theta(\cdot)$ denotes the function that can be instantiated by a deep neural network to approximate ϵ. Note that this loss function is simplified from a variational lower-bound loss to prevent an unstable model training (Li et al., 2023b; Wang et al., 2023e).

Following the preliminaries of the diffusion process, its application in recommendation models presents two challenges, primarily due to the following constraints:

- ***Limitation 1:*** **Anisotropic and directional data structures:** As illustrated in Figure 4.2, the recommendation data often exhibits unique anisotropic and directional structures. The prevalent application of a normal Gaussian noise assumes the uniform distribution of data in the recommendation task. This assumption does not account for the unique and directional nature of the recommendation data, thus impeding the effective learning of the user/item representations.

- ***Limitation 2:*** **Unconditioned diffusion paradigm:** The current diffusion approaches in recommendation essentially perform an unconditioned generation paradigm (Wang et al., 2023e). This approach presents a limitation in effectively denoising the noisy user/item embeddings in recommender systems. Consequently, there is a pressing need to refine these diffusion approaches by conditioning the diffusion process.

4.2.2.2 Directional Gaussian Noise

As outlined in ***Limitation 1***, our preliminary analysis uncovered a pivotal factor contributing to the underwhelming performance of out-of-the-box diffusion models in the top-K recommendation task: the observed directional structure of the data. This further highlights the significance of tackling the challenges posed by anisotropic structures within diffusion models in the recommendation task. To tackle this challenge, we adapt the general graph approach from (Yang et al., 2023) to a recommender system scenario, where we inject directional noise into the forward diffusion process. The transformation from the isotropic Gaussian noise to the anisotropic Gaussian noise ($\epsilon \rightarrow \epsilon'$) ensures a better alignment with the inherent directional structures present in the recommendation data. As a result, the user/item node feature $x_{t,i} \in \mathbf{R}^d$ of node i at time step t is obtained as follows:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon', \tag{4.15}$$

$$\epsilon' = \operatorname{sgn}(x_0) \odot |\bar{\epsilon}|, \tag{4.16}$$

$$\bar{\epsilon} = \mu_\theta + \sigma_\theta \odot \epsilon, \tag{4.17}$$

where x_0 holds the raw embeddings of both users and items. The symbol $\odot$ denotes the pointwise product. In Equation (4.16), we align the noise ϵ' with the embeddings x_0 to ensure that they share the same coordinate signs. This guarantees that adding noise does not result in noisy features pointing in the opposite direction of x_0. By preserving the directionality of the original embeddings, the constraint in Equation (4.16) plays a crucial role in maintaining the inherent data structure during the forward diffusion process. This, in turn, ensures the application of directional noise to avoid a fast decline in the signal-to-noise ratio, preserving the essential information of the anisotropic structures (Yang et al., 2023). In Equation (4.17), we transform

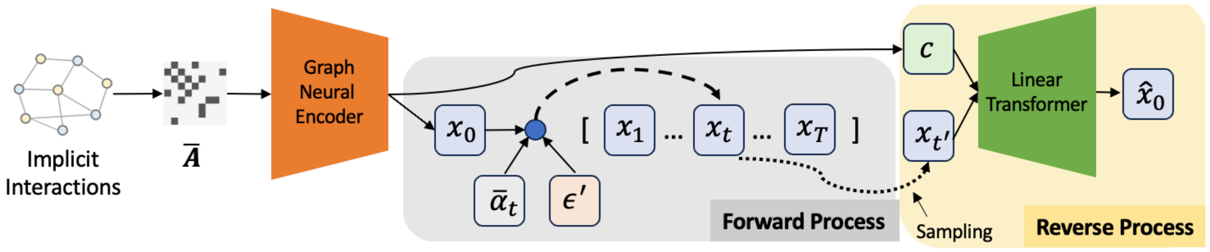


Figure 4.3: An illustration of our proposed DiffGT architecture.

the data-independent isotropic noise into an anisotropic noise. By doing this, each coordinate of the noise vector shares the same empirical mean and the same empirical standard deviation as the corresponding coordinate in the recommendation data. As such, the use of directional noise in our proposed DiffGT model allows to facilitate an effective user/item representation learning from the anisotropic structure of the latent space. In Section 4.2.3.3, we conduct experiments to ascertain the positive impact of directional noise.

4.2.2.3 Diffusion Graph Transformer (DiffGT)

Diffusion models have led to remarkable successes in various inverse problems in different domains, such as image synthesis (Dhariwal and Nichol, 2021; Rombach et al., 2022) and graph/node classification tasks (Gasteiger et al., 2019; Kong et al., 2023; Vignac et al., 2023). As discussed in Section 4.1.1, recent recommender systems have also adopted out-of-the-box diffusion models for denoising the user embeddings. These models, however, overlook the anisotropic nature of recommendation data. In this section, we leverage advanced user-item interaction modelling in graph-based recommender systems (He et al., 2020; Yu et al., 2023b) to investigate the impact of applying directional Gaussian noise in the top-K recommendation task. Integrating the diffusion process with graph-based recommender systems presents additional challenges:

- ***Limitation 3* – Non-informative bipartite interaction graph:** In contrast to diffusion models in the image synthesis task (Rombach et al., 2022), which start with raw images as ground truth, the bipartite user-item interaction graphs in the recommender systems are inherently noisy and less informative, thus not necessarily representing the 'ground truth';

- ***Limitation 4* – Insufficient model architecture:** The prevalent architectures for diffusion models in various domains (Dhariwal and Nichol, 2021; Kong et al., 2023; Wang et al., 2023e), notably variational auto-encoders and transformers, are not compatible with graph-based recommender systems due to the lack of learning parameters and limited expressiveness (Cai et al., 2023). Hence, this mismatch necessitates a graph-adapted architecture to fully leverage the potential of diffusion methods in enhancing the top-K recommendation task.

To address the above two limitations, we introduce the DiffGT model, which incorporates a novel graph transformer architecture to improve the diffusion for an enhanced top-K recommendation performance. Additionally, this model more effectively denoises the implicit interactions by leveraging side information, thereby developing a more accurate representation of a 'ground truth' interaction graph. As illustrated in Figure 4.3, our DiffGT architecture consists of two main components: a graph neural encoder and a linear transformer. Given an interaction graph, we obtain the adjacency matrix $\overline{\mathbf{A}}$, which is enriched by the used side information. For the forward process, once we obtain the original user/item embeddings x_0 encoded by the graph neural encoder, we gradually inject a scheduled directional noise ϵ', controlled by $\bar{\alpha}_t$, to obtain x_t. During the reverse process, we directly denoise $x_{t'}$ at a sampled t' step using the linear transformer to obtain the denoised embeddings $\hat{x_0}$.

Interaction Encoding: As shown in Figure 4.3, our DiffGT model uses a graph-based recommender system to obtain the initial user/item node representations x_0 via neighbourhood aggregation according to an adjacency matrix $\mathbf{A}$. To address ***Limitation 3***, we argue that incorporating additional data, such as the side information of both users and items, could better approximate the interaction graph and become closer to a 'ground truth' graph, thereby addressing the non-informative interaction graph. This information includes, for instance, the genre of the movies, thereby enriching the matrix with more contextually relevant data. Moreover, our DiffGT model employs a light-weight graph-based recommender system (He et al., 2020) to aggregate the augmented interaction graph as follows:

$$\mathbf{X}_G = \frac{1}{1+\ell_1}\left(\mathbf{X}_G^0 + \overline{\mathbf{A}}\mathbf{X}_G^0 + \ldots + \overline{\mathbf{A}}^{\ell_1}\mathbf{X}_G^0\right), \tag{4.18}$$

where $\mathbf{X}_G$ denotes the user and item representations after ℓ_1 layers of graph convolution operations, $\mathbf{X}_G^0$ represents the initial user and item embeddings at layer 0, and $\overline{\mathbf{A}}$ is the updated adjacency matrix, which is enriched by measuring the similarity between user-user and item-item pairs using the used side information. Specifically, we first construct two bipartite cosine-similarity matrices for both the user and item sides:

$$S_{u_1,u_2} = \frac{\mathbf{s}_{u_1}^\top \mathbf{s}_{u_2}}{\|\mathbf{s}_{u_1}\|\,\|\mathbf{s}_{u_2}\|} \qquad S_{i_1,i_2} = \frac{\mathbf{s}_{i_1}^\top \mathbf{s}_{i_2}}{\|\mathbf{s}_{i_1}\|\,\|\mathbf{s}_{i_2}\|}, \tag{4.19}$$

where $\mathbf{s}_u,\ \mathbf{s}_i\ \in\ \mathbb{R}^d$ are the d-dimensional side-feature vectors of user u and item i (e.g. demographics, genres, etc.), respectively. Next, we combine the user-user and item-item similarity martices into one diagonal similarity matrix S:

$$S = \begin{pmatrix} S_U & 0 \\ 0 & S_I \end{pmatrix} \tag{4.20}$$

We then fuse this matrix S with the original interaction adjacency $\mathbf{A}$ as follows:

$$\widetilde{A} \;=\; A \;+\; \gamma\, S, \quad \gamma > 0, \tag{4.21}$$

where γ is a hyper-parameter that balances the contributions of raw interactions and side information. Then, we apply a symmetric normalisation to $\widetilde{A}$ to obtain the updated adjacency matrix $\overline{\mathbf{A}}$:

$$\overline{\mathbf{A}} = D^{-\frac{1}{2}}\, \widetilde{A}\, D^{-\frac{1}{2}}, \tag{4.22}$$

Hence, introducing $\overline{\mathbf{A}}$ aims to enrich the interaction graph while also using the diffusion process to further reveal the true user-item interactions.

Conditional Denoising Transformer: As discussed in Section 4.2.2.1, existing diffusion approaches essentially elaborate an unconditioned denoising paradigm, as outlined as ***Limitation 2***. To address this limitation, we propose to refine the reverse process with a conditioned embedding particularly tailored to the recommender system, as shown in Figure 4.3. In our approach, we train a denoising network f_θ in the reverse diffusion process to denoises the corrupted x_t and to recover the original interaction vector x_0. Then, we let the denoising model f_θ directly predict $\mathbf{x}_0$ by sampling various $\mathbf{x}_t$:

$$p\left(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{c}\right) = \mathcal{N}\left(\mathbf{x}_t; \mu_{\theta t}\left(\mathbf{x}_t, \mathbf{x}_0\right), \beta_t \mathbf{I}\right), \tag{4.23}$$

where $\mathbf{c}$ is the conditioned embedding, typically derived from the users' interacted items in a recommendation setting. Then this conditioned embedding is concatenated with the noisy user/item embeddings to predict $\mathbf{x}_0$. Such personalised interaction information guides the reverse process in an effective manner, which further benefits the downstream recommendation task. Specifically, we emply a linear transformer (Wang et al., 2020c) to denoise the obtained $\mathbf{X}_G$ in the reverse process:

$$\mathbf{X}_T^{\ell_2+1} = \mathrm{LinearAttn}^{\ell_2}\left(\mathbf{X}_G^{\ell_2}, \mathbf{X}_\mathbf{C}\right), \tag{4.24}$$

where $\mathbf{X}_T$ denotes the resulting user/item representations, LinearAttn denotes a linear attention mechanism at the ℓ_2-th layer and $\mathbf{X}_\mathbf{C}$ is the average embedding of the user's interactions.

Efficient Graph Diffusion: To explore an effective and efficient diffusion strategy, aside from our diffusion approach, which operates at the embedding level (i.e., continuous diffusion), we conduct a comparative analysis with another variant of a diffusion approach that optimises the adjacency matrix of user-item interactions (i.e., discrete diffusion) (Haefeli et al., 2022; Vignac et al., 2023). Figure 4.4(a) and 4.4(b) illustrate the discrete and continuous diffusion methods,

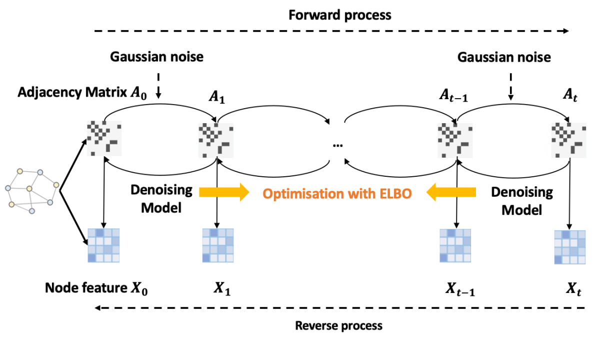


Fig. 4.4(a): An illustration of the discrete diffusion.

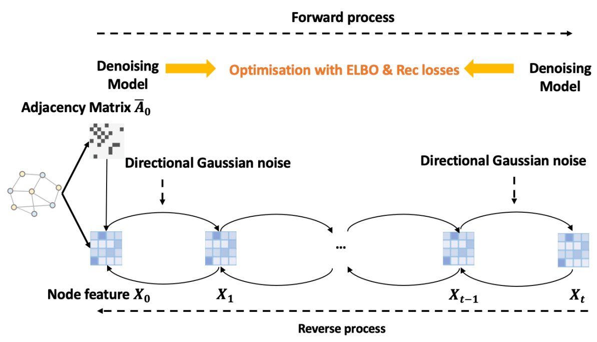

Forward process
Denoising Model
Optimisation with ELBO & Rec losses
Denoising Model
Adjacency Matrix $\bar{A}_0$
Directional Gaussian noise
Directional Gaussian noise
Node feature X_0
X_1
X_{t-1}
X_t
Reverse process

Fig. 4.4(b): An illustration of our proposed continuous diffusion.

respectively. The computation of the posterior distribution in the discrete diffusion is as follows:

$$q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) = \mathbf{x}_{t-1}\left(\boldsymbol{Q}^{t-1}\right)', \quad \text{where } \boldsymbol{Q}^t = \begin{bmatrix} 1-\alpha_t & \alpha_t \\ \alpha_t & 1-\alpha_t, \end{bmatrix} \tag{4.25}$$

where $\boldsymbol{Q}'$ is the transpose of $\boldsymbol{Q}$. α_t is the probability of not changing the edge states in the user-item interaction graph. Since the user-item graph is symmetric in recommender system, we focus on modelling the upper triangular part of the adjacency matrix. From Equation (4.25), the discrete graph diffusion involves a series of matrix multiplications throughout the diffusion steps. In contrast, the continuous graph diffusion, as defined in Equation (4.12), operates an embedding interpolation, which is deemed to be more efficient than a series of matrix multiplications given a fixed number of diffusion steps. In Section 4.2.3.5, we present a detailed efficiency analysis of the discrete and continuous diffusion processes. This efficiency analysis also includes an investigation of a sampled denoising method and the use of a linear transformer as an alternative to the traditional vanilla transformer, in order to enhance DiffGT's efficiency.

Optimisation: To ensure an effective denoising from implicit interactions, we optimise our DiffGT model with three loss functions, namely BPR loss $\mathcal{L}_{bpr}$ (c.f. Equation (2.4)) for the recommendation task, the diffusion loss $\mathcal{L}_{diff}$ for optimising the denoising outcome and the contrastive learning loss $\mathcal{L}_{cl}$ (c.f. Equation (2.11)):

$$\mathcal{L} = \mathcal{L}_{bpr} + \lambda_1 \mathcal{L}_{diff} + \lambda_2 \mathcal{L}_{cl}, \tag{4.26}$$

The diffusion loss $\mathcal{L}_{diff}$ is derived from Equation (4.14), which regulates the denoising of the user and item embeddings. We also aim to maximise the agreement between the original user/item embeddings and the denoised user/item embeddings to generate additional supervision signals as follows:

$$\mathcal{L}_{cl} = -\log \frac{\exp\left(\mathbf{e}^{\top}\mathbf{e}'/\boldsymbol{\tau}\right)}{\sum_{e\neq\mathbf{r}}^{n} \exp\left(\mathbf{e}^{\top}\mathbf{r}/\boldsymbol{\tau}\right)}, \tag{4.27}$$

where $\mathbf{e}$ is the user/item embedding obtained from a graph encoder, $\mathbf{e}'$ is the denoised user/item

Table 4.6: Statistics of the used datasets.

Dataset	Users	Items	Interactions	Density
Movielens-1M	6,040	3,704	1,000,209	4.47%
Foursquare	2,060	2,876	27,149	0.46%
Yelp2018	31,668	38,048	1,561,406	0.13%
Amazon Book	52,643	91,599	2,984,108	0.06%

embedding after $\mathbf{e}_u$ has been processed through the transformer, $\mathbf{r}$ is the embedding of a different user/ item node and τ is a hyper-parameter that adjusts the dynamic range of the resulting loss value. As such, additional gradients generated from contrastive learning facilitate effective denoisation by optimising our graph transformer model in the diffusion process.

4.2.3 Experiments

To demonstrate the effectiveness of our DiffGT [5] model, including its use of directional noise and the proposed graph transformer architecture, we conduct extensive experiments on four real-world public datasets to answer the following research questions:

- **RQ4.4**: Can our proposed DiffGT model, leveraging the diffusion process, outperform existing baselines in the top-K recommendation task?
- **RQ4.5**: What is the performance impact of the various key components of DiffGT, namely its directional noise, its graph transformer, its augmented adjacency matrix, its use of the conditioned embedding and its loss functions?
- **RQ4.6**: Can we efficiently incorporate diffusion when making recommendations?
- **RQ4.7**: How do different parameter settings affect our model performance?

We also examine the generalisation of applying the directional noise diffusion to other recommendation models, in addition to the graph neural recommenders:

- **RQ4.8**: Do our proposed directional diffusion and linear transformer approaches generalise to other types of recommendation models, such as knowledge graph and sequential recommenders?

4.2.3.1 Datasets and Experimental Settings

To answer the first four research questions in relation to the effectiveness and efficiency of DiffGT, we conduct an extensive evaluation on four real-world datasets, namely *Movielens-1M*[6],

[5] To support reproducibility, our code and scripts are publicly available at: https://github.com/zxy-ml84/DiffGT/.

[6] `https://grouplens.org/datasets/movielens/1m/`

Foursquare[7], *Yelp2018*[8] and *Amazon Book*[9], which have not been used in the previous section of this chapter. Table 4.6 presents the statistics of the four used datasets. Following the experimental setup of (He et al., 2020), we randomly split the aforementioned datasets into training, validation, and testing sets with a 7:1:2 ratio. Recall from the discussion in Section 4.2.2.3, that we aim to leverage the abundant side information within these four datasets to augment the user-item interaction graph for effective diffusion. To do so, we follow the methodology of (Meng et al., 2021) for enriching the adjacency matrix with side information. We compute the cosine similarity, which compares two vectors with an inner product, for each pair of user or item feature vectors derived from the side information. We then identify and select the top-n similar users/items accordingly. Next, we update the adjacency matrix by converting a 0 to a 1 for each similar user-user or item-item pair, resulting in an enriched outcome.

All experiments are conducted on a machine equipped with an RTX A6000 GPU for a fair comparison. For the hyper-parameters specific to the loss function of DiffGT, as introduced in Equation (4.11), we tune each of λ_1 and λ_2 within the ranges of $\{0, 0.1, 0.2, ..., 1.0\}$. We also tune τ, as described in Equation (2.4), within the ranges of $\{0, 0.1, 0.2, ..., 1.0, 2.0, ...10.0\}$. For the other hyper-parameters of DiffGT, which are optimised on the validation set, we tune the directional noise factor ϵ in Equation (4.17) within the ranges of $\{0, 0.1, 0.2, ..., 1.0, 2.0, ...10.0\}$, the diffusion step t within the range of $\{10, 20, ..., 100\}$, the factor of controlling the contributions of side information γ within the ranges of $\{0, 0.1, 0.2, ..., 1.0, 2.0, ...10.0\}$, the number of top-$n$ similar neighbours for a user or item within the range of $\{0, 5, ..., 20\}$ and the number of layers in the transformer layer varies over $\{1, 2, ..., 5\}$. For the used baselines, we employ a grid search method to tune all the hyper-parameters. We ensure that the ranges of parameters or the settings, such as the learning rate ($\{10^{-2}, 10^{-3}, 10^{-4}\}$), the batch size (2048), and the initialisation (Xavier) are consistent with the ones used in our DiffGT model, so as to enforce a fair comparison. In contrast to previous studies that rely on a single oracle testing set per dataset, we construct 10 different testing sets for each dataset using different random seeds. This multiple testing setups prevent any single test from favouring specific model characteristics, thus reducing the evaluation bias (Qian et al., 2021). Hence, the reported performance of each run represents the average of the 10 testing sets, further enforcing a fair comparison. Moreover, we adopt two widely used evaluation metrics, namely Recall@K and NDCG@K (c.f. Section 2.4), to evaluate the performance of top-K recommendation. We follow the setup of (He et al., 2020) and (Wang et al., 2023e), using a cutoff of k=20. We also apply early-stopping for both our DiffGT model and the used baselines, which terminates the training when the validation loss fails to decrease for 50 epochs.

[7] https://sites.google.com/site/yangdingqi/home/foursquare-dataset

[8] https://www.yelp.com/dataset [9] https://jmcauley.ucsd.edu/data/amazon/

4.2.3.2 Baselines

We evaluate the effectiveness of our proposed DiffGT model by comparing it with eleven existing state-of-the-art recommendation models. The baselines can be categorised into four groups:

- Graph-based models: **LightGCN, SGL** and **SimGCL** (as introduced in Section 3.1). We additionally introduce a graph transformer recommendation baseline, **GFormer** (Li et al., 2023a), which contrasts the user/item embeddings of a graph encoder and a separate transformer module by selectively dropping out the low-weight edges. Note that our DiffGT model exhibits a well-cascaded graph transformer architecture to gradually denoise the implicit interactions, which is different from the GFormer model;

- Conventional and denoising recommendation models: **MF** (Tang et al., 2016) decomposes the user-item interaction matrix into the product of two low-rank matrices, representing the latent factors of the users and items, respectively; **CDAE** (Wu et al., 2016) introduces random noises to the user-item interactions and uses an auto-encoder for denoising the implicit interactions; **MultiVAE** (Liang et al., 2018) uses Variational Auto-Encoders (VAEs) with multinomial likelihood to model implicit interactions of the target users;

- Diffusion-enhanced recommendation models: **DiffRec** Wang et al. (2023e) is a VAE-based diffusion approach that infers the users' preferences by modelling the interaction probabilities in a denoising manner; **GiffCF** Zhu et al. (2024) performs a diffusion technique on the interaction vectors by applying a smoothing filter over the item-item similarity graph in the forward process and denoising the smoothed interaction vectors with a Multi-layer Perceptron to generate refined user/item embeddings;

4.2.3.3 Overall Performance (RQ4.4)

Table 4.7 reports the results of comparing our DiffGT model with all the used baselines, which are described in Section 3.1. We evaluate our DiffGT model in comparison to four distinct recommendation approaches: conventional recommenders, diffusion-enhanced recommenders, side information-enriched recommenders and graph-based recommender systems.

Performance of DiffGT: From the observed results on four datasets, we conclude that DiffGT significantly outperforms the baselines on both used metrics, as confirmed by a paired t-test with the Holm-Bonferroni correction ($p < 0.05$). The improvements can be attributed to the integration of a graph transformer network that incorporates the directional noise within the forward process, and its denoising process, which is conditioned on personalised information from the user's historical interactions.

Effectiveness of the Transformer Architecture: We also evaluate the graph transformer architecture in DiffGT by comparing its performance to that of classical graph-based recommender

Table 4.7: Experimental results comparing our DiffGT model and the used baselines. The best performance of each model is highlighted in bold. * denotes a significant difference compared to the performance of the baselines using the Holm-Bonferroni corrected paired t-test with $p < 0.05$.

Dataset	Movielens-1M		Foursquare		Yelp2018		Amazon Book	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
MF	0.1638^*	0.2074^*	0.2430^*	0.3424^*	0.0409^*	0.0341^*	0.0631^*	0.0784^*
CDAE	0.2075^*	0.2346^*	0.2915^*	0.3877^*	0.0435^*	0.0363^*	0.0606^*	0.0729^*
MultiVAE	0.2740^*	0.3085^*	0.4151^*	0.5580^*	0.0557^*	0.0481^*	0.0734^*	0.0850^*
DiffRec	0.2756^*	0.3108^*	0.4414^*	0.6087^*	0.0663^*	0.0551^*	0.0826^*	0.0915^*
GiffCF	0.2743^*	0.3074^*	0.4406^*	0.6134^*	0.0685^*	0.0550^*	0.0823^*	0.0927^*
cVAE	0.2243^*	0.2665^*	0.3604^*	0.4747^*	0.0483^*	0.0417^*	0.0684^*	0.0802^*
HIRE	0.2709^*	0.3050^*	0.3634^*	0.4891^*	0.0613^*	0.0520^*	0.0795^*	0.0874^*
LightGCN	0.2697^*	0.3021^*	0.4161^*	0.5707^*	0.0589^*	0.0504^*	0.0742^*	0.0854^*
SGL	0.2723^*	0.3078^*	0.4333^*	0.5978^*	0.0646^*	0.0537^*	0.0788^*	0.0893^*
SimGCL	0.2788^*	0.3112^*	0.4353^*	0.6214^*	0.0701^*	0.0573^*	0.0844^*	0.0935^*
GFormer	0.2812^*	0.3142^*	0.4242^*	0.6051^*	0.0696^*	0.0559^*	0.0860^*	0.0939^*
DiffGT	**0.2903**	**0.3264**	**0.4589**	**0.6612**	**0.0715**	**0.0587**	**0.0912**	**0.0987**

systems (LightGCN, SGL, SimGCL) and another graph transformer method (GFormer). Table 4.7 shows that DiffGT and GFormer, both include a graph transformer architecture, outperform the classical graph-based recommender systems (LightGCN, SGL, SimGCL). These results show the effectiveness of the graph transformer network in modelling the users' preferences. Note that GFormer uses the transformer as a parallel encoder to construct a view for both the user and item embeddings in a contrastive loss. In contrast, our DiffGT model leverages a cascading architecture of a graph encoder and a transformer to effectively facilitate the denoisation in the diffusion process. The difference in performance between DiffGT and GFormer illustrates the superiority of our cascading architecture.

Effectiveness of the Diffusion-enhanced Recommenders: We now discuss the effectiveness of the diffusion-enhanced recommenders (DiffRec, GiffCF, DiffGT) by comparing their performance to the conventional models (MF, CDAE, MultiVAE). From Table 4.7, we observe that the diffusion-enhanced recommenders (DiffRec, GiffCF, DiffGT) significantly outperform the conventional recommenders (MF, CDAE, MultiVAE). The superiority of the diffusion-enhanced recommenders confirms the effectiveness of the diffusion process in effectively denoising the implicit interactions. Moreover, Table 4.7 shows that our DiffGT model significantly outperforms DiffRec and GiffCF, demonstrating the effectiveness of our novel diffusion process in injecting a directional noise via a graph transformer architecture instead of using a normal Gaussian noise.

Effectiveness of Leveraging More Noisy Interactions: By comparing DiffGT with the methods that also leverage side information, we observe from Table 4.7 that DiffGT significantly outperforms the side information-enhanced methods (cVAE, HIRE) by a large margin. This result indicates that DiffGT is indeed more effective at using the side information, such as movie

Table 4.8: Ablation study of the key components of DiffGT. * denotes a significant difference compared to the performance of the DiffGT variants using a paired t-test with $p < 0.05$.

Dataset	Movielens-1M		Foursquare		Yelp2018		Amazon Book	
Variants	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
-Direction	0.2533^*	0.2843^*	0.4016^*	0.5029^*	0.0541^*	0.0474^*	0.0756^*	0.0854^*
-CondDenoising	0.2822^*	0.3145^*	0.4322^*	0.5978^*	0.0703^*	0.0578^*	0.0892^*	0.0975^*
-Transformer	0.2535^*	0.2967^*	0.3631^*	0.5029^*	0.0616^*	0.0507^*	0.0836^*	0.0877^*
-GraphEncGAT	0.2216^*	0.2465^*	0.3826^*	0.5381^*	0.0584^*	0.0541^*	0.0792^*	0.0823^*
-GraphEncGraphSage	0.2452^*	0.2732^*	0.3728^*	0.5197^*	0.0535^*	0.0473^*	0.0780^*	0.0814^*
-Side	0.2858	0.3203	0.4352^*	0.6154^*	0.0704^*	0.0556^*	0.0893^*	0.0975
-CL	0.2701^*	0.3007^*	0.4144^*	0.5609^*	0.0603^*	0.0508^*	0.0866^*	0.0937^*
-DiffL	0.2736^*	0.3084^*	0.4263^*	0.5803^*	0.0616^*	0.0514^*	0.0893^*	0.0955^*
DiffGT	**0.2903**	**0.3264**	**0.4589**	**0.6612**	**0.0715**	**0.0587**	**0.0912**	**0.0987**

genres, than cVAE and HIRE.

In answer to RQ4.4, we conclude that our DiffGT model successfully leverages the directional noise in a graph transformer architecture for an effective diffusion process and outperforms all the existing strong baselines.

4.2.3.4 Ablation Analysis (RQ4.5)

In this section, we ablate each of the key components of DiffGT, namely the directional noise (discussed in Section 4.2.2.2) and the diffusion architecture components (Section 4.2.2.3), including the conditioned denoising, the enriched user-item interaction graph, the graph-adapted diffusion architecture and the loss functions.

Directional Noise: As highlighted under ***Limitation 1*** (see Section 4.2.2.1), recommendation data often exhibits unique anisotropic and directional structures. To enable a gradual addition of noise, and the recovery of the original embeddings in the denoising procedure, we use a directional Gaussian noise in the forward process of DiffGT. To demonstrate the effectiveness of this directional noise, we conduct a comparative analysis by replacing the directional Gaussian noise with a normal Gaussian noise. This substitution corresponds to the row indicated by "-Direction" in Table 4.8. From Table 4.8, we observe that DiffGT significantly outperforms the "-Direction" variant. This finding further confirms the effectiveness of incorporating directional noise, so as to align with the inherent directional structure of the recommendation data, thereby enhancing the recommendation performance.

To further explore the impact of incorporating a normal Gaussian noise or a directional Gaussian noise in our DiffGT model, we calculate the Signal-to-Noise Ratio (SNR) at each forward step of the diffusion process using Linear Discriminant Analysis (LDA) (Xanthopoulos et al., 2013). In particular, the SNR is defined as the ratio between the variance of the original signal and the variance of the added noise in the LDA-projected space. We calculate the SNR values by encoding the noisy user/item embeddings with a pre-trained graph-based recommender system (He et al., 2020) and project these embeddings into a linearly separable space. These

SNR values quantify the informativeness of the learned representations at different steps of the diffusion process. Figure 4.5 shows notable differences in SNR values between using the normal Gaussian noise and the directional Gaussian noise on the MovieLens-1M and Foursquare datasets while similar trends and conclusions are observed for the Yelp2018 and Amazon Book datasets. Figure 4.5 particularly shows that using a normal Gaussian noise in our DiffGT model consistently results in lower SNR values compared to the usage of a directional Gaussian noise. The notable low SNR values indicate that there is a high noise level within the embeddings, thus leading to a denoisation difficulty in the subsequent reverse process. In contrast, the SNR values of using a directional noise exhibit a slight decrease while still maintaining higher values compared to the normal Gaussian noise, which means the usage of a directional noise brings varied noise levels within the embeddings during the diffusion process. The variation in noise levels enables our DiffGT model to adaptively learn to denoise from different noise strengths of embeddings, thus enabling a more effective diffusion process. This observed difference of SNR values also aligns with Equation (4.16), which constrains the added noise to share the same coordinate sign with the embeddings x_0, thereby indicating its data-dependant characteristic. Consequently, our DiffGT model effectively enables the reverse process by looking into the noisy embeddings in different steps while considering the preserved information at different levels.

Conditioned Denoising: As discussed in Section 4.2.2.1 and ***Limitation 2***, existing diffusion methods in recommender systems essentially adopt an unconditioned generation paradigm. To address this limitation, our DiffGT model uses the average embedding of the user interactions as a conditioned embedding to guide the diffusion process. To examine the effectiveness of conditioning the diffusion over an unconditioned diffusion process, we remove the conditioned embedding $\mathbf{c}$ in Equation (4.23) during the reverse process of DiffGT and compare its recommendation performance with our original DiffGT model. We observe from Table 4.8 that the "-CondDenoising" variant, which omits the use of $\mathbf{c}$, underperforms DiffGT. This result indicates that the average embedding of the user interactions effectively guides the reverse process to enable a more effective diffusion process.

Enhanced User-item Interaction Graph: As discussed in Section 4.2.2.3 and ***Limitation 3***, we aim to incorporate additional data, such as the side information, to bring the interaction graph closer to a 'ground truth' graph. To examine the impact of incorporating side information on the effectiveness of diffusion, we conduct a comparative analysis of our DiffGT model's performance with and without an adjacency matrix enhanced by side information. In Table 4.8, the variant without side information is denoted as ”-Side”. The results show that our side information-enhanced DiffGT model significantly outperforms the "-Side" variant in five out of eight instances. Overall, the results show that the use of side information is a promising approach for further enhancing the denoising diffusion process in top-K recommendation.

Graph-adapted Diffusion Architecture: ***Limitation 4*** in Section 4.2.2.3 stipulates that the typical architectures used in the diffusion models are not compatible with the graph-based recom-

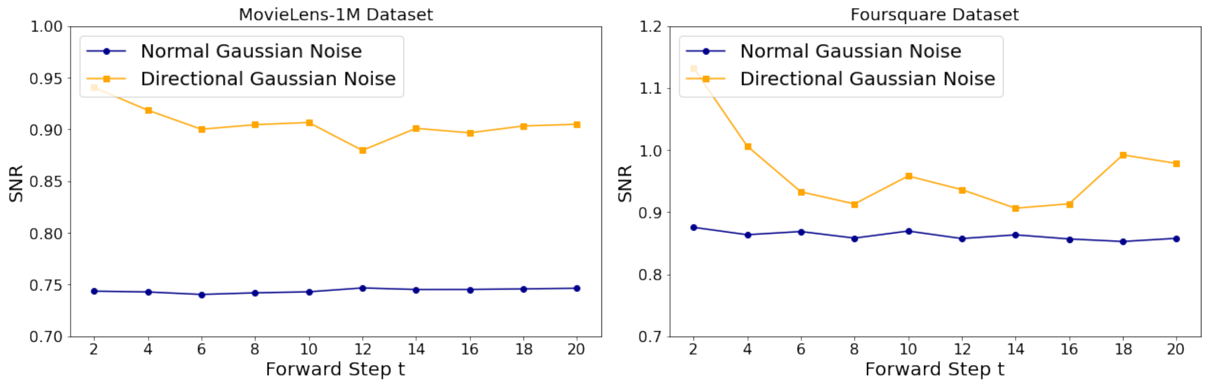


Figure 4.5: The Signal-to-Noise Ratio (SNR) curves along different diffusion steps on MovieLens-1M and Foursquare.

mender systems. To examine the effectiveness of our proposed graph transformer architecture in denoising the implicit interactions, we substitute the *transformer* with a weighted matrix to perform denoising since the *weighted matrix* is the commonly used method for denoising in a graph VAE architecture (Kong et al., 2023). Table 4.8 shows that our DiffGT model outperforms the "-Transformer" variant with the weighted matrix by a large margin. This superior performance demonstrates the suitability of a transformer network in enhancing the diffusion model for top-K recommender systems. We further ablate the usefulness of different graph encoders within the graph transformer architecture by replacing the used LightGCN with GAT (Veličković et al., 2018) and GraphSage (Hamilton et al., 2017), respectively. The latter two resulting variants are denoted as "-GraphEncGAT" and "-GraphEncGraphSage" in Table 4.8. From Table 4.8, we observe that both the "-GraphEncGAT" and "-GraphEncGraphSage" variants are less effective than DiffGT. This result indicates that the used LightGCN model within our graph transformer architecture is more effective than GAT and GraphSage in modelling user-item interactions as an encoder in the diffusion processes.

Loss Functions: As shown in Table 4.8, the inclusion of both the diffusion loss (the "-DiffL" variant) and the contrastive loss (the "-CL" variant) significantly enhances the performance of our DiffGT model. We attribute these improvements to the effect of the diffusion loss and the added-value of the contrastive loss in effectively denoising the user/item embeddings from the implicit interactions.

Overall, in response to RQ4.5, we conclude that our DiffGT model successfully leverages each of its components as well as its loss functions. Notably, the incorporation of the directional noise and the linear transformer significantly contribute to enhancing the model's performance in the top-K recommendation task.

4.2.3.5 Model Efficiency (RQ4.6)

In this section, we study the efficiency of our DiffGT model in terms of both time complexity and training time. In particular, we analyse the time complexity and training time of our DiffGT

Table 4.9: Comparative time complexity. * indicates a statistically significant difference according to a paired t-test with the Holm-Bonferroni correction with $p < 0.05$. $^+$ denotes a statistical equivalence based on the TOST test, also corrected using the Holm-Bonferroni method with $p < 0.05$.

Diffusion Process	Forward Process	Reverse Process	Recall@20	NDCG@20
DiffGT (discrete)	$\mathcal{O}(T \cdot 2E \cdot M^2 \cdot L_1 d)$	$\mathcal{O}(T \cdot N^2 \cdot L_2 d)$	0.2657*	0.2873*
DiffGT (continuous)	$\mathcal{O}(T \cdot 2E \cdot L_1 d)$	$\mathcal{O}(T \cdot N^2 \cdot L_2 d)$	0.2917	0.3269
DiffGT (continuous-linear)	$\mathcal{O}(T \cdot 2E \cdot L_1 d)$	$\mathcal{O}(T \cdot N \cdot L_2 d)$	0.2895$^+$	0.3253
DiffGT (continuous-sampling)	$\mathcal{O}(T \cdot 2E \cdot L_1 d)$	$\mathcal{O}(K \cdot N^2 \cdot L_2 d)$	0.2910$^+$	0.3270$^+$
DiffGT	$\mathcal{O}(T \cdot 2E \cdot L_1 d)$	$\mathcal{O}(N \cdot L_2 d)$	0.2903$^+$	0.3264$^+$

Table 4.10: Efficiency comparison on the Movielens-1M dataset in terms of training time. * indicates a statistically significant difference according to a paired t-test with the Holm-Bonferroni correction with $p < 0.05$. $^+$ denotes a statistical equivalence based on the TOST test, also corrected using the Holm-Bonferroni method with $p < 0.05$.

Model	Time/Epoch	Nbr of Epochs	Training Time	Recall@20	NDCG@20
LightGCN	76s	87	110m	0.2697*	0.3021*
GiffCF	186s	129	400m	0.2743*	0.3074*
DiffGT (discrete)	443s	112	827m	0.2657*	0.2873*
DiffGT (continuous)	280s	71	331m	0.2917	0.3269
DiffGT (continuous-linear)	167s	61	170m	0.2895$^+$	0.3253
DiffGT (continuous-sampling)	235s	69	270m	0.2910$^+$	0.3270$^+$
DiffGT	148s	64	158m	0.2903$^+$	0.3264$^+$

model, and compare it against its variants, which differ in terms of diffusion method (discrete vs. continuous), reverse process component (full vs. sampling denoising), and transformer architecture used (vanilla vs. linear transformer). To ensure a more comprehensive view, we also compare our DiffGT variants with two representative graph-based baselines, namely LightGCN and GiffCF. The results of the time complexity and effectiveness of the DiffGT variants are summarised in Table 4.9, while the training time and convergence speed of the DiffGT variants along with the two graph-based baselines (LightGCN, GiffCF) are summarised in Table 4.10. In the following, we briefly introduce the notation used in our complexity analysis. Let T denotes the number of steps in the diffusion process, E is the number of edges in the user-item bipartite graph, and N is the number of user and item nodes. For the forward process, L_1 is the number of layers in the graph encoder and M is the size of the transition matrix for a discrete diffusion. Since we leverage a transformer network for an effective denoising in the reverse process, L_2 and d represent the number of layers and the feature dimensionality in the used transformer, respectively. K denotes the number of the sampled steps, where $K << T$.

Discrete vs. continuous diffusion: As discussed in Section 4.2.2.3, there are two primary approaches in diffusion models: a discrete diffusion, which is tailored for discrete data such as adjacency matrices, and a continuous diffusion which is suitable for the embeddings. Besides applying a continuous diffusion at the embedding level, it is necessary to examine and compare the effectiveness and efficiency of a discrete diffusion to identify the most appropriate approach within the context of recommender systems. In this section, we compare the model efficiency and the recommendation performance between using a discrete diffusion (as detailed in Equation (4.25)) and a continuous diffusion (as detailed in Equation (4.12)). Table 4.9 presents the comparison of the time complexity and recommendation performance between the use of a discrete and a continuous diffusion in our DiffGT model, respectively. Comparing the discrete and continuous diffusion, a transition matrix is not required for continuous diffusion. As a result, the continuous diffusion is M^2 faster than the discrete diffusion in the forward process. In addition, Table 4.9 shows that the continuous diffusion outperforms the discrete diffusion in terms of effectiveness on both used metrics. For instance, the continuous diffusion variant achieves a higher recall value (0.2917) than the discrete diffusion variant (0.2657). Table 4.10 also shows that continuous diffusion requires fewer training epochs and results in a shorter total training time compared to discrete diffusion. This result indicates that continuous diffusion the better-suited approach for DiffGT, as it enhances recommendation performance while also increasing the model's efficiency.

Efficient denoising: Recall from the conventional reverse process in the out-of-box diffusion baseline (i.e., DiffRec) in Equation (4.13), that we aim to verify the efficiency and effectiveness of our proposed sampled denoising method as well as the used linear transformer in DiffGT. We examine if these components, outlined in Section 4.2.2.3, indeed enable a more efficient reverse process. Table 4.9 presents various DiffGT variants: DiffGT (continuous) incorporating a vanilla transformer across all diffusion steps, DiffGT (continuous-linear) employing a linear transformer, DiffGT (continuous-sampling) utilising a sampled denoising method with reduced steps, and a combination of both in the original DiffGT. We employ a two one-sided equivalence test (TOST) (Schuirmann, 1987) on the above variants to ascertain an effectiveness equivalence with the DiffGT (continuous) variant. Table 4.9 shows that the DiffGT (continuous-sampling) variant, compared to the DiffGT (continuous) variant, reduces the time complexity in the reverse process by $T - K$ times. Despite this reduction, the performance of DiffGT (continuous-sampling) remains on a par with the DiffGT (continuous) variant. Moreover, Table 4.10 further demonstrates that the DiffGT (continuous-sampling) variant requires less total training time than the DiffGT (continuous) variant while maintaining a comparable effectiveness. These findings suggest that the sampled denoising method maintains an effectiveness equivalent to the conventional approach while being more efficient.

We also compare the DiffGT (continuous-linear) with the DiffGT (continuous) variant in Table 4.9. We observe from Table 4.9 that the DiffGT (continuous-linear) variant reduces the quadratic complexity of the number of user/item nodes N^2 in the DiffGT (continuous) variant to a linear factor N during the reverse process, thereby improving computation efficiency of our DiffGT model. In addition, the training time results in Table 4.10 demonstrate that DiffGT (continuous-linear) achieves a notably shorter training time per epoch and a lower total

training time while maintaining a competitive effectiveness with the DiffGT (continuous) variant on the used Movielens-1M dataset. This result demonstrates that our DiffGT model successfully employs a linear transformer to enable an efficient and effective reverse process. Furthermore, we obtain the same conclusion when examining the combined use of the sampled denoising method and the linear transformer – i.e., in the full DiffGT model – as reflected in Table 4.9.
Training Time Efficiency: In addition to comparing DiffGT with all its variants in Table 4.10, we further evaluate the training efficiency of DiffGT in comparison with two representative graph-based recommenders: LightGCN, which serves as the base model of our graph transformer architecture, and GiffCF, a recent graph diffusion-enhanced recommender. From Table 4.10, we observe that our DiffGT model requires more training time per epoch than LightGCN while maintaining a comparable total training time (110 minutes vs. 158 minutes). This result indicates that our DiffGT model ensures a reasonable training efficiency through a faster convergence speed, although each training epoch is more computationally intensive. Furthermore, by comparing the time efficiency of DiffGT with GiffCF in Table 4.10, we observe that our DiffGT model is overall more efficient on terms of training time per epoch, number of epochs and total training time, while also achieving a higher effectiveness on terms of both the Recall@20 and NDCG@20 metrics. This result indicates that DiffGT ensures an effective trade-off between efficiency and effectiveness, thereby outperforming existing graph-based recommenders in both training efficiency and top-k recommendation quality.

Hence, in answer to RQ4.6, we evaluate both the internal design choices of DiffGT and the external comparison against graph-based baselines in terms of computation complexity and training efficiency. We conclude that using continuous diffusion leads to reduced time complexity in the forward process and improved recommendation performance compared to discrete diffusion . In addition, our proposed sampled denoising method and the use of a linear transformer improve the reverse process efficiency by reducing both time per epoch and total training time. Our DiffGT model also demonstrates improved training efficiency and effectiveness compared to existing graph-based recommendation models .

4.2.3.6 Hyper-parameter Study (RQ4.7)

We now study the sensitivity of our DiffGT model to the hyper-parameters. We focus on Recall@20 for reporting the recommendation performance, since we observe the same trends and conclusions with NDCG@20 across the used datasets. For brevity, these additional experiments present the effects of the hyper-parameters for the Movielens-1M and Yelp datasets, since the results on the Foursquare and Amazon Book datasets lead to similar conclusions. We primarily analyse four important parameters in our DiffGT model, namely: (1) the scaling factor λ_1, which controls the influence of diffusion loss during the diffusion process; (2) the contrastive factor λ_2, regulating the strength of the contrastive loss during training; (3) the directional noise factor ϵ, which represents the strength of the applied directional noise in the forward process;

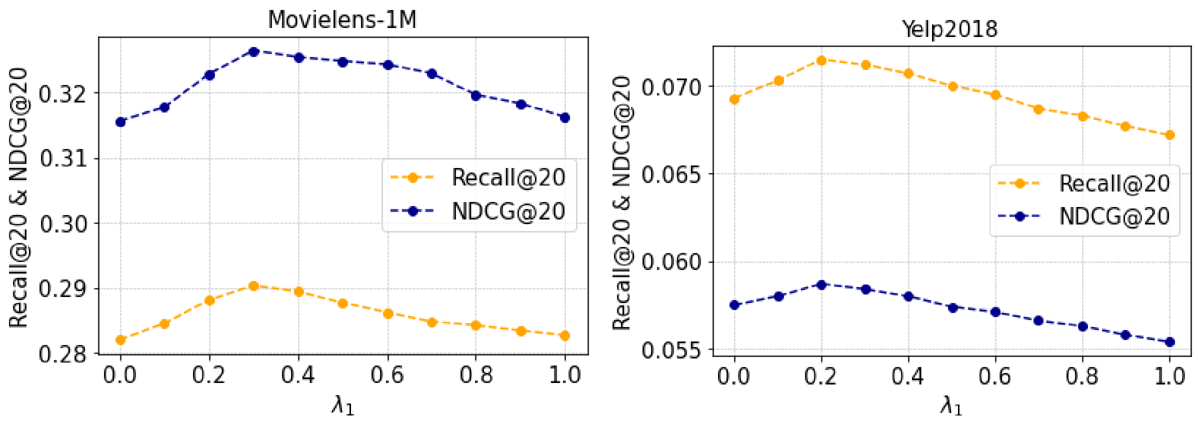


Figure 4.6: Performance of our DiffGT model with respect to different values of λ_1 on the Movielens-1M and Yelp datasets.

(4) the diffusion step t. Figure 4.6 and Figure 4.7 illustrate the effects of the hyper-parameters λ_1, λ_2, ϵ, and t on our DiffGT model's performance.

Impact of the scaling factor λ_1: The scaling factor λ_1 indicates the importance of the diffusion loss, which quantifies the difference between the applied and predicted Gaussian noises in DiffGT. As mentioned in Section 4.1.2, we vary λ_1 within the range {0.0, 0.1, ..., 0.9, 1.0} with a step size of 0.1. Figure 4.6 shows that DiffGT achieves its peak performance at $\lambda_1 = 0.3$ on the Movielens-1M dataset and $\lambda_1 = 0.2$ on the Yelp datasets, respectively. This indicates that lower values of λ_1 are generally more effective, providing stable and best-performing performance across different datasets by enabling more control over the diffusion process.

Impact of the contrastive factor λ_2: The contrastive factor λ_2 indicates the importance of the contrastive loss, balancing the contribution of this contrastive loss within the joint training loss in Equation (4.11). Similar to λ_1, we vary λ_2 within the range {0.0, 0.1, ..., 0.9, 1.0} with a step size of 0.1. From Figure 4.7, we observe that the best performance of our DiffGT model occurs at λ_2 = 0.2 on both used datasets, with a marked performance degradation at higher λ_2 values. A high λ_2 value indicates that DiffGT emphasises the similarity between original user/item embeddings and denoised user/item embeddings as measured by the contrastive loss, over the BPR pairwise ranking loss in the recommendation task. These results highlight the importance of carefully choosing λ_2 to tailor the loss function, thereby improving the model's performance.

Impact of the directional noise factor ϵ: The directional noise factor ϵ determines the strength of the added noise during the forward process. As introduced in Section 4, we vary ϵ within the ranges of $\{0, 0.1, 0.2, ..., 1.0, 2.0, ...10.0\}$. Figure 4.8 shows that an ϵ value of 0.1 yields the best performance for DiffGT on both datasets. This result indicates that a minimal level of added noise enhances the model's ability to accurately interpret both isotropic and directional structures within the user/item embeddings, rather than disrupting these embeddings with higher noise levels.

Impact of the diffusion step t: The diffusion step t determines the number of steps used to

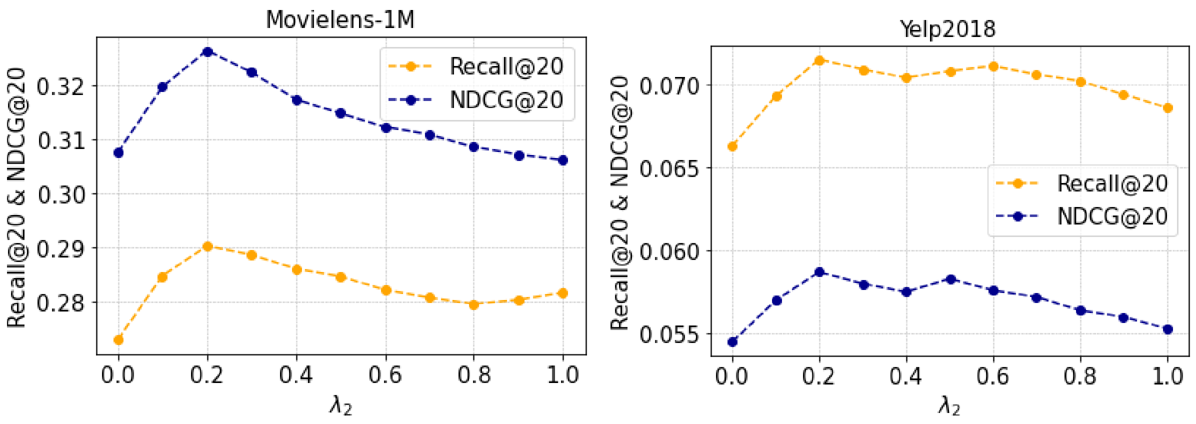


Figure 4.7: Performance of our DiffGT model with respect to different values of λ_2 on the Movielens-1M and Yelp datasets.

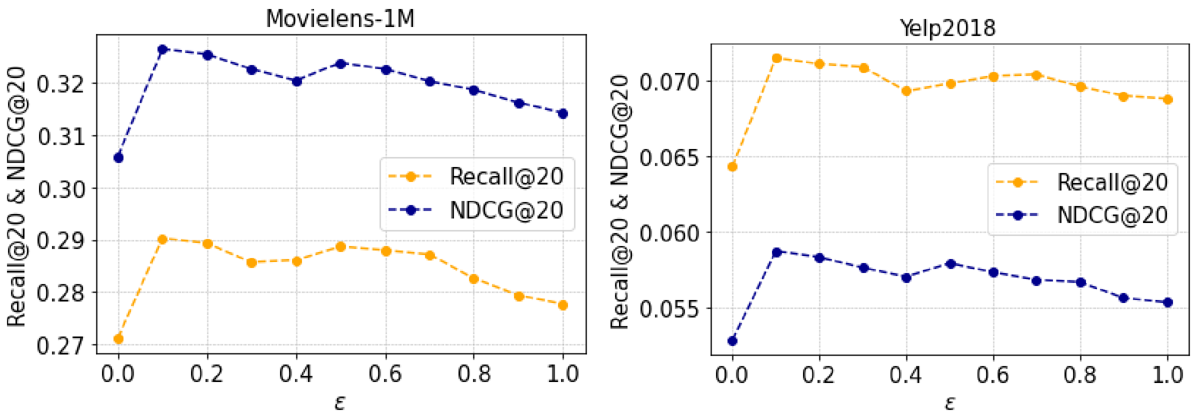
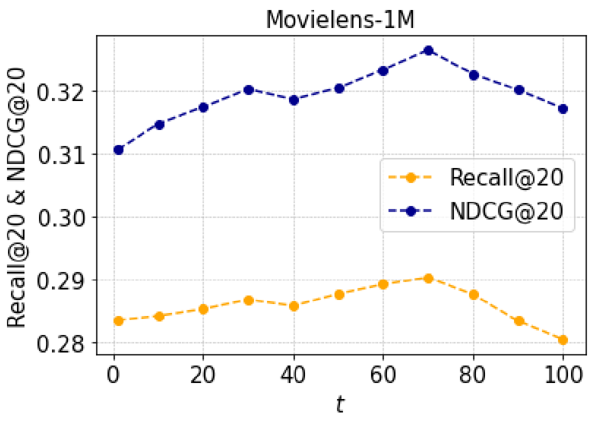


Figure 4.8: Performance of our DiffGT model with respect to different values of ϵ on the Movielens-1M and Yelp datasets.

transform the original user/item representation into a noisy representation, as discussed in Section 3. From Figure 4.9, we observe that DiffGT's performance initially increases with an improved t on both used datasets. This improvement shows that more diffusion steps enhance the model's ability to capture effective representations from both isotropic and directional data structures, thereby improving recommendation accuracy. However, excessively increasing the diffusion steps harms the model's performance, highlighting the need for a strategic selection of t to optimise DiffGT's effectiveness.

In response to RQ4.7, our hyper-parameter study shows that the performance of our DiffGT model remains relatively stable across the variations in these hyper-parameters λ_1, λ_2, ϵ, and t. This suggests that, while varying these parameters can fine-tune the performance, our DiffGT model maintains a consistent level of effectiveness across a range of settings on the used datasets.



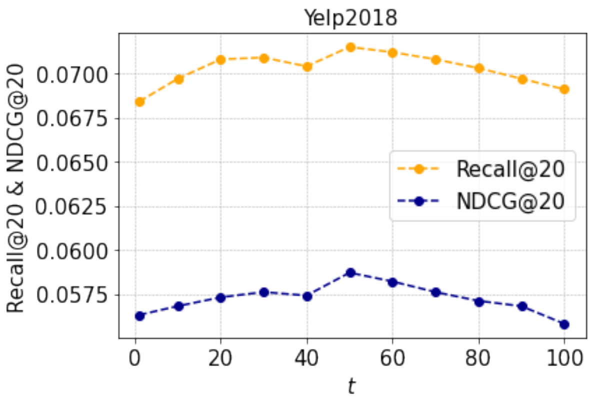


Figure 4.9: Performance of our DiffGT model with respect to different values of t on the Movielens-1M and Yelp datasets.

4.2.3.7 Generalisation (RQ4.8)

In this section, we extend the use of the directional diffusion and a linear transformer to other recommendation models so as to assess the generalisation of our proposed innovations beyond graph-based recommender systems. First, we apply the techniques to knowledge graph (KG) recommenders on the Amazon-Book (McAuley et al., 2015) and MIND (Wu et al., 2020a) datasets. Then, we investigate their effectiveness using sequential recommenders on the Beauty and Steam (Kang and McAuley, 2018; McAuley et al., 2015) datasets. Specifically, we integrate the diffusion process with a directional noise into the obtained user/item embeddings once they have been encoded by a knowledge graph (KG) or a sequential recommender. Then, we denoise these noisy embeddings using the linear transformer. Figure 4.10 shows the recommendation performance of several recent KG and sequential recommenders, comparing their original "Baseline" performance with that of their enhanced "Diff+Baseline" versions (e.g., CKE vs. DiffCKE), which use our proposed directional noise and linear transformer techniques. From Figure 4.10, we observe that the KG recommenders (CKE (Zhang et al., 2016), KGAT (Wang et al., 2019a), KGIN (Wang et al., 2021d), KGCL (Yang et al., 2022a), MCCLK (Zou et al., 2022)) augmented with our diffusion methodology significantly outperform their original variants in 90% tested cases (9 out of 10 instances) on the Amazon-Book and MIND datasets. Such notable improvements consolidate the effectiveness of our proposed directional diffusion and linear transformer techniques, since the encoding methods (i.e., graph convolution) used in the KG recommenders are similar to those in the graph-based recommender systems. These results highlight the strong generalisation of our injected directional noise and the used linear tranformer to KG-enriched recommenders. Next, we move to the sequential recommenders that aim to predict the next items that the users will interact with. Figure 4.10 shows that the sequential recommenders (GRU4Rec (Hidasi et al., 2015), SASRec (Kang and McAuley, 2018), BERT4Rec (Sun et al., 2019), S3Rec (Zhou et al., 2020b)), augmented with our new diffusion approach, which includes both directional noise and a linear transformer, show a superior perfor-

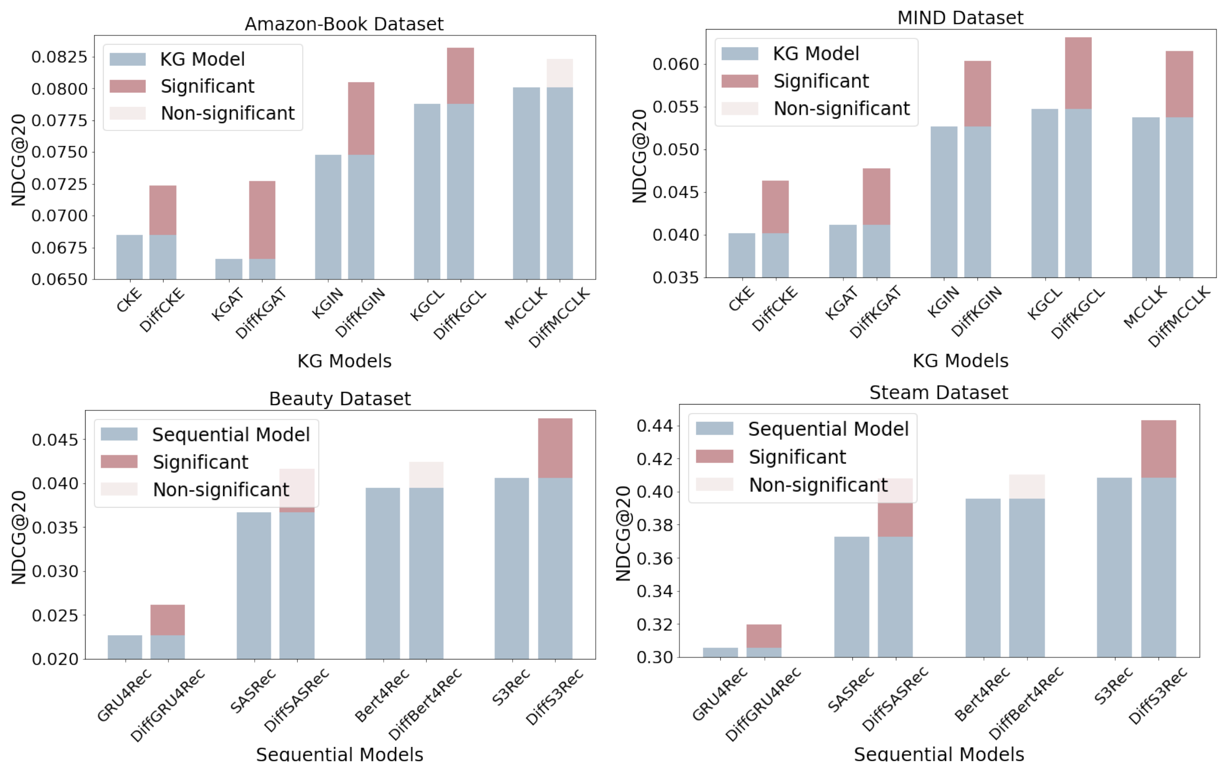


Figure 4.10: Recommendation performance of different KG and sequential recommendation models, comparing the baseline ('Baseline') and the models enhanced with our directional noise and linear transformer ('Diff+Baseline'). A red colour on the bar indicates a significant difference between the tested models and their corresponding directional diffusion-enhanced approaches according to a paired t-test with $p < 0.05$.

mance for 4 out of 5 models (except Bert4Rec) on both the Beauty and Steam datasets, compared to their original counter-parts. The absence of a significant improvement in Bert4Rec could be attributed to Bert4Rec's bi-directional mechanism, which is inherently more sensitive to noise due to its reliance on the full contextual understanding of the entire sequence. In this case, the injected directional noise might disrupt this bi-directional balance, resulting in a no notable performance improvement. In contrast, SASRec, which uses a unidirectional self-attention, appears to be more robust to such a noise as it does not rely on the entire sequence of interactions.

In response to RQ4.8, we conclude that our proposed directional noise and the used linear transformer effectively generalise to both KG-enriched and sequential recommendation models.

4.2.4 Discussion

This section demonstrates a further step in advancing graph neural architectures: moving beyond message-passing enhancements to a higher architectural level through the integration of a graph transformer, thereby providing additional empirical support for our thesis that more expressive graph architectures can improve recommendation performance. In doing so, we introduce a graph transformer architecture that complements the earlier message-passing enhancement in another architectural dimension, jointly contributing toward our thesis goal of developing more expressive graph neural architectures for top-K recommendation.

4.3 Conclusions

In this chapter, we investigated two complementary architectural directions toward strengthening graph neural architectures for top-K recommendation to align with our thesis statement. We first examined the effectiveness of incorporating Laplacian Positional Encoding (LapPE) and Random Walk Positional Encoding (RWPE) in an SSL paradigm to enhance the expressiveness of graph-based recommender systems at the message-passing level. Our results provide the first empirical validation of the architectural component of our thesis statement: the graph neural architecture can be strengthened by addressing key limitations of conventional graph-based recommender systems, particularly the over-smoothing problem. Our PGCL model can significantly outperformance all baselines (according to the paired t-test with the Holm-Bonferroni correction for $p < 0.01$), as demonstrated by the extensive experiments conducted on three public datasets (see Table 4.1). An ablation study in Section 4.1.3.4 showed that the Laplacian eigenvector is more beneficial than the random walk operator. Moreover, we showed that LapPE significantly enhances the expressive power of graph neural architectures in Section 4.1.3.5. By mitigating the over-smoothing problem on user/item embeddings, LapPE allows for the successful stacking of deeper layers, resulting in improved recommendation performance compared to conventional GNNs, as shown in Table 4.5. However, improvements at the message-passing level alone do not fully address the architectural limitations of graph-based recommender systems. Notably, current models still lack mechanisms to selectively filter or prioritise neighbour information under noisy implicit interactions. This gap necessitates a higher-level architectural enhancement beyond the improvement at the message-passing level.

Next, the second part of this chapter introduced a new graph neural architecture that integrates a transformer component with the graph encoder, along with an SSL-enhanced directional diffusion technique, to explicitly denoise implicit interactions. In particular, we proposed the Diffusion Graph Transformer (DiffGT), a novel diffusion model using a forward phase to introduce controlled noise and a reverse phase to iteratively refine (denoise) the user representations for top-K recommendation. This architectural improvement – from enhancing message passing to introducing a graph transformer – represents two approaches toward our thesis goal of developing more expressive graph neural architectures for top-K recommendation. In Table 4.7, our extensive experiments on four datasets (see Table 4.6) showed that DiffGT significantly outperformed eleven strong existing baselines. We also conducted an ablation study (see Table 4.8) and confirmed the positive impact of each component within our DiffGT model, namely directional noise, conditioned denoising, side information-enhanced interaction graph, and graph transformer architecture. Finally, in Figure 4.10, we showed that the effectiveness of our directional noise and the used linear transformer effectively generalised to other recommenders in Section 4.2.3.7, such as knowledge graph-enriched and sequential recommenders.

Despite proposing more expressive graph neural architectures, as discussed in Section 3.2, existing graph-based recommender systems still lack dedicated methods for effectively integrating multi-modal features to learn richer user and item representations. Next, in Chapter 5, we argue that mining rich signals and enabling alignments from multiple modalities in the SSL paradigm can result in improved top-K recommendation performance. Therefore, we aim to address the insufficient modality encoding issue prevalent in graph-based recommender systems by investigating more effective modality fusion methods, such as modality-specific graph augmentations and deep modality alignment within graph-based recommender systems.

Chapter 5

Self-supervised Modality Fusion

In Chapter 4, we introduced new graph neural architectures at both the message-passing level and the higher architectural level for graph-based recommender systems in a Self-supervised Learning (SSL) manner. These contributions provided empirical validation for the thesis statement in Section 1.2, in particular demonstrating that enhancing the expressiveness of graph neural architectures can improve the performance of graph-based recommender systems. However, as argued in Section 3.2, graph-based recommender systems still lack dedicated methods for effectively fusing multi-modal features, such as item images and descriptions, thereby limiting their ability to learn richer user and item representations, especially in a multi-modal setting. We refer to this problem as the insufficient modality encoding issue, which impedes effective modality fusion in graph-based recommender systems. As discussed in Section 2.3.2, SSL provides a promising direction for addressing this limitation by generating auxiliary supervision signals to support more effective multi-modal fusion. In particular, contrasting augmented views across modalities and enforcing cross-modal alignment within the SSL paradigm offer promising directions for enhancing user/item representations. Unlike Chapter 4, which focused on top-K generation recommendation, this chapter focuses on top-K recommendation in a multi-modal setting. Therefore, we propose new modality fusion methods for graph-based recommender systems by leveraging SSL-based augmentation and alignment strategies across modalities, thereby improving top-K multi-modal recommendation performance.

As mentioned in Section 3.2, existing multi-modal graph-based recommender systems, such as MMGCN, treat each modality with equal importance in the modality fusion process by simply concatenating user/item embeddings from each modality-specific interaction graph. However, we argue that these approaches are not sufficient to fuse multi-modal features, since the used methods cannot fully disentangle the users' interests in different modalities. On the other hand, as discussed in Section 3.2, the existing approaches (He and McAuley, 2016b; Wei et al., 2019; Zhang Jinghao et al., 2021) primarily depend on the features extracted individually from different modalities through pre-trained modality-specific encoders, and exhibit only shallow alignments between different modalities, thereby limiting the fusion process to capture the underlying relationships between the modalities. To tackle these problems, in this chapter, we propose two multi-modal graph-based recommender systems with new modality fusion methods that address the limitations identified in Section 3.2. The first model explores the effectiveness of two novel graph augmentations (i.e., *modality edge dropout* and *modality masking*) along with *challenging negative samples* within an SSL framework to obtain a more effective modality fusion. The second model aims to further enhance the fusion quality by introducing deeper modality alignment, using *Large Multi-modal (LMM) encoders* (e.g., CLIP, VLMo, BEiT-3) for an improved top-K multi-modal recommendation performance. The remainder of this chapter is structured as follows:

- Section 5.1 presents the first proposed model, MMGCL, which leverages modality edge dropout, modality masking augmentations and challenging negative samples in an SSL loss (as detailed in Section 2.3) to enable the contribution of each modality during graph representation learning;
- Section 5.2 describes the second approach, which extends MMGCL and other multi-modal recommenders by exploring modality fusion at the feature extraction stage through deeper modality alignment using LMM encoders. This includes exploring strategies such as using pre-trained and fine-tuned LMM encoders, as well as the evaluation of an end-to-end training of these encoders, so as to enhance modality fusion for an improved top-K multi-modal recommendation performance across all graph-based recommender systems;
- Section 5.3 summarises the key findings and insights presented in this chapter and discusses their contributions to the thesis statement.

5.1 Modality Fusion with Contrastive Learning

As introduced in Section 3.2, existing graph-based models (Sun et al., 2020; Wu et al., 2019a) incorporated multi-modal information to generate node representation in a uni-graph. Alternatively, Wei et al. (2019) leveraged the learned user preferences from each modality graph respectively, but without disentangling the users' interests in different modalities. Consequently, the aforementioned approaches (Sun et al., 2020; Wei et al., 2019; Wu et al., 2019a), which either unified or homogenised multi-modal channels, treated each modality with equal importance. This could potentially lead to suboptimal modality fusion for example by overemphasising a particularly modality in the learned representations.

As described in Section 2.3.3, SSL can enable graph-based recommender systems to construct additional supervised signals from correlation within raw data. In this section, we investigate the possibility of leveraging SSL to explore correlations among modalities in order to alleviate the equal importance problem in multi-modal recommendations. On the other hand, contrastive learning has recently become a dominant component in SSL. A typical way (Wu

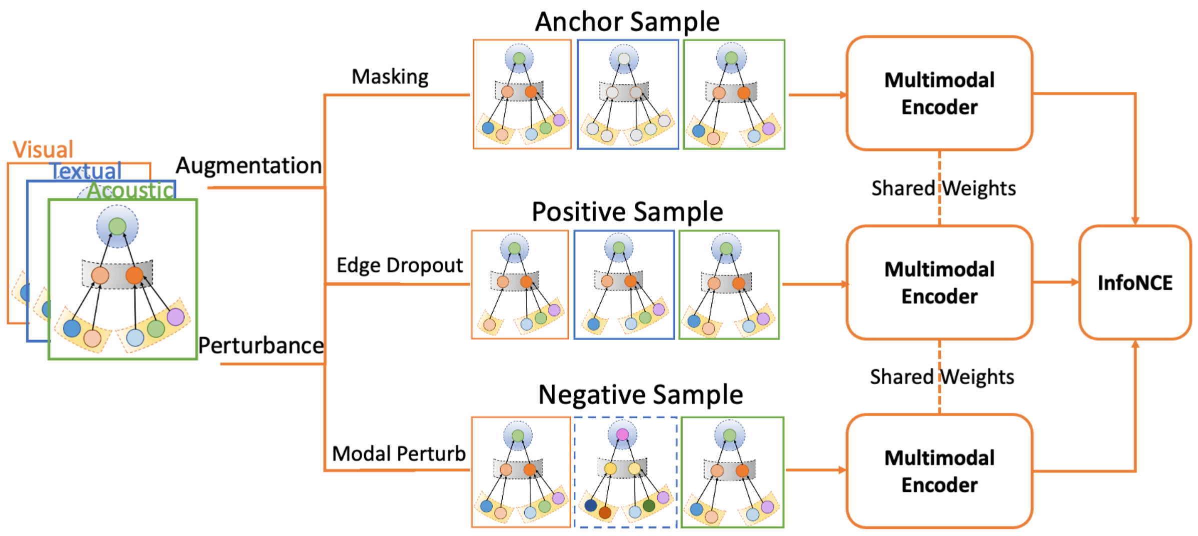


Figure 5.1: Overview of our proposed MMGCL model

et al., 2021a) to apply SSL-based contrastive learning to recommendation on graphs is by generating multiple views by perturbing the user-item bipartite graphs. Then the views are contrasted by maximising the agreement between different views of the same node (i.e. positive pairs) and increasing the disagreement compared to other nodes (i.e. negative pairs). However, existing SSL approaches based on graphs, such as SGL (c.f. Section 3.1), cannot be directly applied to multi-modal recommendations because they fail to generate informative views for nodes from multiple modalities. In addition, they cannot provide an estimation of the users' interests in different modalities as auxiliary signals during training.

To address the equal importance problem, we propose a novel learning method called Multi-Modal Graph Contrastive Learning (MMGCL) to explicitly enhance modality fusion in multi-modal recommendations. For this purpose, MMGCL leverages positive pairs <anchor sample, positive sample> of nodes, where the anchor sample refers to the original node representation and the positive sample is its augmented representation. To generate these positive samples, MMGCL introduces two graph augmentation techniques: *modality edge dropout* and *modality masking*. The first removes a small portion of edges from graphs of different modalities and the second technique selectively masks one particular modality of user/item features to obtain more informative views that can be contrasted in SSL. Furthermore, MMGCL generates challenging negative samples by perturbing one particular modality of the positive sample, enabling the model to learn correlations between modalities. The joint contribution of the above techniques encourages the encoder to capture inter-modality correlations, ensuring the effective contribution of each modality in modality fusion.

5.1.1 Multi-modal Graph Contrastive Learning

In this section, we describe our proposed MMGCL model, the architecture of which is illustrated in Figure 5.1. Next, we introduce the proposed modality masking and modality edge dropout as multi-view graph augmentations to be used in the SSL loss. We then introduce the proposed challenging negative samples and apply them to a multi-task learning loss for an effective modality fusion in top-K recommendation.

5.1.1.1 Task Definition

In this section, we extend the notations introduced in Chapter 4 from the top-K recommendation setting to a top-K multi-modal recommendation setting. Specifically, Chapter 4 formulated the top-K recommendation task as learning user–item representations from a bipartite interaction graph $\mathcal{G}$, where users $\mathcal{U}$ and items $\mathcal{I}$ are represented as nodes and $\mathcal{E}$ represents observed interactions as the edges. Following the setting of MMGCN (as introduced in Section 3.2), we construct modality-specific bipartite graphs for each modality, where nodes represent users and items, and edges indicate observed interactions. Formally, we denote $m \in \mathcal{M} = \{v, a, t\}$ as the modality indicator, where v, a, and t correspond to the visual, acoustic, and textual modalities, respectively. Given the modality-specific interaction graphs $\mathcal{G}v$, $\mathcal{G}a$, and $\mathcal{G}_t$, our goal is to estimate user preferences through a multi-modal graph encoder f that integrates information from all modality-specific graphs and recommends the top-K items (e.g., micro-videos) for each target user u.

5.1.1.2 Multi-view Graph Augmentation

Inspired by (Hassani and Khasahmadi, 2020; Tian et al., 2020), who applied multi-view representation learning for contrastive learning, we devise two augmentation operators on the multi-modal graphs $\mathcal{G}_v$, $\mathcal{G}_a$ and $\mathcal{G}_t$, namely *modality edge dropout* and *modality masking*, to create a multi-view representation $V(\mathcal{G})$ of each node. It is worth noting that our approach can be extended to arbitrary modalities, though it is constrained by the number of modalities available in the dataset, as detailed in Section 5.1.2.1. Figure 5.1 provides an overview of MMGCL.

Modality Masking: As illustrated in Figure 5.1, we apply a masking pattern on a particular modality of user/item features as follows:

$$V_1(\mathcal{G}) = \begin{cases} (\mathcal{V}_v, \mathcal{E}_v) \parallel (\mathcal{V}_a, \mathcal{E}_a) \parallel (M_1 \odot \mathcal{V}_t, \mathcal{E}_t) & \text{with } p_t \\ (\mathcal{V}_v, \mathcal{E}_v) \parallel (M_1 \odot \mathcal{V}_a, \mathcal{E}_a) \parallel (\mathcal{V}_t, \mathcal{E}_t) & \text{with } p_a \\ (M_1 \odot \mathcal{V}_v, \mathcal{E}_v) \parallel (\mathcal{V}_a, \mathcal{E}_a) \parallel (\mathcal{V}_t, \mathcal{E}_t) & \text{with } p_v \end{cases} \tag{5.1}$$

where $\parallel$ represents the concatenation operator, p_v, p_a, p_t are the individual probabilities to control which of the three modalities are masked and $p_v + p_a + p_t = 1$. M_1 is the masking vector on the node set $\mathcal{V}$. We implement this masking operator by replacing a particular modality of user/item features with a randomly initialised embedding in the input layer. The masking step can be interpreted as a special case of a 100% dropout rate. As such, this augmentation is expected to increase the contribution of each modality with the consistent absence of the masked modalities during training.

Modality Edge Dropout: This randomly removes edges in each modality graph with a dropout ratio ρ. The resulting view is represented as:

$$V_2(\mathcal{G}) = (\mathcal{V}_v, M_2 \odot \mathcal{E}_v) \parallel (\mathcal{V}_a, M_2 \odot \mathcal{E}_a) \parallel (\mathcal{V}_t, M_2 \odot \mathcal{E}_t) \tag{5.2}$$

where $\mathcal{V}$ is the node set and M_2 is the masking vector on edge set $\mathcal{E}$. This derived view is created by a set of sub-graphs from the original multi-modal graphs and can still preserve the users' main intentions on different modalities. As such, this augmentation is expected to capture the useful patterns of a node on each uni-modal graph and further enriches the representations by concatenation.

5.1.1.3 Challenging Negative Samples

Hard negative mining has been effectively applied in multi-modal fusion scenarios where particular modalities tend to dominate the learned representations (Liu et al., 2020b; Xie et al., 2020b). Similarly, we leverage a perturbation strategy to generate negative samples in contrastive learning. Given a collection of node sample pairs $\{s_1^i, s_2^i\}_{i=1}^{\mathcal{N}}$, we contrast the positive pair $x = \{s_1^i, s_2^i\}$ and the negative pair $y = \{s_1^i, s_2^j\}$. For example, given an anchor sample s_1^1 that consists of three modalities $(c_{1,i}^v, c_{1,i}^a, c_{1,i}^t)$ and its positive sample s_2^1 represented as $(c_{2,i}^v, c_{2,i}^a, c_{2,i}^t)$, we propose a perturbed negative sample s_2^j represented as $(c_{2,j}^v, c_{2,d(j)}^a, c_{2,j}^t,)$, where $d(\cdot)$ is a perturbing function producing a random index from the sample set. As a result, the multi-modal encoder has to discriminate between the positive sample and the negative sample with only one modality difference. Thus, with the perturbed negative sample, it becomes especially challenging for a network to tell whether the sample containing the perturbed modality $c_{2,d(j)}^m$ is truly negative. Therefore, the challenging negative samples encourage learning the correlation of different modalities with the perturbing function.

5.1.1.4 Multi-task Training in Multi-modal Recommendation

Having obtained a positive pair from two randomly augmented views and a negative pair with a positive sample and challenging negatives, we adopt the InfoNCE loss (c.f. Equation (2.11)), to maximise the agreement of positive pairs and minimise that of the negative pairs. To improve recommendations with self-supervised learning, we adopt a multi-task training strategy to jointly optimise the BPR loss $\mathcal{L}_{bpr}$ (c.f. Equation (2.4)) and the SSL loss $\mathcal{L}_{ssl}$ (c.f. Equation (2.11)): $\mathcal{L} = \lambda_1 \mathcal{L}_{ssl} + \lambda_2 \mathcal{L}_{bpr}$.

Table 5.1: Statistics of the TikTok and MovieLens-1M datasets.

	TikTok	MovieLens-1M
Users	48,524	12,674
Items	84,236	4,214
Interactions	4,751,504	1,013,573
Interaction density(%)	0.0012	0.0084
Visual Dimension	128	128
Acoustic Dimension	128	128
Textual Dimension	128	100

5.1.2 Experiments

To demonstrate the effectiveness of MMGCL and illustrate the reasons for this effectiveness, we conduct experiments to answer the following research questions:

- **RQ5.1**: How does MMGCL perform top-K multi-modal recommendation compared with the general and multi-modal baselines?
- **RQ5.2**: How do different augmentations (i.e., modality masking and modality edge dropout) and the negative sampling strategy impact MMGCL's performance?
- **RQ5.3**: Can we leverage MMGCL to achieve faster convergence speed compared to the used baselines?

5.1.2.1 Datasets and Experimental Settings

To evaluate our MMGCL model, we conduct experiments on two public multi-modal datasets: *TikTok*[1] and *MovieLens* [2]. The statistics of the datasets are listed in Table 5.1. For each dataset, we follow (Wei et al., 2019) to remove the users with less than ten interactions and preserve items with more than ten interactions. Moreover, we proceed with a dimension reduction operation (Du et al., 2020) on the visual features from a 1024 dimension vector to 128 dimensions in the MovieLens-1M dataset to reduce the redundancies of the embeddings.

5.1.2.2 Baselines

To evaluate the effectiveness of our model, we compare MMGCL with baselines, including NGCF, LightGCN and SGL, where NGCF and LightGCN are introduced in Section 3.1. In addition, details of MMGCN can be found in Section 3.2.

[1] `http://ai-lab-challenge.bytedance.com/tce/vc/` [2] `https://grouplens.org/datasets/mov`

Table 5.2: Effectiveness of MMGCL and the baselines. Improvements over the baselines are statistically significant with a $p-$value < 0.01 using the student's t-test.

Dataset	TikTok		MovieLens-1M	
Methods	Recall@10	NDCG@10	Recall@10	NDCG@10
NGCF	0.1783	0.3861	0.4376	0.3162
LightGCN	0.1896	0.4323	0.4695	0.3381
MMGCN	0.1935	0.4315	0.4684	0.3359
SGL	<u>0.1951</u>	<u>0.4357</u>	<u>0.4702</u>	<u>0.3510</u>
MMGCL	**0.2067**	**0.4681**	**0.4943**	**0.3713**
%Improve.	5.95%	7.44%	5.13%	5.77%
p-value	$4.34e-10$	$5.83e-10$	$1.97e-7$	$3.46e-7$

5.1.2.3 Evaluation Protocol and Hyper-parameter Settings

We randomly split a given dataset into training, validation, and testing sets with an 8:1:1 ratio. We adopt Recall@K and NDCG@K (c.f. Section 2.4) to evaluate the performance of top-K recommendations. We set K to 10 and report the averaged performance achieved for all users in the testing set. The negative items of each user are defined as those having no interactions with the user. We adopt the Xavier initialisation to initialise all the model parameters and use Adam optimiser for model optimisation with a batch size of 1024. We apply early-stopping during training, terminating the training when the validation loss does not decrease for 50 epochs. Moreover, we tune the hyper-parameters on the validation set. The learning rate is selected from $\{10^{-2}, 10^{-3}, 10^{-4}\}$. For those hyper-parameters unique to MMGCL, we tune λ_1, τ and ρ within the ranges of $\{0.1, 0.2, 0.5, 1.0\}$, $\{0, 0.1, 0.2, ..., 1.0\}$ and $\{0, 0.1, 0.2, ..., 0.9\}$, respectively. Moreover, we also tune p_v, p_a, p_t in the same range of $\{0, 0.1, 0.2, ..., 1.0\}$.

In the following sections, we aim to validate our argument that graph-based recommender systems can be enhanced by leveraging multiple modalities within a self-supervised learning (SSL) paradigm. To this end, we evaluate the effectiveness of our MMGCL model against existing baselines. We also perform an ablation study on the proposed graph augmentations and challenging negative samples to further confirm their usefulness. Finally, we examine the training efficiency of our model in comparison to the baselines.

5.1.2.4 Performance Comparison with Baselines (RQ5.1)

Table 5.2 compares the performance of our MMGCL model to that of four recommendation baselines, including both established general (NGCF, LightGCN, SGL) and multi-modal recommendation models (MMGCN). In Table 5.2, the top and second-best results are highlighted in bold and underlined, respectively. We also evaluate the statistical significance of the difference in performance between our MMGCL model and that of the used baselines according to the student t-test. As shown in Table 5.2, MMGCL significantly outperforms all baseline models on both Recall and NDCG metrics for both used datasets. This indicates the effectiveness of our proposed modality-specific graph augmentations and challenging negative sampling method within our MMGCL model. In particular, we observe from Table 5.2 that MMGCL outperforms MMGCN by a large margin, which demonstrates the rationality and effectiveness of incorporating self-supervised learning in top-K multi-modal recommendation. One possible reason for this phenomenon is that certain modalities of MMGCN dominate in the learned representations and the rest of the modalities are ignored. Table 5.2 also shows that MMGCL outperforms SGL notably. This result indicates that applying self-supervised learning on modality-specific interaction graphs is indeed more effective than that of modelling ID-based interaction graphs (c.f. Section 3.2). Hence, we answer RQ5.1 as follows: our MMGCL model effectively leverages graph augmentations and challenging negative samples within the SSL paradigm across multiple modality-specific graphs, significantly enhancing multi-modal recommendation performance compared to all baselines.

5.1.2.5 Effect of Graph Augmentations and Challenging Negatives (RQ5.2)

In this section, we aim to investigate the effectiveness of our proposed graph augmentations and negative sampling method in top-K multi-modal recommendation. Figure 5.2 reports the performance outcomes of MMGCL' variants on both Recall and NDCG metrics across the two used datasets. We use ED/MM/CN as the abbreviations of Modality Edge Dropout, Modality Masking and Challenging Negatives, respectively. As shown in Figure 5.2, the MMGCL variant with multi-view (MMGCL$_{ED+MM}$) outperforms those with single views (MMGCL$_{MM}$ & MMGCL$_{ED}$). This combination of ED and MM augmentations demonstrates the successful exploitation of the fundamental supervisory signal, namely the co-occurrence of multiple views of the users' preferences. For the comparison between single-view augmentations, the performance of MMGCL$_{MM}$ is on a par with MMGCL$_{ED}$ on TikTok but not competitive with MMGCL$_{ED}$ on Movielens. One possible reason is that TikTok has more effective pre-extracted embeddings than MovieLens-1M and MMGCL$_{MM}$ masks the most effective content from multiple modalities. Moreover, MMGCL$_{ED}$ outperforms MMGCL$_{MM}$ in all instances across both metrics and datasets (4 out of 4), thereby indicating that perturbing the graph structure can capture more useful inherent patterns on the user's potential interests. Next, as shown in Figure 5.2, the performance improvement indeed stems from the proposed challenging negatives, as evidenced by the comparison between MMGCL$_{ED+MM+CN}$ and MMGCL$_{ED+MM}$, where the latter only applies the augmentations. This reveals that our negative sampling method provides more factual negative samples by the modality-specific perturbing strategy. Hence, we answer RQ5.2 as follows: MMGCL successfully leverages the multi-view augmentation across modalities to learn effective representations and further enhances performance by incorporating challenging negatives during training under an SSL paradigm.

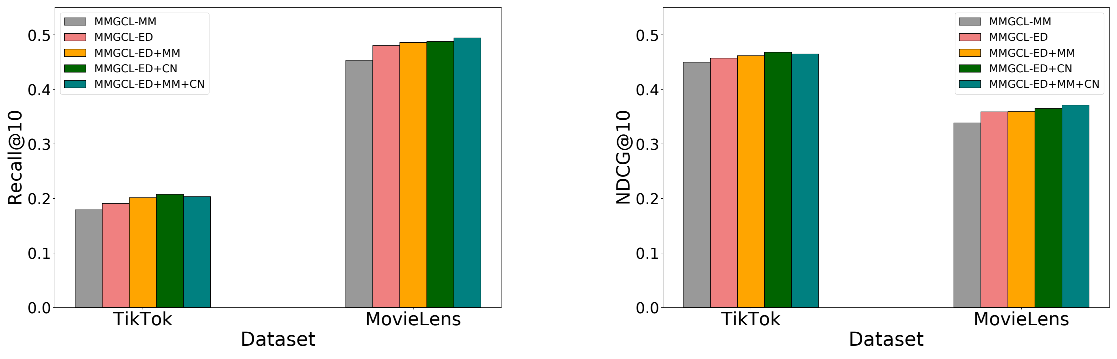


Figure 5.2: Performances in terms of Recall@10 and NDCG@10 on the MovieLens-1M and TikTok datasets.

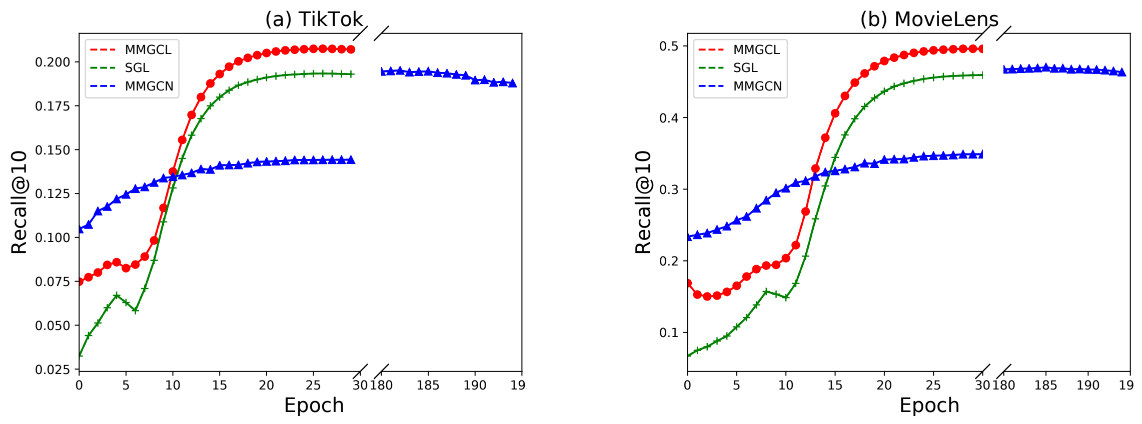


Figure 5.3: Training curves of MMGCL, SGL and MMGCN on the MovieLens-1M and TikTok datasets.

5.1.2.6 Training Efficiency (RQ5.3)

Previous work (Wu et al., 2021a) has shown the superiority of self-supervised learning in training efficiency. Thus, we further study the training efficiency on the implementations of multi-view augmentation and the challenging negative samples. Figure 5.3 shows the training curves of MMGCL, SGL and MMGCN on Recall across the TikTok and MovieLens-1M datasets. We only report the Recall metric since we observe same conclusion on NDCG. We observe from Figure 5.3 that MMGCL is much faster to converge than MMGCN on both datasets. In particular, MMGCL arrives at the best performance after 20 epochs, while MMGCN takes more than 180 epochs in these two datasets, respectively. This suggests that our proposed MMGCL method can greatly reduce the training time compared to MMGCN while achieving significant improvements and outperforming SGL through all epochs in Figure 5.3. Hence, we answer RQ5.3 as follows: Our proposed MMGCL model successfully improves training efficiency with an SSL loss and provides large gradients during training by contrasting augmented positive samples and challenging negative samples.

5.1.3 Discussion

In this section, we have demonstrated that modality edge dropout, modality masking, and challenging negative samples effectively enhance modality fusion within the SSL paradigm. These techniques enable the model to better learn inter-modality correlations, thereby improving top-K multi-modal recommendation performance. We have validated that graph-based recommender systems can be enhanced by mining richer supervision signals from multiple modalities through an more effective modality fusion. Although the proposed graph augmentations and challenging negatives generate more effective multi-modal user/item representations, existing multi-modal graph-based recommender systems, as discussed in Section 3.2, still rely heavily on features extracted independently from each modality using pre-trained modality-specific encoders. This leads to only shallow modality alignment. Ideally, these systems should capture deeper cross-modal relationships to further enhance modality fusion for top-K multi-modal recommendation. Therefore, in Section 5.2, we extend MMGCL and other multi-modal graph-based recommenders by enhancing the fusion quality through deeper modality alignment at feature extraction stage using using Large Multi-modal (LMM) encoders, aiming to further improve modality fusion and top-K recommendation performance.

5.2 Modality Fusion with Deep Alignment

While the modality-specific graph augmentations proposed in MMGCL (Section 5.1) offer a promising way to enhance modality fusion, as a further step, we now investigate a more fundamental bottleneck that affects not only MMGCL but all multi-modal graph-based recommender systems. As discussed in Section 3.2, existing multi-modal graph-based recommender systems (Kim et al., 2022; Liu et al., 2022c; Pan et al., 2022) perform modality fusion by integrating the extracted multi-modal features into user/item representations without sufficiently addressing the complex and inherent correlations between different modalities (Wei et al., 2023). To understand how each modality contributes to the recommendation performance, we conduct a preliminary analysis using two well-known multi-modal recommendation models, namely MMGCL (proposed in Section 5.1) and LATTICE (c.f. Section 3.2), across three commonly used recommendation datasets. As illustrated in Figure 5.4, both MMGCL and LATTICE fail to effectively fuse the multi-modal features. Indeed, contrary to our expectations, these models show a suboptimal performance when integrating both the visual and textual features compared to when using single modality data, such as the textual or visual features independently. As discussed in Section 3.2, we argue that the suboptimal performance is caused by the shallow alignment in modality fusion, since the used methods cannot address the complex and inherent correlations between these modalities. To address this issue of shallow alignment in multi-modal recommendation, we propose to leverage deep alignment techniques to enhance the suboptimal modality fusion. Recent progress in multi-modal representation learning has concentrated on developing

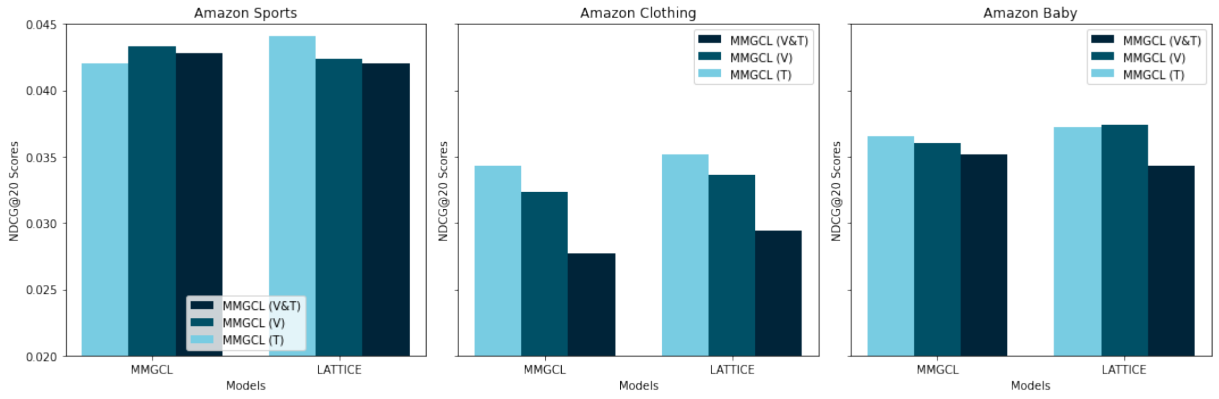


Figure 5.4: Comparison of the NDCG@20 scores of the MMGCL and LATTICE models using different modality inputs across the used datasets. V & T are the abbreviations for Visual & Textual, respectively.

Large Multi-Modal (LMM) encoder architectures that facilitate deeper alignment across different modalities (c.f. Figure 5.6). From a structural standpoint, such LMM encoders may be dual-stream, such as CLIP (Radford et al., 2021), or single-stream (unified), such as VLMo (Bao et al., 2022) and BEiT-3 (Wang et al., 2023c), where both types of architecture aim to mitigate information loss and capture cross-modal interactions (Radford et al., 2021) (c.f. Figure 5.5). These LMM encoders enforce deep alignment between the modalities by using multiple transformer layers, which consolidate various modalities into a unified semantic space. Moreover, these LMM encoders leverage pre-training on extensive web data corpora to obtain foundational knowledge about the relationships between item images and descriptions. Indeed, the LMM encoders have been shown to result in an increased effectiveness on downstream tasks such as image-text retrieval (Rao et al., 2022) and zero-shot classification (Bao et al., 2022). However, to-date, these LMM encoders have not been used for top-K multi-modal recommendation.

In this section, we tackle the above knowledge gap by providing a comprehensive comparison between traditional multi-modal recommendation approaches that do not attempt to explicitly align multi-modal item embeddings using SSL, and those same approaches when enhanced with LMM encoders across four multi-modal (text+image) recommendation datasets.[3] By doing so, we provide strong conclusions on whether the SSL-enhanced deep alignment techniques benefit state-of-the-art multi-modal recommender systems, as well as actionable insights on how such encoders should be trained.

[3] Although our study primarily focuses on visual and textual modalities, the findings may provide insights into the generalisation of such approaches to other modalities, warranting further investigation in future research.

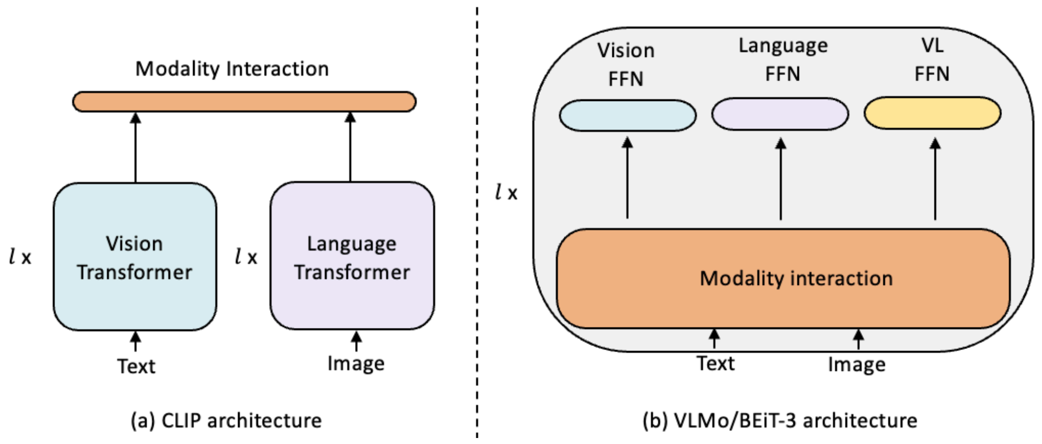


Figure 5.5: The architectures of the existing large multi-modal encoders.

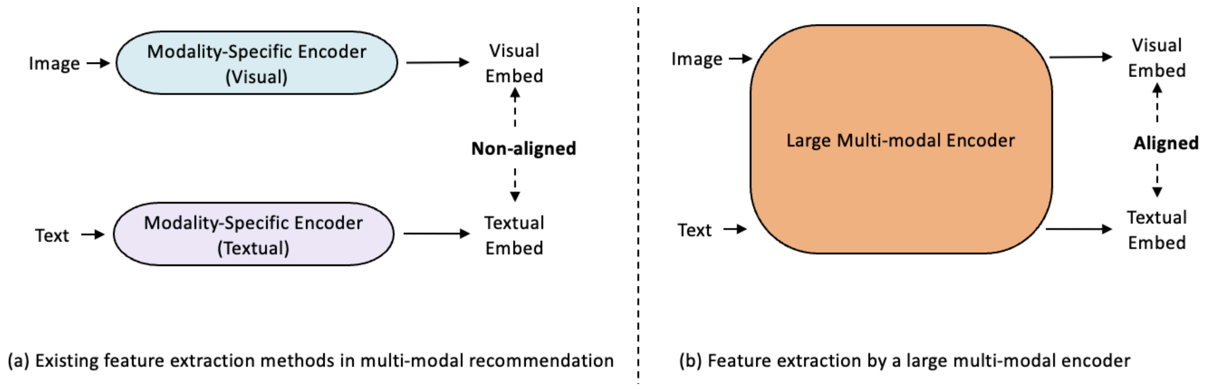


Image
Modality-Specific Encoder (Visual)
Visual Embed
Non-aligned
Text
Modality-Specific Encoder (Textual)
Textual Embed
(a) Existing feature extraction methods in multi-modal recommendation
Image
Large Multi-modal Encoder
Visual Embed
Aligned
Text
Textual Embed
(b) Feature extraction by a large multi-modal encoder

Figure 5.6: An illustration of different feature extraction methods.

5.2.1 Large Multi-modal (LMM) Encoders

The remarkable success of transformer-based pre-training in the Natural Language Processing (NLP) community has led to extensive research on multi-modal pre-training, particularly as various large-scale multi-modal corpora have emerged. The self-attention mechanism finds global patterns by examining all word connections, regardless of distance, and effectively captures long-range dependencies without using fixed windows or sequential processing. This inherent characteristic allows a transformer to operate in a modality-agnostic manner compatible with various modalities. Recent studies (Rao et al., 2022; Wang et al., 2023f; Yi and Ounis, 2024; Zhai et al., 2023; Zhu et al., 2023) have shown that when pre-trained on large-scale multi-modal corpora, transformer-based models not only significantly outperform their competitors (such as traditional recurrent neural networks and convolutional neural networks) across a wide range of multi-modal downstream tasks, and are effective for zero-shot scenarios, where it is important to be able to generalise to new tasks or domains without any task-specific fine-tuning or additional training. In the literature of multi-modal encoders, there are primarily two lines of approaches: (1) *dual-stream architectures*, such as VSE (Faghri et al., 2018), CLIP (Radford et al., 2021), ViLBERT (Lu et al., 2019), which consist of a vision transformer and a language transformer. The vision transformer processes images, while the language transformer handles textual data. Both encoders generate embeddings for their respective inputs, which are then aligned using several fusion layers; (2) *unified architectures*, such as VLMo (Bao et al., 2022) or BEiT-3 (Wang et al., 2023c), which jointly process multi-modal data into a Mixture-of-Modality-Experts (MoME) Transformer to obtain contextualised representations and align the visual and language feature vectors. The MoME Transformer is used to encode different modalities, with a mixture of modality experts replacing the feed-forward network of a standard Transformer. Each MoME Transformer block captures modality-specific information by switching to a different modality expert and employs multi-head self-attention (MSA) shared across modalities to align visual and text content. To highlight the difference between VLMo and BEiT-3, it is important to note that they use different pre-training strategies and have different architectural designs. VLMo typically involves a variety of SSL-based auxiliary tasks in a stage-wise training process, including image-text matching, image-text contrast, and word-patch alignment tasks. In contrast, BEiT-3 uses a more simplified SSL-based pre-training methodology as it trains with both single-modal and multi-modal datasets, thereby avoiding the multi-stage complexity inherent to the VLMo's methodology. From an architectural design standpoint, BEiT-3 set itself apart by expanding the number of its transformer layers from 24 to 40. Such an increase in the number of layers number leads to a deeper alignment between modalities, allowing to handle more complex multi-modal data representations with a higher effectiveness (Wang et al., 2023c). As discussed in Section 3.2, feature alignment and feature extraction are important research points in multi-modal recommender systems. These extraction and alignment processes capture and exploit the relationships between different modalities and generate more effective representations for downstream tasks. Despite the benefits offered by the LMM encoders, the integration of these LMM encoders into recommender systems has not been investigated extensively to-date. Hence, in the second part of this chapter, we choose three representative LMM encoders, CLIP, VLMo and BEiT-3, each representing distinct architectural approaches to multi-modal encoders, for the purpose of extracting and aligning multiple modalities in the context of the top-k multi-modal recommendation task under various training paradigms.

5.2.2 Probing Large Multi-modal Encoders in Recommendation

In light of the advantages of the deep alignment techniques for multi-modal representation learning, as discussed in Section 5.2.1, we detail the settings in which we extend the use of the LMM encoders for the recommendation task. First, we describe the process of using multi-modal embeddings, obtained from CLIP, VLMo and BEiT-3, to initialise the user/item embeddings in the existing recommendation models. Then, we describe the methods for fine-tuning these

LMM encoders using the recommendation datasets and illustrate the integration of these LMM encoders with the existing recommendation models in an end-to-end approach.

Table 5.3: Notations used in this section.

Symbol	Description
m	the multi-modal embeddings of the token
l	the layer number
LN	the layer normalisation operation
MSA	the multi-head self-attention operation
FFN	the feed-forward network
p_{i2t}	the softmax-normalised image-to-text similarities
p_{t2i}	the softmax-normalised text-to-image similarities
N	the batch size
T	the length of the text sequence
$w_j^{(i)}$	the j-th word in the i-th text sequence
$w_{<j}^{(i)}$	the prefix of the i-th text sequence up to the j-th word
$\mathcal{U}$	the user set
$R(u)$	the ground-truth set of items that user u has interacted with
$\hat{R}(u)$	the ranked list of items
M, M'	the masking vectors on the edge set $\mathcal{E}$
N_u	the neighbour nodes of user u in the interaction graph
N_i	the neighbour nodes of item i in the interaction graph
$R(u)$	the ground-truth set of items that user u has interacted with
$\hat{R}(u)$	the ranked list of items generated by a recommender

5.2.2.1 Preliminaries on Large Multi-modal Encoders

We first introduce the most representitive LMM encoders, showing the input representations, propagation functions, and SSL-based pre-training objectives of these LMM encoders.

Input representations: In the following, we describe the input of the LMM encoders we use in this section:

- **CLIP [4]:** For CLIP, raw images and texts are encoded into image and text vector representations. CLIP leverages the Vision Transformer (ViT) architecture (Kolesnikov et al., 2022) to process image representations by dividing the input image into non-overlapping patches, flattening them into vectors, and linearly projecting them to create patch embeddings. Text representations are generated using the GPT-2 (Radford et al., 2019) model, after tokenising the raw text input using byte pair encoding (BPE) and adding positional embeddings.

[4] We choose CLIP as the dual-stream architecture due to its demonstrated effectiveness in diverse tasks such as information retrieval and visual question answering (Bao et al., 2022; Radford et al., 2021). In addition, there are currently no other dual-stream encoders that are directly applicable to the recommendation context without extensive modifications to their architecture and functionality.

- **VLMo & BEiT-3:** Similar to CLIP, raw images and texts are encoded into image, and text vector representations. Image representations are created by splitting input images into patches, flattening them, and linearly projecting them to form patch embeddings. A learnable special token [I_CLS] is added to the sequence, and image input representations are computed by summing the patch embeddings, 1D position embeddings, and linear projection, respectively. Text representations are generated using BERT tokenisation and WordPiece for subword units, with a start-of-sequence token ([T_CLS]) and a boundary token ([T_SEP]) added. Text input representations are generated by summing the word, position, and type embeddings.

To illustrate how the multi-modal embeddings of CLIP, VLMo and BEiT-3 can be used as initial item embeddings in multi-modal recommendation models, we use the VLMo-Base Plus model variant as an example. The resulting text embeddings for all items from the VLMo-Base Plus encoder have a shape of [item number, text_token_length+2, 544], where each item comprises the number of raw text tokens along with [CLS] and [SEP] tokens. As for image embeddings, each item has an embedding shape of [197, 544]. We obtain the item visual and text embeddings by selecting the 544-dimensional vector corresponding to the [CLS] token for each item embedding, which should encapsulate rich, high-level information in each modality.
Propagation functions: In the following, we describe the propagation functions of the used LMM encoders:

- **CLIP:** CLIP leverages a dual-stream architecture to encode distinct modalities, incorporating a separate stream for each modality—visual and textual—while maintaining shared multi-head self-attention layers to enable alignment and interaction between visual and textual content. Each stream consists of a series of transformer blocks. Hence, the propagation function is defined as follows:

$$h_i^{(l)} = \text{LN}\left(h_i^{(l-1)} + \text{MSA}(h_i^{(l-1)}, h_i^{(l-1)}, m_i)\right) \tag{5.3}$$

  where $h_i^{(l)}$ is the hidden state of the i-th token in the l-th layer, LN and MSA are the layer normalisation operation and the multi-head self-attention mechanism, respectively. and m_i is the corresponding multi-modal embedding of the token.

- **VLMo & BEiT-3:** As unified LMM encoders, VLMo and BEiT-3 both use the MoME transformer to encode different modalities, with a mixture of modality experts substituting the feed-forward network of a standard Transformer (Vaswani et al., 2017). Each MoME transformer block captures modality-specific information by switching to a different modality expert and employs multi-head self-attention (MSA) shared across modalities to align visual and textual content. Hence, the propagation function for both VLMo

& BEiT-3 is defined as follows:

$$h_i^{(l)} = \mathrm{LN}\left(h_i^{(l-1)} + \mathrm{MSA}(h_i^{(l-1)}, h_i^{(l-1)}, m_i) + \mathrm{FFN}(h_i^{(l-1)}, m_i)\right) \tag{5.4}$$

where FFN is the feed-forward network. As such, this mechanism of MoME-FFN is capable of selecting an expert among multiple modality experts to process the input according to the modality of the input vectors and the index of the Transformer layer. There are three modality experts: vision expert (V-FFN), language expert (L-FFN), and vision-language expert (VL-FFN). The choice of a given expert depends on the input modality and the layer within the transformer architecture. The contextualised representations for image-only, text-only, and image-text inputs are obtained accordingly.

Pre-training objectives of the LMM encoders: In order to study the impact of finetuning the LMM encoders, we first describe their training objectives before adapting these encoders to the recommendation task:

- **Image-Text Contrastive (ITC) loss:** All LMM encoders (CLIP, VLMo and BEiT-3) leverage an SSL-based ITC loss, which aims to encourage the model to learn a joint embedding space where the similarity between an image and its corresponding text is maximised, while the similarity between mismatched image-text pairs is minimised. ITC loss is defined as follows:

$$\mathcal{L}_{\mathrm{ITC}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{p_{i2t}(i)}{\sum_{j\neq i}^{N} p_{i2t}(j)} - \frac{1}{N}\sum_{i=1}^{N} \log \frac{p_{t2i}(i)}{\sum_{j\neq i}^{N} p_{t2i}(j)} \tag{5.5}$$

  where N is the batch size, while $p_{i2t}(i)$ and $p_{t2i}(i)$ are the softmax-normalised image-to-text and text-to-image similarities of the i-th pair, respectively. As a result, a joint embedding space is learned using a contrastive loss (Wu et al., 2021a), where the similarity between an image and its corresponding text encourages the encoder to generate more aligned embeddings.

- **Masked Language Modeling (MLM) loss:** This loss function is exclusively used by the VLMo and BEiT-3 encoders. These encoders randomly select and mask tokens in the text sequence with a 15% masking probability, as in BERT (Kenton and Toutanova, 2019a). Both the VLMo and BEiT-3 encoders are trained to predict these masked tokens, using unmasked tokens and visual cues. The MLM loss is computed as follows:

$$\mathcal{L}_{\mathrm{MLM}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{T_i} \log p(w_j^{(i)} | w_{<j}^{(i)}, m_i) \tag{5.6}$$

  where N is the batch size, T_i is the length of the i-th text sequence, $w_j^{(i)}$ is the j-th word in the i-th text sequence, $w_{<j}^{(i)}$ is the prefix of the i-th text sequence up to the j-th word, and

Table 5.4: The used training losses for the existing recommendation models.

Method	VBPR	MMGCN	MMGCL	SLMRec	LATTICE
BPR	✓	✓	✓	×	✓
Contrastive loss	×	×	✓	✓	×

m_i is the corresponding multi-modal embedding of the i-th text sequence. As such, predicting masked tokens in the presence of a visual context enables the encoder to generate embeddings that better capture the joint representation of image and text data, leading to enhanced multi-modal representation learning.

- **Image-Text Matching (ITM) loss:** This is also an SSL-based loss solely used by VLMo. VLMo uses the final hidden vector of the [T_CLS] token to represent the image-text pair, employing a cross-entropy loss for binary classification. Hard negatives are sampled from the training examples for this purpose. ITM loss is formulated as follows:

$$\mathcal{L}_{\text{ITM}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log p(y_i = 1|\mathbf{h}_i) + (1 - y_i)\log p(y_i = 0|\mathbf{h}_i)\right] \tag{5.7}$$

where y_i is the ground truth label (matched or unmatched) for the i-th image-text pair, and $\mathbf{h}_i$ is the final hidden vector of the [T_CLS] token representing the pair. The loss is computed using a binary cross-entropy loss function. By using a binary cross-entropy loss and hard negative mining, VLMo is expected to learn to differentiate between matched and unmatched image-text pairs. This process encourages the encoder to generate more accurate and semantically aligned multi-modal embeddings, thereby improving the model's overall capability to mine relationships between image and text data.

5.2.2.2 Training Strategies of the LMM Encoders

In this section, we present various training paradigms for incorporating the LMM encoders into existing multi-modal recommendation models. We discuss the optimisation objectives used to tune both the LMM encoders and the recommendation models. We aim to identify the best training paradigm that facilitates the effective integration of the LMM encoders, leveraging their strengths to enhance the performance of existing recommendation models:

- **Two-stage training** involves first fine-tuning the LMM encoders on item images and texts, and then using the derived item embeddings as initial embeddings for each modality in the existing multi-modal recommendation models. This fine-tuning process is designed to enhance the adaptability and performance of the LMM encoders in the context of recommendation scenarios. Specifically, we tune CLIP, VLMo and BEiT-3 using the ITC loss (Bao et al., 2022; Radford et al., 2021), as detailed in Equation (5.5);

Algorithm 5.1 End-to-end training

```
Input: Dataset path, raw images, concatenated texts (title, description, brand, categorical info)
Output: Trained recommendation model
Step 1: Load data data_loader ← DataLoader(dataset_path, raw_images, concatenated_texts)
Step 2: Initialise and load pre-trained weights for CLIP/VLMo/BEiT-3 encoder clip/vlmo/beit3 ← CLIP()/VLMo()/BEiT-3() clip/vlmo/beit3.load_pretrained_weights()
Step 3: Generate embeddings with CLIP/VLMo/BEiT-3 encoder image_embeddings, text_embeddings, image_text_embeddings ← clip/vlmo/beit3.generate_embeddings(data_loader)
Step 4: Integrate embeddings into the recommendation model rec_model ← REC(image_embeddings, text_embeddings, image_text_embeddings)
Step 5: End-to-end training for $epoch \leftarrow 1$ to $num\_epochs$ do
    // Forward pass
    user_item_scores ← rec_model.forward()
    // Compute loss
    loss ← compute_loss(user_item_scores, ground_truth)
    // Backward pass and optimisation
    optimiser.zero_grad() loss.backward() optimiser.step()
    // Update model with new embeddings
    rec_model.update(image_embeddings, text_embeddings, image_text_embeddings)
    // Evaluate and print performance metrics
    evaluate_and_print_metrics(epoch, rec_model)
end
```

- **End-to-end training** typically needs the seamless integration of the LMM encoders so as to jointly optimise both the LMM encoders and the existing recommendation models with the recommendation loss. This integration must account for the different types of losses used by each model, such as the BPR loss (Rendle et al., 2009) and the contrastive loss (Wu et al., 2021a). Table 5.4 presents the training losses used in each recommendation model. Notably, this end-to-end training process does not incorporate the previously mentioned pre-training losses of the LMM encoders in Section 5.2.2.1, such as the ITC, MLM and ITM losses. Although we acknowledge that some recommendation models (e.g., LATTICE) facilitate feature alignment and may have overlapping concepts with CLIP, VLMo and BEiT-3, our analysis focuses on the impact of integrating these LMM encoders into various recommendation architectures. Specifically, we leverage the original implementations provided by the authors for the CLIP, VLMo and BEiT-3 architectures to process the item images and their corresponding text to extract the visual and textual embeddings. These embeddings are then used to initialise the item representations in the downstream recommendation models. Algorithm 5.1 presents the pseudo-code for end-to-end training.

5.2.3 Experiments

In this section, we conduct experiments to examine the effectiveness of LMM encoders on four public datasets, in comparison to five existing state-of-the-art multi-modal recommendation models. To evaluate the impact of the deep alignment of LMM encoders in the top-K recommendation task, we conduct experiments to answer the following five research questions:

- **RQ5.3**: Do the pre-trained LMM encoders (CLIP, VLMo and BEiT-3) outperform existing modality-specific extractors in multi-modal recommendation?
- **RQ5.4**: Is the two-stage training paradigm, which includes fine-tuning the LMM encoders with raw images and textual item descriptions, more effective in enhancing the recommendation performance than the pre-trained LMM encoders?
- **RQ5.5**: Does the end-to-end training paradigm enhance the deep alignment of modalities within the existing multi-modal recommendation models more effectively than the two-stage training approach?
- **RQ5.6**: Does the use of LMM encoders improve the contribution of each modality to the overall recommendation performance in comparison to the use of existing modality-specific encoders?
- **RQ5.7**: How do the dual-stream (i.e., CLIP) and unified (i.e., VLMo, BEiT-3) LMM encoders compare in terms of efficiency and effectiveness in multi-modal recommendation?

5.2.3.1 Datasets and Experimental Settings

To assess the performance of CLIP, VLMo and BEiT-3 in the top-K recommendation task, we carry out experiments on four different datasets compared to the use of Tiktok and MovieLens-1M in Section 5.1. In particular, we use three widely used Amazon Review datasets [5] – Sports and Outdoors (abbreviated as Sports), Clothing, Shoes and Jewelry (referred to as Clothing), and Baby – as well as an additional video recommendation dataset: Bilibili [6]. These datasets provide rich multi-modal data instead of pre-extracted embeddings, such as item image URLs and their descriptions, enabling a robust and representative evaluation of the multi-modal recommendation performance (Zhang Jinghao et al., 2021). Processing the datasets is a crucial step in multi-modal recommendation, since we need to handle unprocessed datasets containing extensive information from different modalities. Following the dataset processing protocol, widely used in previous works, we transform the ratings into binary values of 0 or 1, indicating whether the user has rated the item (Meng et al., 2024a,b; Yi et al., 2023a; Zhang Jinghao et al., 2021; Zhou et al., 2023a). In line with previous works, we filter out users and items with more than

[5] `https://jmcauley.ucsd.edu/data/amazon/` [6] `https://github.com/westlake-repl/NineRec`

Table 5.5: Statistics of the used multi-modal recommendation datasets.

	Amazon Sports	Amazon Clothing	Amazon Baby	Bilibili
Users	35,598	39,387	19,445	16,525
Items	18,287	22,499	7,037	3,506
Interactions	295,366	271,001	160,522	115,576
Interaction Density	0.00045	0.00030	0.00012	0.00020

Table 5.6: Comparison of the visual/textual dimensions and the number of model parameters of the modality-specific and large multi-modal encoders.

Uni-modal and Multi-modal Encoders	Values
Visual Dimension: CNN/CLIP/VLMo/BEiT-3	4096/768/544/544
Textual Dimension: Sentence-Transformer/CLIP/VLMo/BEiT-3	384/768/544/544
Parameters (million): CNN+Sentence-Transformer/CLIP/VLMo/BEiT-3	170/151/167/228

5 interactions in a given dataset (Zhang Jinghao et al., 2021). The exact statistics of the used datasets are presented in Table 5.5. Unlike existing approaches (He and McAuley, 2016b; Wei et al., 2019; Yi et al., 2022; Zhang Jinghao et al., 2021) that use pre-extracted features within the datasets, we download the raw images from the item URLs and encode them with the LMM encoders, instead of using pre-extracted 4096-dimensional visual features of items (Ni et al., 2019). For the textual features, we employ the title, description, brand, and categorical information of items and also encode them with the LMM encoders. In contrast to existing approaches that use Sentence-Transformer to extract 384-dimensional textual embeddings (Zhang Jinghao et al., 2021), we use CLIP and VLMo/BEiT-3 to encode the raw text of items into 768 and 544 dimensions, respectively. Table 5.6 presents a detailed comparison of all existing modality-specific and LMM encoders along with their respective modality dimensions.

5.2.3.2 Evaluation Protocol

Similar to the evaluation setting in (Yi et al., 2023b; Zhang Jinghao et al., 2021; Zhou et al., 2023a), we randomly split the datasets into training, validation, and testing sets with an 8:1:1 ratio. To perform negative sampling for each user, we sample items that have no prior interactions with the user from the history of observed user-item interactions. We use Recall@K and NDCG@K (c.f. Section 2.4) to evaluate the performance of top-K recommendation. We follow (Zhang Jinghao et al., 2021) in setting K = 20 and report the average performance achieved for all users in the testing set. We use the Adam (Kingma and Ba, 2014) optimiser in the LMM enhanced models and the five baseline models. We apply an early-stopping strategy during training, terminating the training when the validation loss does not decrease for 50 epochs.

5.2.3.3 Baselines

To examine the effectiveness of the LMM encoders, we compare the performance of recommendation models using pre-trained modality-specific encoders — employing CNN and Sentence-Transformer independently for each modality — with that of the pre-trained LMM encoders like CLIP, VLMo and BEiT-3, which jointly extract information from both modalities. In this section, we choose five state-of-the-art multi-modal recommendation models for comparison. These include the established baselines VBPR, MMGCN, SLMRec, and LATTICE (introduced in Section 3.2), alongside our proposed MMGCL model (proposed in Section 5.1).

5.2.3.4 Model Checkpoints and Hyperparameter Settings

All used baselines (VBPR[7],MMGCN[8], MMGCL[9], SLMRec[10], LATTICE[11]) and the LMM encoders (CLIP ViT-B/16[12], VLMo-Base Plus[13], BEiT-3-Base[14]) are implemented with PyTorch and trained on a GPU A6000 with 48GB of memory. To facilitate a comparison of the impact of the LMM encoders on the recommendation effectiveness, we make deliberate choices for the CLIP, VLMo and BEiT-3 variants based on their reported performances in the literature. Specifically, for CLIP, we opt for the ViT-B/16 variant as the image encoder, motivated by its superior performance in image tasks when compared to other CNN image encoders within the CLIP model framework (Radford et al., 2021). For VLMo, our choice is the VLMo-Base Plus model, similarly driven by its demonstrated effectiveness (Bao et al., 2022). We evaluate each encoder used in this study using a comparable range of parameters – specifically, parameter counts of 151 million, 167 million and 228 million – so as to ensure a fair comparison. The architectural differences between CLIP's ViT-B/16 and VLMo-Base/BEiT-3-Base are depicted in Figure 5.5, which offers a visual juxtaposition of their structures.

In our experiments, we use the authors' original code for VLMo, CLIP and BEiT-3, but note that their code for VLMo is implemented with the PyTorch Lightning framework. We converted it into pure PyTorch code without using the PyTorch Lightning library. Our motivations for converting the original code from PyTorch Lightning to pure PyTorch include greater customisation, compatibility, and performance considerations. By using pure PyTorch, we gain more control and flexibility, allowing us to tailor the code to specific requirements, which is beneficial for future research needs. While optimising the LMM encoders, we observe a marked acceleration in training speed upon transitioning from PyTorch Lightning to pure PyTorch. In our quest to pinpoint the most effective hyperparameters, we undertake extensive parameter searches for each recommendation dataset, using metrics from the validation set. These evaluations are per-

[7] `https://github.com/DevilEEE/VBPR` [8] `https://github.com/weiyinwei/MMGCN`

[9] `https://github.com/zxy-ml84/MMGCL` [10] `https://github.com/zltao/SLMRec`

[11] `https://github.com/CRIPAC-DIG/LATTICE` [12] `https://github.com/openai/CLIP`

[13] `https://github.com/microsoft/unilm/tree/master/vlmo`

[14] `https://github.com/microsoft/unilm/tree/master/beit3`

Table 5.7: Experimental results comparing the use of modality-specific encoders with the use of the LMM encoders in multi-modal recommenders. The best performance of each model is highlighted in bold. * denotes a statistically significant difference between each recommendation model and its LMM encoder variant using the paired t-test with $p < 0.05$.

Dataset	Amazon Sports		Amazon Clothing		Amazon Baby		Bilibili	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
VBPR	0.0771	0.0349	0.0611	0.0277	0.0740	0.0329	0.0891	0.0502
VBPR$_{VLMo}$	0.0848*	0.0373*	0.0577*	0.0266*	0.0774*	0.0346*	0.0930*	0.0536*
VBPR$_{BEiT-3}$	**0.0854***	**0.0382***	0.0613*	0.0280	**0.0784***	**0.0355***	**0.0942***	**0.0545***
VBPR$_{CLIP}$	0.0802*	0.0357*	**0.0675***	**0.0296***	0.0762*	0.0336*	0.0920*	0.0527*
MMGCN	0.0475	0.0201	0.0246	0.0100	**0.0642**	**0.0266**	0.0774	0.0428
MMGCN$_{VLMo}$	0.0484*	0.0208*	0.0252*	0.0104*	0.0538*	0.0224*	0.0790*	0.0444
MMGCN$_{BEiT-3}$	0.0488*	0.0213*	0.0250	**0.0106***	0.0564*	0.0240*	0.0824*	0.0485*
MMGCN$_{CLIP}$	**0.0494***	**0.0216***	**0.0253***	0.0101	0.0606*	0.0257	**0.0835***	**0.0501***
MMGCL	0.0913	0.0428	0.0607	0.0277	0.0790	0.0352	0.1124	0.0596
MMGCL$_{VLMo}$	0.0941*	0.0446*	0.0648*	0.0299*	0.0822*	0.0370*	0.1233*	0.0624*
MMGCL$_{BEiT-3}$	0.0953*	**0.0453***	0.0663*	0.0307*	**0.0836***	**0.0384***	0.1252*	0.0632*
MMGCL$_{CLIP}$	**0.0959***	0.0447*	**0.0703***	**0.0321***	0.0804*	0.0362*	**0.1260***	**0.0652***
SLMRec	0.0901	0.0417	0.0623	0.0287	0.0810	0.0361	0.1132	0.0605
SLMRec$_{VLMo}$	**0.0919***	0.0426*	0.0653*	0.0304*	0.0823*	0.0374*	0.1232*	0.0624*
SLMRec$_{BEiT-3}$	0.0917*	**0.0430***	0.0661*	**0.0315***	**0.0828***	0.0375*	**0.1250***	**0.0646***
SLMRec$_{CLIP}$	0.0909*	0.0427*	**0.0671***	0.0304*	**0.0821***	**0.0381***	0.1243*	0.0637*
LATTICE	**0.0944**	**0.0424**	**0.0704**	**0.0336**	**0.0860**	**0.0374**	**0.1181**	**0.0621**
LATTICE$_{VLMo}$	0.0758*	0.0312*	0.0512*	0.0209*	0.0817*	0.0348*	0.1042*	0.0567*
LATTICE$_{BEiT-3}$	0.0807*	0.0364*	0.0578*	0.0235*	0.0822*	0.0345*	0.1066*	0.0581*
LATTICE$_{CLIP}$	0.0785*	0.0332*	0.0593*	0.0263*	0.0830*	0.0354*	0.1034*	0.0561*

formed during both the fine-tuning phase and the end-to-end training. Specifically, we experimented with learning rates spanning $\{1e{-}5, 3e{-}5, 5e{-}5, 1e{-}4, 1e{-}3\}$ and varied batch sizes, including $\{32, 64, 128, 256, 1024\}$. The grid search procedure is conducted in accordance with the available code of MMRec[15] and is applied to all model variants we evaluated in the experiments. In addition, we set the weight decay rate to 0 throughout all the experiments, thus removing model regularisation to enable a comprehensive extraction of complex patterns inherent in the multi-modal recommendation scenario. Our complete code, along with the associated documentation are publicly available at `https://github.com/zxy-ml84/LMM4Rec/`, so as to facilitate the reproducibility of our work.

5.2.3.5 Pre-trained Modality-specific vs. Pre-trained LMM Encoders (RQ5.4)

As discussed in Section 5.2.3.4, to ensure a fair comparison, we primarily focus on the results obtained from recommendation models that use pre-trained modality-specific encoders. These models employ CNN and Sentence-Transformer independently for each modality, providing

[15] `https://github.com/enoche/MMRec`

a consistent baseline for evaluating the impact of deep alignment through the incorporation of LMM encoders (CLIP, VLMo and BEiT-3). These results are compared with those using CLIP ViT-B/16, VLMo-Base Plus and BEiT-3 Base, which jointly extract information from both modalities. To evaluate the statistical significance of performance differences between the five selected recommendation models with and without the integration of LMM encoders, we use the paired t-test ($p < 0.05$). Table 5.7 presents the results of our conducted experiments, comparing the performance of recommendation models using pre-trained modality-specific encoders (VBPR, MMGCN, MMGCL, SLMRec, LATTICE) with those employing the pre-trained LMM encoders (VBPR$_{CLIP/VLMo/BEiT-3}$, MMGCN$_{CLIP/VLMo/BEiT-3}$, MMGCL$_{CLIP/VLMo/BEiT-3}$, SLMRec$_{CLIP/VLMo/BEiT-3}$, LATTICE$_{CLIP/VLMo/BEiT-3}$) in the context of a multi-modal recommendation task. From the table, we observe that over the four used datasets, 72.5% of the experimental instances (87 out of 120) tested with the recommendation models (VBPR, MMGCN, MMGCL, SLMRec) using CLIP/VLMo/BEiT-3 as encoders, significantly outperform the models using the original modality-specific encoders. This observation demonstrates the effectiveness of using the LMM encoders as feature extractors, which enables collaborative multi-modal feature generation and mitigates the issue of heterogeneity between visual and textual modalities. We now focus on comparing the LATTICE variants, a model that exhibits distinct trends among the five baselines. Indeed, we observe that LATTICE performs generally better than LATTICE$_{CLIP}$ and LATTICE$_{VLMo/BEiT-3}$ on all four used datasets. This contrasts with the other baseline models, which are generally improved by CLIP, VLMo and BEiT-3, and suggests a possible discrepancy between the LMM encoders and LATTICE in terms of feature alignment. Recall from Section 3.2 that LATTICE constructs item-item graphs based on the semantic similarities across different modalities. This objective is conceptually in conflict with that of CLIP/VLMo/BEiT-3, which generates deeply aligned features that could lead to a denser item-item graph in LATTICE. Consequently, the features extracted by LMM encoders may result in inadequate item-item graphs, leading to a decline in performance. On the other hand, as observed in Table 5.7, the recommendation models employing CLIP (dual-stream) as an encoder and those using VLMo/BEiT-3 (unified) as an encoder exhibit similar performances for all four datasets. To determine the best choice of the training setting for an effective deep modality alignment, we conduct further experiments in the remainder of the section, exploring both fine-tuning and end-to-end training paradigms for LMM encoders.

Overall, to answer RQ5.3, we conducted a large-scale empirical evaluation of the pre-training setup. Our exploration addressed five dimensions of multi-modal recommendation and their combinations: recommendation models, multi-modal extractors, training paradigms, datasets, and metrics. From the experiments, we conclude that the pre-trained LMM encoders are more effective at extracting visual and textual features from raw images and texts, especially when compared to methods using CNN and Sentence-Transformer.

5.2.3.6 The Effect of Fine-tuning Large Multi-modal Encoders (RQ5.4)

In Section 5.2.3.5, we have successfully examined the effectiveness of deep alignment using pre-trained LMM encoders (CLIP, VLMo and BEiT-3) in multi-modal recommendation models. Instead, in this section, we assess the effectiveness of fine-tuning CLIP, VLMo and BEiT-3 in improving the recommendation performance compared to their pre-trained configurations. Table 5.8 presents the performance changes observed in the used recommendation models after fine-tuning CLIP, VLMo and BEiT-3 with item images and descriptions from the used datasets. Table 5.8 shows that, in 70.0% of the experimental instances (84 out of 120) across all four datasets, the fine-tuned LMM encoders significantly improve the recommendation performance compared to their pre-trained configurations. This observation suggests that the LMM encoders become better adapted to the recommendation domain when they are fine-tuned with the recommendation datasets, thereby facilitating improved multi-modal representation learning with deeper aligned visual and textual embeddings. Consequently, the fine-tuned CLIP, VLMo and BEiT-3 encoders produce enhanced, well-aligned item embeddings, resulting in more accurate and contextually relevant embeddings for multi-modal recommendation models. However, we observe that there is no performance improvement for MMGCL$_{CLIP-FT}$, MMGCL$_{VLMo-FT}$ and MMGCL$_{BEiT-3-FT}$ compared to their respective pre-trained variants on the Amazon Clothing dataset. One potential reason for the observed decrease in performance could be that the fine-tuning process for this MMGCL model has led to overfitting the training data, causing a decrease in performance on the validation and test data. This overfitting could occur if the MMGCL model becomes too specialised in capturing the patterns in the training data, resulting in a decreased ability to generalise to unseen validation and test data. In line with the observations from Section 5.2.3.5, we find that the LATTICE variants using the fine-tuned CLIP/VLMo/BEiT-3 encoders continue to underperform when compared to the ones using the pre-trained CLIP/VLMo/BEiT-3 on all four datasets. This confirms our assumption that this is caused by the conceptual conflict between LATTICE, which aims to construct item-item graphs based on semantic similarities across different modalities, and the fine-tuned CLIP/VLMO/BEiT-3. Since the fine-tuned CLIP/VLMO/BEiT-3 encoders generate more closely aligned multi-modal features than the pre-trained ones, this conflict becomes more pronounced, potentially affecting the performance of LATTICE.

Overall, in response to RQ5.4, we conclude that fine-tuning the LMM encoders with raw images and texts from the used recommendation datasets generally enhances the multi-modal recommendation performance compared to the use of pre-trained configurations in most of the recommendation models used. However, exceptions may arise for certain models with inherent conceptual conflicts between the fine-tuned encoders and their inherent objectives.

Table 5.8: Experimental results comparing the use of pre-trained multi-modal encoders with the use of fine-tuned multi-modal encoders in multi-modal recommenders. The best performance of each model is highlighted in bold. * denotes a statistically significant difference between each recommendation model's pre-trained LMM variant and its fine-tuned variant using the paired t-test with $p < 0.05$. PT/FT are the abbreviations for Pre-Training and Fine-Tuning, respectively.

Dataset	Amazon Sports		Amazon Clothing		Amazon Baby		Bilibili	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
VBPR$_{VLMo-PT}$	0.0848	0.0373	0.0577	0.0266	0.0774	0.0346	0.0930	0.0536
VBPR$_{VLMo-FT}$	**0.0877***	**0.0395***	**0.0614***	**0.0280***	**0.0804***	**0.0356***	**0.0938**	**0.0547***
VBPR$_{BEiT-3-PT}$	0.0854	0.0382	0.0613	0.0280	0.0784	0.0355	0.0942	0.0545
VBPR$_{BEiT-3-FT}$	**0.0860***	**0.0391***	**0.0657***	**0.0306***	**0.0791***	**0.0357***	**0.0961***	**0.0556***
VBPR$_{CLIP-PT}$	0.0802	0.0357	0.0675	0.0296	0.0762	0.0336	0.0920	0.0527
VBPR$_{CLIP-FT}$	**0.0818***	**0.0364***	**0.0715***	**0.0313***	**0.0790***	**0.0344***	**0.0941***	**0.0538***
MMGCN$_{VLMo-PT}$	0.0484	0.0208	0.0252	0.0104	0.0538	0.0224	0.0790	0.0444
MMGCN$_{VLMo-FT}$	**0.0511***	**0.0218***	**0.0262***	**0.0111***	**0.0539**	**0.0230***	**0.0821***	**0.0488***
MMGCN$_{BEiT-3-PT}$	0.0488	0.0213	0.0250	0.0106	0.0564	0.0240	0.0824	0.0485
MMGCN$_{BEiT-3-FT}$	**0.0503***	**0.0225***	**0.0260***	**0.0111***	**0.0587***	**0.0254***	**0.0852***	**0.0507***
MMGCN$_{CLIP-PT}$	0.0494	0.0216	0.0253	0.0101	**0.0606**	**0.0257**	0.0835	0.0501
MMGCN$_{CLIP-FT}$	**0.0515***	**0.0229***	**0.0263***	**0.0114***	0.0598*	**0.0257**	**0.0867***	**0.0516***
MMGCL$_{VLMo-PT}$	0.0941	0.0446	0.0648	**0.0299**	0.0822	0.0370	0.1233	0.0624
MMGCL$_{VLMo-FT}$	**0.0980***	**0.0452***	**0.0653***	0.0294	**0.0825***	**0.0383***	**0.1254**	**0.0638***
MMGCL$_{BEiT-3-PT}$	0.0953	0.0453	0.0663	0.0307	0.0836	0.0384	0.1252	0.0632
MMGCL$_{BEiT-3-FT}$	**0.0970***	**0.0478***	**0.0680***	**0.0313***	**0.0846***	**0.0396***	**0.1266***	**0.0646***
MMGCL$_{CLIP-PT}$	0.0959	0.0447	**0.0703**	**0.0321**	0.0804	0.0362	0.1260	0.0652
MMGCL$_{CLIP-FT}$	**0.1007***	**0.0466***	0.0686*	0.0310*	**0.0827***	**0.0381***	**0.1276***	**0.0668***
SLMRec$_{VLMo-PT}$	0.0919	0.0426	0.0653	0.0304	0.0823	0.0374	0.1232	0.0624
SLMRec$_{VLMo-FT}$	**0.0952***	**0.0447***	**0.0655**	**0.0312***	**0.0836***	**0.0393***	**0.1255**	**0.0635**
SLMRec$_{BEiT-3-PT}$	0.0917	0.0430	0.0661	0.0315	0.0828	0.0375	0.1250	0.0646
SLMRec$_{BEiT-3-FT}$	**0.0938***	**0.0449***	**0.0675***	**0.0323***	**0.0836***	**0.0391***	**0.1282***	**0.0674***
SLMRec$_{CLIP-PT}$	0.0909	0.0427	0.0671	0.0304	0.0821	0.0381	0.1243	0.0637
SLMRec$_{CLIP-FT}$	**0.0936***	**0.0441***	**0.0677***	**0.0311***	**0.0835***	**0.0395***	**0.1260***	**0.0644**
LATTICE$_{VLMo-PT}$	**0.0758**	**0.0312**	**0.0512**	**0.0209**	**0.0817**	**0.0348**	**0.1042**	**0.0567**
LATTICE$_{VLMo-FT}$	0.0749*	0.0307	0.0501*	0.0207	0.0808*	0.0346	0.1027	0.0538*
LATTICE$_{BEiT-3-PT}$	**0.0807**	**0.0364**	**0.0578**	**0.0235**	**0.0822**	**0.0345**	**0.1066**	**0.0581**
LATTICE$_{BEiT-3-FT}$	0.0778*	0.0333*	0.0561*	0.0227	0.0796*	0.0324*	0.1025*	0.0534*
LATTICE$_{CLIP-PT}$	**0.0785**	**0.0332**	**0.0593**	**0.0263**	**0.0830**	**0.0354**	**0.1034**	**0.0561**
LATTICE$_{CLIP-FT}$	0.0781	0.0332	0.0578*	0.0248*	0.0812*	0.0348	0.1018	0.0529*

5.2.3.7 Training Paradigm: Two-stage vs. End-to-end (RQ5.5)

Another aspect not previously explored in the literature is the evaluation of the impact of adopting an end-to-end training paradigm when integrating the LLM encoders into the recommendation models. We conduct experiments to fill this gap and to gauge the benefits of an end-to-end training approach when integrating CLIP, VLMo and BEiT-3 into the recommendation models. In these experiments, we aim to determine (1) which is the most effective training paradigm, and (2) whether the commonly used losses in the recommendation task further enhance the modality alignment in a multi-modal recommendation system. Table 5.9 presents a detailed comparison of the results between the two-stage training and end-to-end training approaches. The end-to-end CLIP variants exhibit an enhanced performance in 90% of the experimental instances (36 out of

Table 5.9: Experimental results comparing the two-stage training with the end-to-end training when incorporating the LMM encoders in multi-modal recommenders. The best performance of each model is highlighted in bold. * denotes a statistically significant difference between each recommendation model's fine-tuned LMM variant and its end-to-end variant using the paired t-test with $p < 0.05$. FT/ETE denotes Fine-Tuning and End-To-End training, respectively.

Dataset	Amazon Sports		Amazon Clothing		Amazon Baby		Bilibili	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
VBPR$_{VLMo-FT}$	**0.0877**	**0.0395**	**0.0614**	**0.0280**	**0.0804**	**0.0356**	**0.0938**	**0.0547**
VBPR$_{VLMo-ETE}$	0.0706*	0.0329*	0.0445*	0.0199*	0.0799	0.0346	0.0920	0.0531*
VBPR$_{BEiT-3-FT}$	0.0860	0.0391	**0.0657**	**0.0306**	0.0791	0.0357	0.0961	0.0556
VBPR$_{BEiT-3-ETE}$	**0.0874***	**0.0404***	0.0631*	0.0287*	**0.0811***	**0.0370***	**0.0983***	**0.0604***
VBPR$_{CLIP-FT}$	0.0818	0.0364	**0.0715**	**0.0313**	**0.0790**	**0.0344**	0.0941	0.0538
VBPR$_{CLIP-ETE}$	**0.0892***	**0.0395***	0.0674*	0.0295*	0.0766	0.0340	**0.1174***	**0.0644***
MMGCN$_{VLMo-FT}$	0.0511	0.0218	0.0262	0.0111	0.0539	0.0230	0.0821	0.0488
MMGCN$_{VLMo-ETE}$	**0.0522***	**0.0221**	**0.0306***	**0.0127***	**0.0574***	**0.0245***	**0.0850***	**0.0511***
MMGCN$_{BEiT-3-FT}$	0.0503	0.0225	0.0260	0.0111	0.0587	0.0254	0.0852	0.0507
MMGCN$_{BEiT-3-ETE}$	**0.0575***	**0.0237***	**0.0346***	**0.0150***	**0.0613***	**0.0275***	**0.0876***	**0.0532***
MMGCN$_{CLIP-FT}$	0.0515	0.0229	0.0263	0.0114	0.0598	0.0257	0.0867	0.0516
MMGCN$_{CLIP-ETE}$	**0.0592***	**0.0253***	**0.0323***	**0.0134***	**0.0621***	**0.0267***	**0.0889***	**0.0538***
MMGCL$_{VLMo-FT}$	**0.0980**	**0.0452**	**0.0653**	**0.0294**	**0.0825**	**0.0383**	0.1254	0.0638
MMGCL$_{VLMo-ETE}$	0.0951*	0.0435*	0.0634*	0.0279*	0.0815*	0.0365*	**0.1283***	**0.0663***
MMGCL$_{BEiT-3-FT}$	0.0970	0.0464	0.0680	0.0313	0.0846	0.0396	0.1266	0.0646
MMGCL$_{BEiT-3-ETE}$	**0.1063***	**0.0486***	**0.0922***	**0.0412***	**0.0930***	**0.0414***	**0.1301***	**0.0685***
MMGCL$_{CLIP-FT}$	0.1007	0.0466	0.0686	0.0310	0.0827	0.0381	0.1276	0.0668
MMGCL$_{CLIP-ETE}$	**0.1065***	**0.0467**	**0.0855***	**0.0378***	**0.0876***	**0.0388***	**0.1303***	**0.0696***
SLMRec$_{VLMo-FT}$	0.0952	0.0447	0.0655	**0.0312**	**0.0836**	**0.0393**	0.1255	0.0635
SLMRec$_{VLMo-ETE}$	**0.0994***	**0.0461***	**0.0681***	0.0306*	0.0790*	0.0363*	**0.1268***	**0.0651***
SLMRec$_{BEiT-3-FT}$	0.0938	0.0449	0.0675	0.0323	0.0836	0.0391	0.1282	0.0674
SLMRec$_{BEiT-3-ETE}$	**0.1023***	**0.0476***	**0.0839***	**0.0371***	**0.0894***	**0.0396**	**0.1297***	**0.0681***
SLMRec$_{CLIP-FT}$	0.0936	0.0441	0.0677	0.0311	0.0835	0.0395	0.1260	0.0644
SLMRec$_{CLIP-ETE}$	**0.1058***	**0.0448***	**0.0859***	**0.0382***	**0.0889***	**0.0387***	**0.1316***	**0.0703***
LATTICE$_{VLMo-FT}$	0.0749	0.0307	0.0501	0.0207	0.0808	0.0346	0.1027	0.0538
LATTICE$_{VLMo-ETE}$	**0.1012***	**0.0451***	**0.0788***	**0.0361***	**0.0889***	**0.0393***	**0.1207***	**0.0640***
LATTICE$_{BEiT-3-FT}$	0.0778	0.0333	0.0561	0.0227	0.0796	0.0324	0.1025	0.0534
LATTICE$_{BEiT-3-ETE}$	**0.1010***	**0.0472***	**0.0835***	**0.0397***	**0.0910***	**0.0406***	**0.1289***	**0.0667***
LATTICE$_{CLIP-FT}$	0.0781	0.0332	0.0578	0.0248	0.0812	0.0348	0.1018	0.0529
LATTICE$_{CLIP-ETE}$	**0.1015***	**0.0451***	**0.0789***	**0.0361***	**0.0892***	**0.0393***	**0.1252***	**0.0658***

40 instances) across all four datasets compared to their respective fine-tuned configurations, with 97.2% of these instances showing significant improvements. In contrast, the end-to-end VLMo variants do not show similar improvements and even lead to a decline in performance. On the other hand, the end-to-end BEiT-3 variants exhibit an enhanced performance in 92.5% of the experimental instances (37 out of 40) across all four datasets compared to their respective fine-tuned configurations. This observation suggests that the end-to-end training paradigm facilitates a seamless integration of the CLIP and BEiT-3 encoders into the existing recommendation models, whereas it does not produce the same level of compatibility for VLMo. The performance decline in the end-to-end VLMo integration might be attributed to its architecture. As discussed in Section 5.2.1, unlike BEiT-3, which has 40 transformer layers and 228 million parameters, VLMo has only 24 transformer layers and 167 million parameters. The performance differences

bewteen BEiT-3 and VLMo in Table 5.9 indicate the importance of deeper modalities alignment, achieved through additional layers, in effectively modelling the complex and inherent correlations between different modalities. In Section 5.2.3.9, we analyse the architectures of the LMM encoders, evaluating both these architectures' effectiveness and efficiency. Another interesting finding from Table 5.9 is that the LATTICE$_{VLMO/CLIP-ETE/BEiT-3}$ model overcomes the inferior performance exhibited by both the pre-trained and fine-tuned versions of the LATTICE model in Sections 5.2.3.5 and 5.2.3.6, outperforming LATTICE$_{VLMO/CLIP-FT/BEiT-3}$ in 100% of the experimental instances across the four datasets. This observation indicates that the end-to-end training paradigm addresses the conceptual conflict between LATTICE and the LMM encoders by effectively facilitating a deep alignment between item modalities adapted to the specific recommendation task, and guided by the recommendation loss. This process results in more semantically informative item-item graphs for LATTICE. This simultaneous learning process allows the multi-modal models to produce complementary representations, thereby minimising the potential conflicts between LATTICE's item-item graph learning and the multi-modal encoders' feature alignment. Hence, this integrated training strategy enables the LATTICE model to capitalise on the strengths of the multi-modal encoders, resulting in an improved performance for top-K multi-modal recommendation.

Overall, in answer to RQ5.5, we conclude that end-to-end training is more suitable for the multi-modal recommendation task when incorporating dual-stream LMM encoders (i.e., CLIP) into existing models, while unified LMM encoders (i.e., VLMo) do not exhibit the same benefits. Furthermore, this training paradigm effectively addresses the conceptual conflict between the LMM encoders and existing models that explicitly facilitate multi-modal feature alignment. Consequently, the end-to-end training strategy leads to further enhancements in aligning the modalities within the multi-modal recommendation models, particularly when these models (e.g., LATTICE) are not compatible with the fine-tuned LMM encoders.

5.2.3.8 Modality Contribution Analysis (RQ5.6)

As detailed in Figure 5.4, the existing multi-modal recommendation models do not adequately explore the interdependencies between modalities due to the use of modality-specific encoders. In particular, at the start of Section 5.2, we have already discussed how the use of the modality-specific encoders results in a suboptimal and shallow alignment between modalities on three commonly used Amazon datasets. In this section, we investigate whether incorporating the LMM encoders into the existing multi-modal recommendation models improves the contribution of each modality to the overall recommendation performance in comparison to the use of existing modality-specific encoders. We report the recommendation performance with NDCG@20, as the observations with Recall@20 show similar trends. For conciseness, we also focus on the two best recommendation models, namely MMGCL and LATTICE, and use CLIP since it is the strongest LMM encoder on the three Amazon datasets. However, we observe similar trends

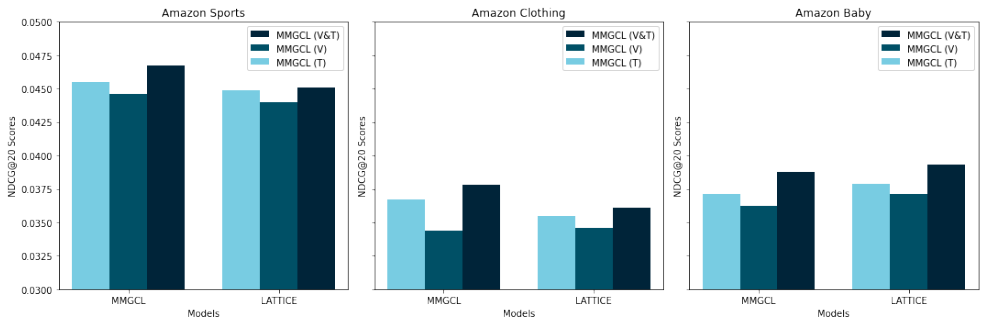


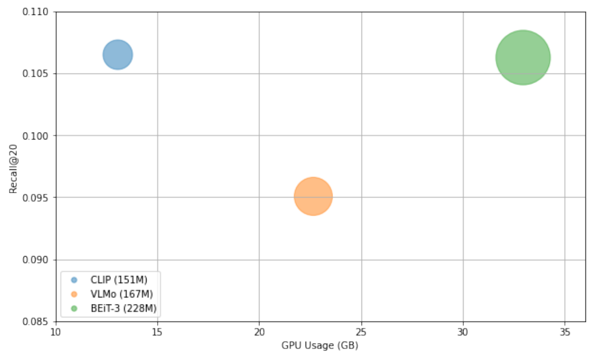
Figure 5.7: Comparison of the NDCG@20 scores of the MMGCL and LATTICE models after integrating the CLIP LMM encoder, with different modality inputs across the commonly used Amazon datasets. V/T are the abbreviations for Visual/Textual, respectively.

and conclusions with the other models and LMM encoders. Figure 5.7 presents the results of the MMGCL and LATTICE models when integrated with the LMM encoder (i.e., CLIP), either using single or multiple types of modalities as input. The comparison of the results from Figure 5.7 with those in Figure 5.4 allows us to assess whether the integration of an LMM encoder into existing multi-modal recommendation models enhances the effectiveness of the combined modalities (i.e., textual and visual) in contributing to the recommendation performance. This comparison aims to determine if deep modality alignment prevents a single modality (i.e., textual or visual) from dominating the recommendation performance, as previously observed in Figure 5.4. Indeed, Figure 5.4 shows that when using modality-specific encoders, the textual embeddings (T) lead to a better recommendation performance than when using the combined textual and visual embeddings (V&T). In contrast, Figure 5.7 shows that after integrating the LMM encoders, the combined embeddings (V&T) in the models (MMGCL, LATTICE) outperform each single modality (V or T). This result suggests that the SSL-trained LMM encoders (c.f. Section 5.2.2.1) can successfully exploit the deep modality alignment, indicating that the inclusion of additional modalities in a model should indeed intrinsically augment its knowledge base, thereby enhancing its performance in the top-K multi-modal recommendation task.

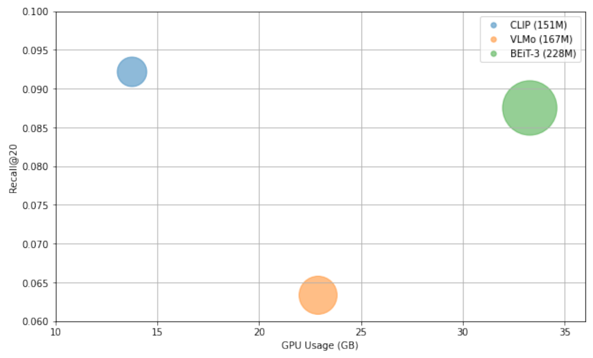
In response to RQ5.6, we conclude that the SSL-trained LMM encoders (c.f. Section 5.2.2.1) do enhance the contribution of each modality by achieving a deeper alignment across diverse modalities more effectively than recommendation models that use modality-specific encoders. The observed improvement is independent of the recommendation model used in the multi-modal recommendation task.

5.2.3.9 Efficiency vs. Effectiveness (RQ5.7)

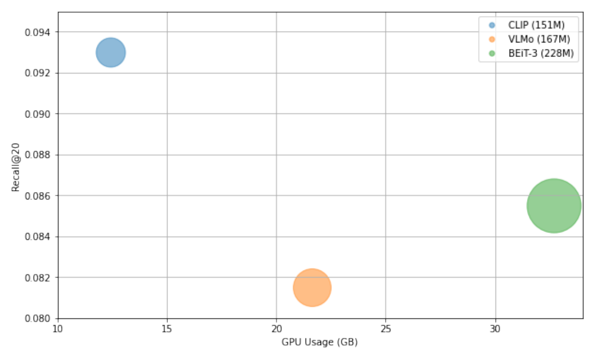
In this section, we evaluate the LLM encoders' efficiency, particularly in terms of GPU usage, training time and inference time across the used datasets with their corresponding effectiveness. This analysis allows us to compare the encoders' efficiency and effectiveness in the multi-modal



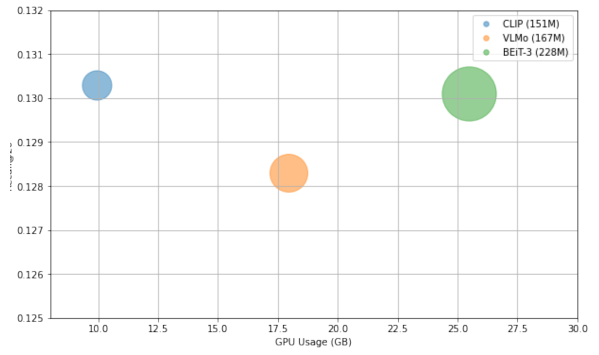
Amazon Sports Dataset



Amazon Clothing Dataset



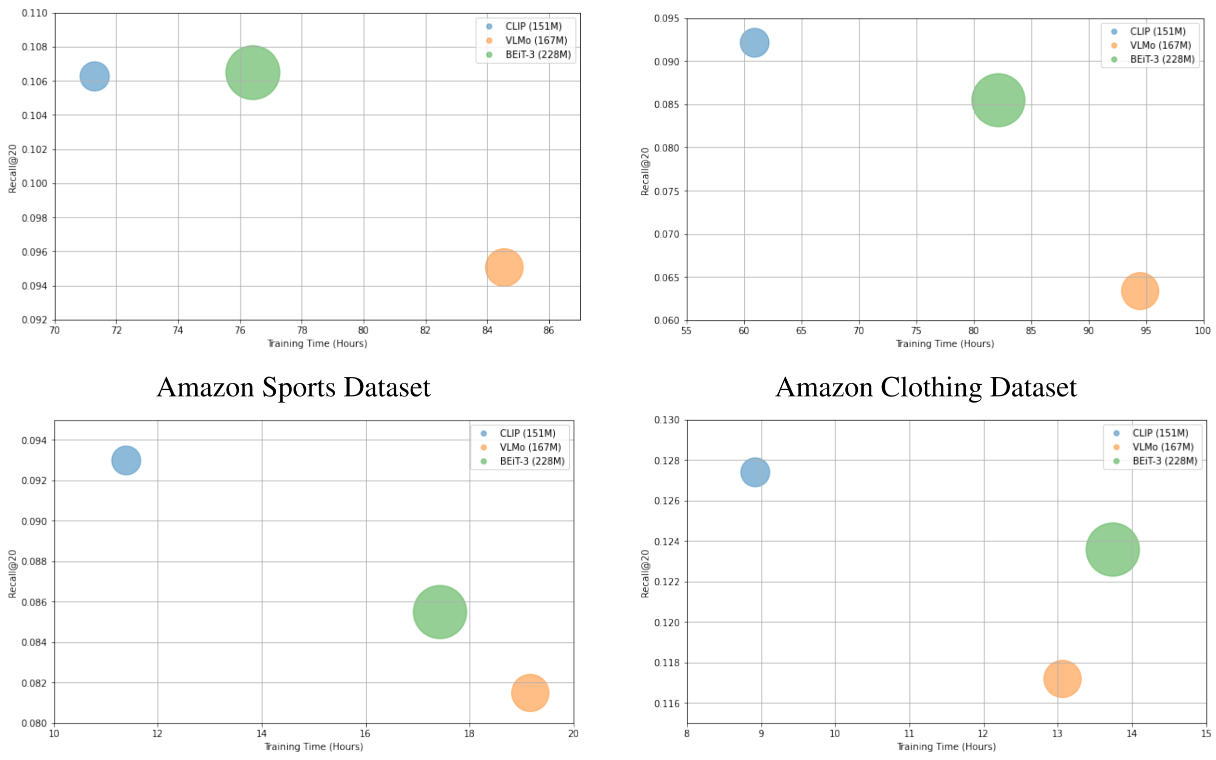
Amazon Baby Dataset



Bilibili Dataset

Figure 5.8: Effectiveness vs. GPU usage of the MMGCL model with the LMM encoders integration.

recommendation task. We first report the GPU consumption of the LMM encoders, including those with dual-stream (i.e., CLIP) or unified (i.e., VLMo and BEiT-3) architectures, and analyse their performance in the multi-modal recommendation task. In Section 5.2.3.1, we highlighted the number of parameters of the used LMM encoders. Our aim is to determine the deep alignment architecture that is the most effective in terms of both recommendation performance and GPU usage. Figure 5.8 plots the recommendation effectiveness against the GPU usage of the three LMM encoders (CLIP, VLMo and BEiT-3) with their corresponding number of model parameters on all four datasets (Amazon Sports, Amazon Clothing, Amazon Baby, Bilibili). For conciseness, Figure 5.8 reports the results of integrating all three LMM encoders into the MMGCL model on all four used datasets. However, the trends and conclusions on the other baseline models (i.e., VBPR, MMGCN, SLMRec, and LATTICE) are similar. The size of the LMM encoders' dots in the figures is proportional to their number of of parameters. From Figure 5.8, we observe that among the unified encoders, BEiT-3 outperforms VLMo in terms of effectiveness but requires approximately 30% more memory. This observation suggests that a higher number of parameters contributes to a better recommendation performance for a unified LMM encoder. On the other hand, when comparing CLIP and BEiT-3, we observe that CLIP consumes less GPU memory than BEiT-3 while maintaining a comparable effectiveness. This suggests that, to-date, the dual-stream LMM encoder, CLIP, provides a better architecture for



Amazon Sports Dataset

Amazon Clothing Dataset

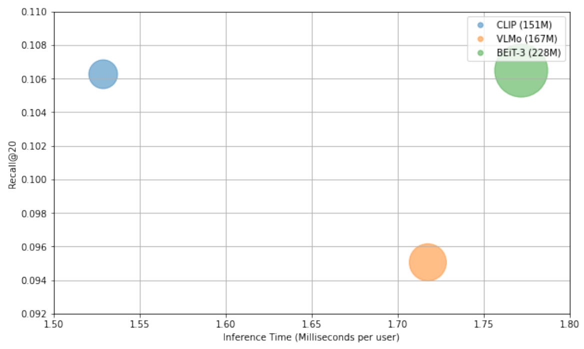
Amazon Baby Dataset

Bilibili Dataset

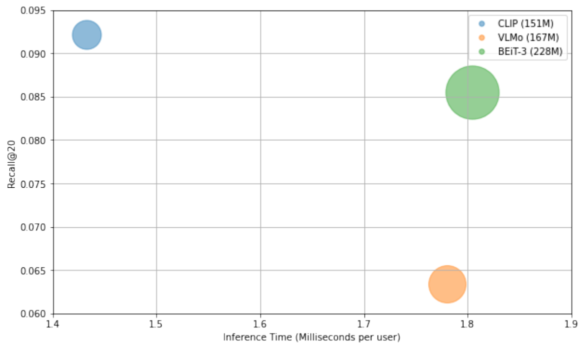
Figure 5.9: Effectiveness vs. training time of the MMGCL model with the LMM encoders integration.

the multi-modal recommendation task, in terms of balancing effectiveness and GPU usage.

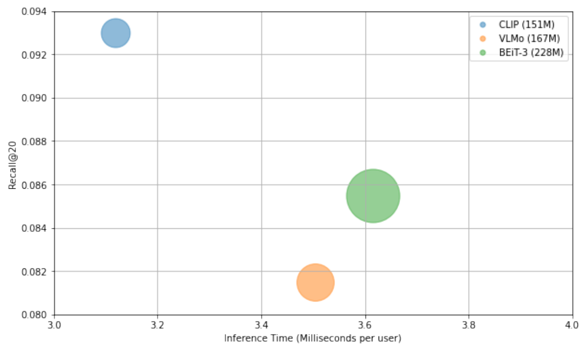
Next, we analyse the training and inference times of the LMM encoders (CLIP, VLMo and BEiT-3). We aim to determine which architecture is more time-efficient on all four used datasets. Specifically, we calculate the total training time by multiplying the average training time per epoch by the number of epochs required for convergence. We report this training time in hours. For calculating the inference time, we base our computation on the inference time per LMM encoder and the total user count across all used datasets. The inference times are reported in miliseconds. Similar to Fig 5.8, Figure 5.9 and Figure 5.10 report the training and inference times of the MMGCL model (as proposed in Section 5.1) on all four used datasets, since we also observe the same conclusions using the other baseline models. From Figure 5.9, we observe that CLIP needs less training time when compared to VLMo and BeiT-3 on all used datasets. For instance, CLIP shows a reduction in training time of 7.8% and 15.5% when compared to BEiT-3 and VLMo on the Amazon Sports dataset, respectively, while maintaining a comparable effectiveness. Figure 5.10 also shows that CLIP reduces the inference time when compared to BEiT-3 and VLMo across all used datasets. For example, CLIP is 14.3% and 15.7% faster than BEiT-3 and VLMo, respectively, on the Amazon Sports dataset, while achieving a comparable recommendation effectiveness. These aforementioned consistent observations do suggest that the dual-steam architecture, CLIP, with fewer model parameters is indeed more time-efficient and effective than the unified LMM encoders (i.e., VLMo and BEiT-3) on all used datasets.



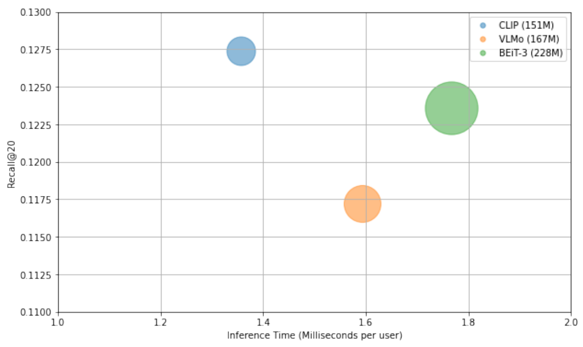
Amazon Sports Dataset



Amazon Clothing Dataset



Amazon Baby Dataset



Bilibili Dataset

Figure 5.10: Effectiveness vs. inference time of the MMGCL model with the LMM encoders integration.

In response to RQ5.7, we conclude that a higher number of model parameters does always improve the recommendation performance of a unified LMM encoder on all the used datasets. Moreover, the dual-stream LMM encoder, CLIP, is superior to the unified LMM encoders in terms of achieving both recommendation effectiveness and efficiency. Our results align with recent studies on a different task, namely multi-modal information retrieval, further supporting the superior performance of dual-stream architectures in comparison to unified architectures (Liu et al., 2024d; Wang et al., 2024).

5.2.4 Discussion

The second part of this chapter (Section 5.2) demonstrated a further step in enhancing modality fusion within an SSL paradigm. Moving beyond MMGCL's focus on modality-specific graph augmentations, we investigated the complementary dimension of deep modality alignment at the feature extraction stage using LMM encoders. This provides additional empirical support for our thesis that leveraging richer supervision signals from multiple modalities can improve top-K recommendation performance.

5.3 Conclusions

In this chapter, we investigated two complementary modality fusion methods for effectively mining self-supervised signals for top-K recommendation to align with our thesis statement in Section 1.2. We first proposed the MMGCL model, incorporating two novel graph augmentations and a challenging negative sampling strategy to enable the model to capture correlations across modalities and ensure that each modality contributes effectively to the fusion. Our MMGCL significantly outperformed all baselines (according to the paired t-test with the Holm-Bonferroni correction for $p < 0.01$), as shown in Table 5.2. Furthermore, Figure 5.2 confirmed the positive impact of proposed augmentations and challenging negative samples.

While our previous improvements enhanced SSL learning through graph augmentations and challenging negative samples, existing multi-modal approaches still rely heavily on features extracted independently from each modality using pre-trained modality-specific encoders. In the second part of this chapter, we leveraged Large Multi-Modal (LMM) encoders as new feature extractors to extend MMGCL and other multi-modal graph-based approaches, enabling the capture of deeper cross-modal relationships for a more expressive and effective modality fusion in top-K multi-modal recommendation. Our experimental results on four widely-used datasets (see Table 5.5) demonstrated that both pre-trained and fine-tuned CLIP, VLMo, and BEiT-3 encoders effectively extract and align visual and textual features from raw images and texts, significantly enhancing the performance of existing state-of-the-art multi-modal recommendation models (see Tables 5.7 and 5.8). We also evaluated different training paradigms for the LMM encoders, with results in Table 5.9 showing that end-to-end training further improves cross-modal alignment, leading to superior recommendation performance compared to pre-trained and fine-tuned configurations. Finally, as shown in Figure 5.10, we compared the architectures of the LMM encoders and found that the dual-stream encoder, CLIP, offers the best overall balance between effectiveness and efficiency for the recommendation task.

Although the proposed graph augmentations and deep modality alignment techniques enhance modality fusion in graph-based recommender systems, as discussed in Section 3.2, there are still longstanding isolation problems – isolated feature extraction process and isolated modality encoding process – that impede the effective mining of self-supervised signals across multiple modalities and remain unresolved in multi-modal recommender systems. In the next chapter, we investigate how to overcome these longstanding isolation problems using an SSL-enhanced unified graph transformer architecture for improved top-K multi-modal recommendation.

Chapter 6

Unified Multi-modal Architecture

In Chapter 5, we proposed modal-specific graph augmentations and deep modality alignment techniques to enhance the modality fusion process within multi-modal graph-based recommender systems. These contributions supported our hypothesis that graph-based recommender systems can learn more effective user/item representations by better mining self-supervised signals across multiple modalities. While the proposed approaches in Chapter 5 improve one critical stage of the top-K multi-modal recommendation pipeline, they address only the modality fusion process in isolation. In practice, a multi-modal graph-based recommender system involves a broader workflow: from data ingestion and feature extraction, through multi-modal fusion, to representation learning and ranking. However, as discussed in Section 3.2, existing multi-modal graph-based recommender systems typically use isolated processes for both feature extraction and modality encoding (fusion). Such isolated processes can harm the recommendation performance. In particular, multi-modal feature extraction, which aims to derive meaningful patterns and information from a variety of data types, is fundamental to the initial stage of any multi-modal recommendation pipeline. Indeed, effectively performing such feature extraction is critical for ensuring high-quality recommendations (Liu et al., 2024b; Yi and Ounis, 2024; Zhou et al., 2023a). Moreover, as discussed in Section 3.2, adopting SSL at this stage can further enhance the feature extraction and fusion process by generating auxiliary supervision signals across modalities. This enables the model to mine more effective cross-modal patterns and correlations, thereby producing richer and more discriminative user/item embeddings for the top-K multi-modal recommendation task. On the other hand, as discussed in Section 3.2, multi-modal fusion is also crucial for creating user/item representations from various contextual data, including the items' images and descriptions. Such a fusion typically occurs after feature extraction. Figure 6.1 illustrates how multi-modal recommendation typically consists of a pipeline of workflows ranging from data ingestion and feature extraction, to multi-modal fusion and ranking. Within this pipeline, two important problems emerge, namely, the use of (i) an isolated feature extraction process (c.f. Isolation 1 in the figure) and (ii) an isolated modality encoding process (c.f. Isolation 2). The 'Isolation 1' problem refers to the challenges that arise from using *an*

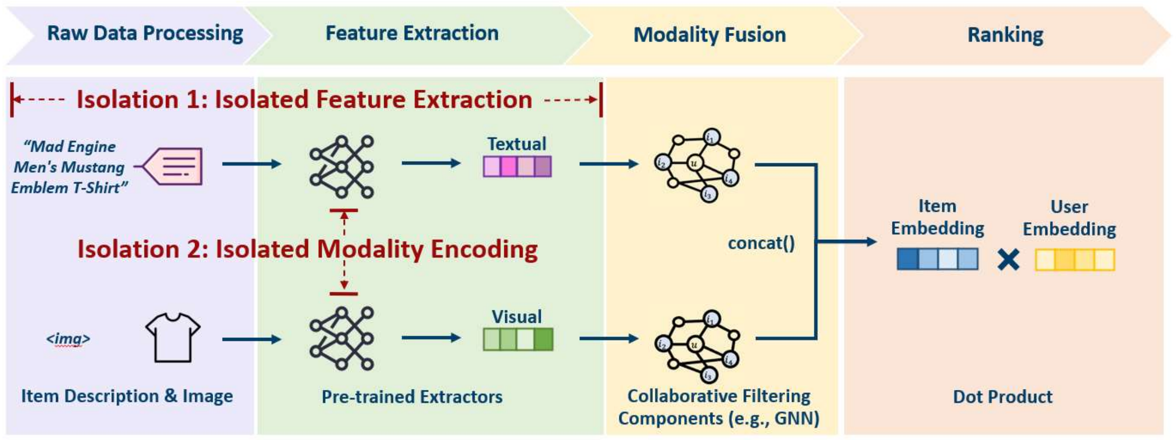


Figure 6.1: Recommendation working flow in existing multi-modal recommender systems.

isolated feature extraction process within each workflow of multi-modal recommender systems. Specifically, the pre-trained feature extractors operate independently alongside the subsequent modality fusion component. This isolated extraction process potentially leads to the incorporation of non-relevant information to the subsequent fusion component, thereby reducing the overall effectiveness in top-K multi-modal recommendation. On the other hand, as indicated by 'Isolation 2' in Figure 6.1, the existing recommender systems apply an individual component (e.g., a collaborative filtering component) to process each modality separately before finally fusing them using simple concatenations (Wei et al., 2019; Yi et al., 2022). This separation inherently prevents any interaction between modalities during the encoding process. Moreover, simple concatenations can introduce bias from the dominant modality (Liu et al., 2022a). As a result, such *an isolated modality encoding process* misses opportunities to jointly optimise the user/item embeddings resulting from the different modalities, thereby potentially reducing the overall recommendation effectiveness.

To address these isolated processes for both feature extraction (Isolation 1) and modality modelling (Isolation 2), we introduce a Unified Graph Transformer (UGT) model for top-K multi-modal recommendation. Our UGT model combines a multi-way transformer with a newly designed unified Graph Neural Network (GNN) in a novel cascading graph transformer architecture. We use the multi-way transformer as the extraction component and integrate its output into the unified GNN, which serves as the modality fusion component, in order to address the 'Isolation 1' problem. This new unified GNN within our proposed UGT architecture allows to address the 'Isolation 2' problem by uniformly aggregating the users/items' neighbour information and their corresponding multi-modal features, thereby enriching the multi-modal fusion for better user/item representations. The integration of the multi-way transformer and the unified GNN enables a simultaneous optimisation of the extraction of effective multi-modal features and their subsequent fusion, thereby solving the isolated feature extraction process. Our work is the first to use a multi-way transformer as an extraction component in multi-modal recommendation, thus going beyond its current use in the literature for encoding various modalities (e.g., images, text) within a shared transformer block (Bao et al., 2022) in Image-Text retrieval or Visual Question Answering (VQA) (Ge et al., 2024; Wang et al., 2023d). By leveraging pre-training on extensive web data corpora, a multi-way transformer obtains foundational knowledge about the relationships between images and texts (Yi et al., 2025). Instead, we apply a multi-way transformer in a recommendation setting to transform heterogeneous data into a unified latent space, thereby producing aligned embeddings from each modality. We then use our proposed unified GNN component to seamlessly fuse the resulting aligned multi-modal embeddings, representing a significant departure from existing models that process modalities separately. The remainder of this chapter is organised as follows:

- Section 6.1 presents the tackled top-K multi-modal recommendation task and the methodology of unifying the multi-modal recommendation pipeline using our UGT model;
- Section 6.2 describes the experimental setup and our research questions in this chapter;
- From Section 6.2.4 to Section 6.2.11, we analyse the results from the experiments to answer the research questions along with a qualitative analysis in Section 6.2.12;
- Section 6.3 summarises key conclusions of this chapter.

6.1 Unifying the Multi-modal Recommendation Pipeline

In this section, we describe how our UGT model unifies the multi-modal recommendation pipeline for top-K recommendation in a multi-modal setting. First, we introduce the architecture of our UGT model in Section 6.1.2, which is illustrated in Figure 6.2. Section 6.1.3 presents the unified multi-modal encoding process of our proposed multi-way transformer. Then, in Section 6.1.4, we introduce the unified GNN, which explicitly fuses multi-modal features and user-item interactions into final user/item representations. Finally, in Section 6.1.5, we describe the optimisation of UGT with its corresponding losses.

6.1.1 Task Definition

As introduced in Section 3.2, a top-K multi-modal recommendation task aims to effectively rank items for users based on their preferences across item modalities. Following the same notations defined in Section 5.1.1.1, let $\mathcal{U} = \{u\}$ denote the user set and $I = \{i\}$ denote the item set. The input ID embeddings of user u and item i are $\mathrm{E}_{id} \in \mathbb{R}^{d\times(|U|+|I|)}$. d is the dimension of the user/item embedding. Then, we denote each item modality feature as $\mathbf{E}_{i,m} \in \mathbb{R}^{d_m\times|I|}$, where d_m is the dimension of that modality's features, $m \in \mathcal{M}$ is a modality, and $\mathcal{M}$ is the set of modalities. In this chapter, we mainly consider visual v and textual t modalities because the datasets used only contain these two types of *raw data*, hence $\mathcal{M} = \{v, t\}$. The user's historical

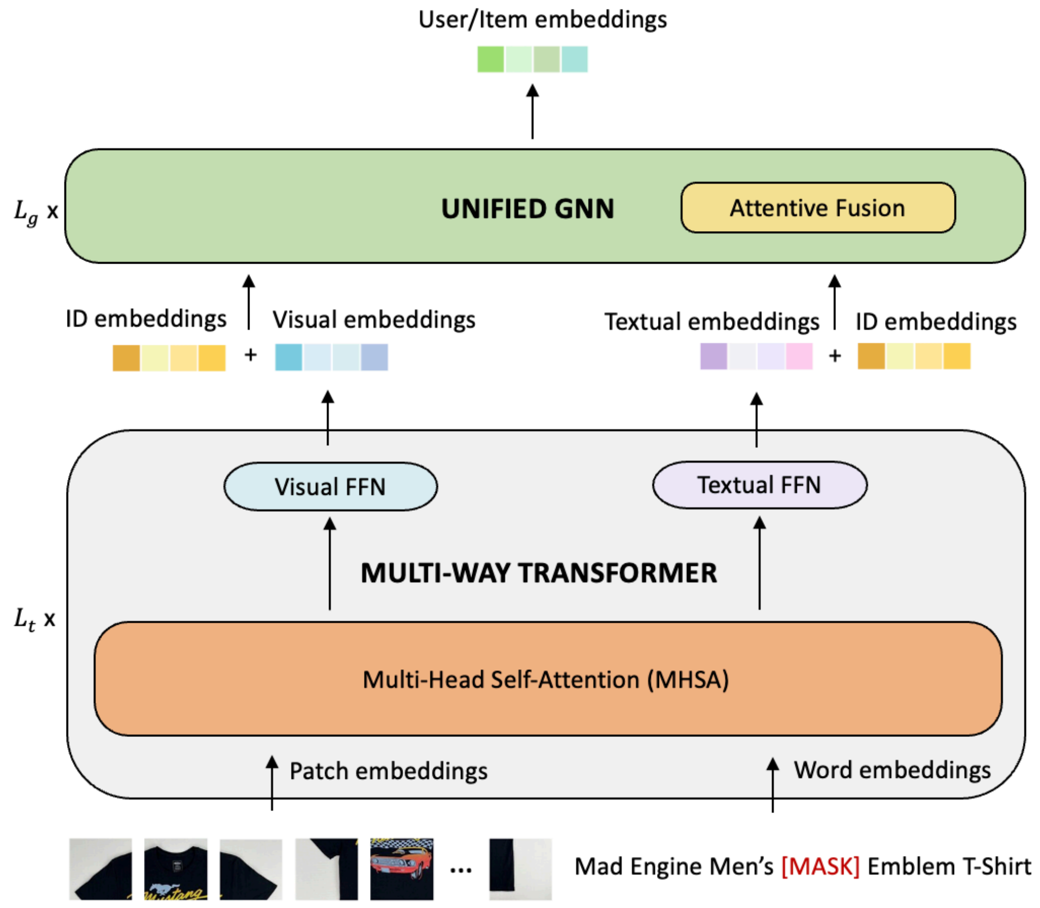


Figure 6.2: Our Unified multi-modal Graph Transformer (UGT).

behaviour data is denoted by $\mathcal{R} \in \{0,1\}^{|U|\times|I|}$, where each entry $\mathcal{R}_{u,i} = 1$ if user u clicked item i, otherwise $R_{u,i} = 0$. Following the same definition provided in Section 4.1.2.1, the historical interaction data $\mathcal{R}$ can be seen as a user-item bipartite graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{\mathcal{U} \cup I\}$ denotes the set of nodes and $\mathcal{E} = \{(u,i) \mid u \in \mathcal{U}, i \in I, R_{ui} = 1\}$ denotes the set of edges. The purpose of the top-K multi-modal recommendation model is to accurately predict the users' preferences by ranking items for each user according to the predicted preference scores $\hat{y}_{ui}$.

6.1.2 UGT Model Overview

Figure 6.2 shows the model architecture of UGT, which is composed of two main components: an extraction component (i.e., a multi-way transformer) and a modality fusion component (i.e., a unified GNN). Given an item image, an item text and an item ID, we follow the common practice in the computer vision domain (Kolesnikov et al., 2022) by splitting the item image into patches, flattening these patches, and linearly projecting them to form patch embeddings. Similarly, we obtain the word embeddings from an item's description (Radford et al., 2019). Our UGT model then uses the patch embeddings from the image and the word embeddings as inputs. In our UGT model, the multi-way transformer acts as a unified feature extractor, returning the extracted item's visual and textual embeddings before their combination with the user/item ID embeddings in a later new unified GNN component. We design this integration between the extraction and fusion components to address the isolated extraction problem (Isolation 1) in existing multi-modal recommender systems. As illustrated in Figure 6.2, once the extracted multi-modal item features have been obtained, our model's unified GNN component jointly combines these features with their ID embeddings to produce user/item embeddings based on the user-item interaction graph. The unified GNN is needed in order to obtain a smoother user/item embedding by aggregating the multi-modal features from the nodes' neighbours. This aggregation includes both the user-item interactions and the semantic information, which is derived from the multi-modal features processed by the multi-way transformer, into the final user/item representations. As shown in Figure 6.2, within the unified GNN, we incorporate an attentive-fusion component to fuse the jointly-learned and aggregated embeddings from different modalities, in order to enhance the multi-modal user/item representations. The improved joint multi-modal features in our UGT model are designed to overcome the limitations of modelling each modality separately (Isolation 2) in multi-modal recommendation.

6.1.3 Unified Multi-modal Encoding

As discussed in Section 6.1.2, we propose a multi-way transformer paired with a fusion component to alleviate the isolated extraction problem (Isolation 1) in existing multi-modal recommender systems. Specifically, we use this multi-way transformer, built upon the same backbone as VLMo and BEiT-3 (c.f. Section 5.2), to encode the raw images and textual descriptions of items, replacing the feed-forward network with modality experts in a standard transformer (Vaswani et al., 2017). Using the modality experts allows to capture modality-specific information, with each modality expert specifically handling different item modalities. It is important to note that our used multi-way transformer is pre-trained on a large corpus of web-scale multi-modal data, thereby enabling it to learn rich representations before being fine-tuned for the top-K multi-modal recommendation task (Bao et al., 2022). The multi-way transformer, combined with the fusion component shown in Figure 6.2, is designed to unify the extraction and fusion processes, addressing the problem of isolated extraction (Isolation 1).

6.1.4 Unified Multi-modal Fusion

In Section 3.2, we argued for the need for a unified approach to process the extracted multi-modal features for an effective fusion, instead of handling each modality individually. Hence, we introduce our new unified Graph Neural Network (GNN), which seamlessly fuses the extracted visual and textual embeddings in a joint manner, while being closely integrated with the multi-way transformer. The propagation function of our unified GNN is defined as follows:

$$x_i^{(l_g)} = (1+\epsilon) \cdot x_{i-vt}^{(l_g-1)} + \left(x_{u-vt}^{(l_g)} + x_{u-id}^{(l_g)}\right) \tag{6.1}$$

where x_i denotes the item representations in the l_g-th graph convolution layer; x_{i-vt} and x_{u-vt} denote the joint multi-modal embeddings of the item and its neighbours; x_{u-id} denotes the ID embeddings of the item's neighbours and ϵ is a scaling factor that controls the contribution of an item's self-connection with its associated joint multi-modal features. The user embeddings are similarly calculated. To obtain the user/item ID embeddings from the neighbours, we use LightGCN (c.f. Equation (3.1)) to propagate the ID embeddings of the users and items over the user-item interaction graph.

Another key addition to our UGT model is to propose an attentive-fusion method to enhance the joint embeddings based on the extracted multi-modal item features, as illustrated in Figure 6.2. Given the obtained visual and textual embeddings, h_{i-v} and h_{i-t}, derived from Equation (5.4), we combine the multi-modal embeddings with an attentive-fusion approach to obtain a unified joint embedding, as follows:

$$x_{i-vt}^{(l_g)} = \alpha h_{i-v}^{(l_g)} \| (1-\alpha) h_{i-t}^{(l_g)}, \tag{6.2}$$

where α is a learned parameter for the attentive-fusion and $\|$ represents the concatenation operation. As a consequence, this fusion approach is designed to uniformly enhance the resulting user/item embeddings by weighting the importance of each modality, in contrast to the simple concatenation operation used in previous works (Wei et al., 2023; Yi et al., 2022). As such, our UGT model uses the enhanced joint multi-modal features to address the problem of isolated modality encoding (Isolation 2) in top-K multi-modal recommendation.

6.1.5 Model Optimisation

To ensure the effective extraction and multi-modal fusion of the visual and textual modalities, our UGT model employs the Image-Text Contrastive (ITC) loss to train the model in a joint embedding space where the similarity between an item's image and its corresponding text description is maximised, while minimising the similarity between mismatched image-text pairs (Li et al., 2022). The ITC loss is defined as follows:

$$\mathcal{L}_{\text{ITC}} = -\frac{1}{N}\sum_{z=1}^{N} \log \frac{p_{v2t}(z)}{\sum_{z' \neq z}^{N} p_{v2t}(z')} - \frac{1}{N}\sum_{z=1}^{N} \log \frac{p_{t2v}(z)}{\sum_{z' \neq z}^{N} p_{t2v}(z')} \tag{6.3}$$

where N is the batch size, while $p_{v2t}(z)$ and $p_{t2v}(z)$ are the softmax-normalised image-to-text and text-to-image similarities of the z-th pair, respectively. As a result, this ITC loss acts as a self-supervised signal that guides our UGT model to learn effective extraction and fusion of multi-modal features from raw data.

We also implement the same multi-task training strategy as in Equation (6.4) of Section 5.1 in our UGT model that simultaneously optimises the BPR loss (see Equation (2.4)) along with the ITC loss $\mathcal{L}_{ITC}$:

$$\mathcal{L} = \mathcal{L}_{BPR} + \lambda_c \mathcal{L}_{ITC} + \lambda \|\Theta\|_2, \tag{6.4}$$

where λ_c and λ are hyper-parameters that control the strengths of the ITC loss and the L_2 regularisation, respectively, and Θ is the set of model parameters. Hence, we optimise both the multi-way transformer and the unified GNN using the loss functions above. These loss functions prevent the incorporation of non-relevant information that could be harmful to the learned user/item embeddings, further supporting our UGT model in addressing the isolated extraction problem (Isolation 1) in a unified manner. By applying Equation (5.4), we use the multi-way transformer to encode the raw features from each modality into multi-modal item embeddings. Next, we fuse the extracted visual and textual item embeddings using the attentive fusion as detailed in Equation (6.2). We then integrate the resulting joint item embeddings with the user/item ID embeddings and the user-item interactions within our unified GNN, as detailed in Equation (6.1), so as to obtain the final user/item embeddings. In summary, we use Equation (6.4) to optimise both the multi-way transformer and the unified GNN within our UGT model, thereby facilitating a unified approach to address the isolated extraction problem (Isolation 1). In addition, by incorporating an attentive-fusion component, the unified GNN uniformly fuses the multi-modal features, thereby tackling the problem of isolated modality encoding (Isolation 2) in top-K multi-modal recommendation.

6.2 Experiments

We conduct experiments to validate the effectiveness of our UGT model on three public datasets, in comparison to 13 strong baselines, including established and existing state-of-the-art recommender systems. To examine and analyse the effectiveness of our UGT model, we conduct experiments to answer the following 8 research questions:

- **RQ6.1**: How does our proposed UGT model perform compared with existing multi-modal recommender systems?
- **RQ6.2**: How do the two components of UGT, namely the unified GNN and the multi-way transformer as well as the UGT's loss function affect the performance of the model?
- **RQ6.3**: How do different parameters (i.e. ϵ, λ_c) in our UGT model affect its performance?
- **RQ6.4**: Can our UGT model demonstrate more efficient feature extraction and modality fusion processes compared to the best-performing baselines?

- **RQ6.5**: Does the UGT model exhibit a better alignment of modalities compared to the strongest identified baseline ?
- **RQ6.6**: Does the UGT model improve the contribution of each modality to the overall recommendation performance in comparison to the existing multi-modal recommenders?
- **RQ6.7**: Is our UGT model more effective compared to the best-performing multi-modal recommender system in a cold-start scenario?
- **RQ6.8**: Does the UGT model demonstrate a superior out-of-domain performance compared to the best-performing multi-modal recommender system?

In addition, we conduct a qualitative analysis in Section 6.2.12, illustrating how our UGT model leverages multi-modal knowledge from the item's images and descriptions to recommend items, in comparison with the most effective baseline identified from our experiments.

6.2.1 Datasets and Experimental Settings

In order to evaluate the effectiveness of our UGT model in the top-K multi-modal recommendation task, we conduct experiments on the same three Amazon Review datasets (Sports, Clothing, Baby) introduced in Section 5.2. In this chapter, we choose these datasets due to their comprehensive coverage of user-item interactions and their abundant multi-modal data, including the items' image URLs and textual descriptions, which enables an end-to-end evaluation from raw data. These datasets are also unique in providing extensive raw data, unlike other multi-modal recommendation datasets such as TikTok[1] and Kwai[2] introduced in Section 5.1, which do not provide raw data in relation to the various modalities. In addition, these Amazon Review datasets have been widely used to evaluate by the most existing state-of-the-art baselines, such as (Wei et al., 2023; Zhang Jinghao et al., 2021; Zhou et al., 2023b), further confirming their suitability for our analysis[3].

6.2.2 Evaluation Protocol

Following the same evaluation setting in Chapter 5, we randomly split the datasets into training, validation, and testing sets using an 8:1:1 ratio. Both our UGT model and the used baselines have their hyper-parameters optimised using a grid search on the validation set, and refined by the Adam (Kingma and Ba, 2014) optimiser. For the other hyper-parameters unique to our UGT model (ϵ and λ_c), we tune the parameters as follows: $\epsilon \in \{0, 0.1, 0.2, ..., 1.0\}$ and

[1] `http://ai-lab-challenge.bytedance.com/` [2] https://www.kuaishou.com/activity/uimc [3] With respect to scalability, we have successfully implemented our method on datasets as large as the Amazon Electronics dataset, which includes up to 192k users and 63k items. This confirms that our UGT model is applicable for large-scale recommendation tasks.

$\lambda_c \in \{0, 0.1, 0.2, ..., 1.0\}$. The used multi-way transformer in UGT is initialised from the publicly available *BEiT3-base-itc* checkpoint[4], as validated for its effectiveness in Section 5.2: 12 transformer layers, feed-forward expansion ratio of 4 and 12 self-attention heads. We use Recall@K and NDCG@K (c.f. Section 2.4) to examine the top-K recommendation performance in a multi-modal setting. We set K to 10 and 20 following (Zhang Jinghao et al., 2021), and report the average performance achieved for all users in the testing set. We apply an early-stopping strategy that terminates the training of both UGT and the baseline models if no decrease in validation loss is observed over 50 epochs. All used baselines and our UGT model are implemented with PyTorch and are trained on a GPU A6000 with 48GB memory. Our source code, raw data processing scripts, and related documentation are publicly available at: `https://github.com/terrierteam/UGT4MMRec/`.

6.2.3 Baselines

We evaluate the effectiveness of our UGT model on four top-k general recommender systems and nine multi-modal recommender systems, which use isolated extraction methods and isolated modality fusion methods. Specifically, the four general recommender systems include ItemKNN, NFCG, LightGCN and DGCF, where these general recommenders were introduced in Section 3.1. In addition, we use VBPR, MMGCN, MMGCL, SLMRec, LATTICE, BM3 and FREEDOM as multi-modal recommendation baselines (c.f. Section 3.2). To ensure a fair comparison, we employ CNN and Sentence-Transformer as modality-specific extractors in the used multi-modal baselines (Zhou et al., 2023a). These extractors are used to derive item features from the raw images and descriptions within the used datasets. In doing so, we ensure that both the CNN and Sentence-Transformer, as well as our multi-way transformer, have a similar number of parameters (170 million vs. 167 million), ensuring a fair comparison. Unlike the seven multi-modal approaches introduced in Section 3.2, we further compare UGT with two additional graph transformer models designed for top-K multi-modal recommendation:

- **PGMT (Liu et al., 2021c)** constructs an item-item graph to aggregate the multi-modal features of items and leverages a transformer network to model the item embeddings along with their contextual neighbours within the obtained graph;

- **LightGT (Wei et al., 2023)** uses a vanilla transformer network to model the users' preferences by considering their interacted items along with individually pre-extracted multi-modal features and integrates these preferences with the aggregated user/item ID embeddings to predict the user-item interactions.

In the following sections, we conduct an extensive evaluation of UGT. We compare UGT against all baselines (Section 6.2.4), analyse the contribution of each component within UGT (Section 6.2.5), and perform a comprehensive hyper-parameter study (Section 6.2.6). We further

[4] `https://github.com/microsoft/unilm/tree/master/beit3`

Table 6.1: Experimental results comparing our UGT model with the used baselines. The best performance of each model is highlighted in bold. * denotes a significant difference compared to the indicated baseline performance using the paired t-test with the Holm-Bonferroni correction for $p < 0.05$.

Dataset	Sports				Clothing				Baby			
Methods	Recall@10	Recall@20	NDCG@10	NDCG@20	Recall@10	Recall@20	NDCG@10	NDCG@20	Recall@10	Recall@20	NDCG@10	NDCG@20
ItemKNN	0.0372*	0.0558*	0.0225*	0.0306*	0.0294*	0.0441*	0.0103*	0.0122*	0.0363*	0.0588*	0.0163*	0.0230*
NGCF	0.0502*	0.0798*	0.0282*	0.0356*	0.0361*	0.0528*	0.0131*	0.0167*	0.0425*	0.0688*	0.0204*	0.0286*
LightGCN	0.0523*	0.0820*	0.0306*	0.0389*	0.0378*	0.0557*	0.0145*	0.0185*	0.0441*	0.0720*	0.0228*	0.0303*
DGCF	0.0515*	0.0806*	0.0290*	0.0365*	0.0366*	0.0532*	0.0136*	0.0175*	0.0416*	0.0677*	0.0201*	0.0290*
VBPR	0.0509*	0.0771*	0.0280*	0.0349*	0.0409*	0.0611*	0.0226*	0.0277*	0.0479*	0.0740*	0.0262*	0.0329*
MMGCN	0.0290*	0.0475*	0.0154*	0.0201*	0.0151*	0.0246*	0.0077*	0.0100*	0.0391*	0.0642*	0.0201*	0.0266*
MMGCL	0.0617*	0.0913*	0.0351*	0.0428*	0.0410*	0.0607*	0.0227*	0.0277*	0.0521*	0.0790*	0.0283*	0.0352*
SLMRec	0.0605*	0.0901*	0.0341*	0.0417*	0.0430*	0.0623*	0.0238*	0.0287*	0.0527*	0.0810*	0.0288*	0.0361*
LATTICE	0.0633*	0.0944*	0.0334*	0.0424*	0.0484*	0.0704*	0.0280*	0.0336*	0.0539*	0.0860*	0.0291*	0.0374*
BM3	0.0661*	0.0970*	0.0350*	0.0438*	0.0535*	0.0797*	0.0305*	0.0358*	0.0542*	0.0863*	0.0296*	0.0380*
FREEDOM	0.0605*	0.0918*	0.0352*	0.0449*	0.0538*	0.0809*	0.0290*	0.0356*	0.0551*	0.0874*	0.0303*	0.0385*
PMGT	0.0524*	0.0796*	0.0303*	0.0371*	0.0403*	0.0608*	0.0219*	0.0264*	0.0500*	0.0764*	0.0271*	0.0338*
LightGT	0.0652*	0.0953*	0.0347*	0.0440*	0.0514*	0.0755*	0.0299*	0.0353*	0.0544*	0.0867*	0.0294*	0.0381*
UGT	**0.0705**	**0.1034**	**0.0391**	**0.0477**	**0.0603**	**0.0922**	**0.0330**	**0.0402**	**0.0602**	**0.0930**	**0.0325**	**0.0406**
%Improv.	6.66%	6.60%	11.10%	6.24%	12.08%	13.97%	8.20%	12.29%	9.26%	6.41%	7.26%	5.45%

examine the model's efficiency (Section 6.2.8), investigate the alignment of intermediate visual and textual embeddings to gauge whether they indicate a better representation quality (Section 6.2.9), and assess the contribution of each modality to the overall top-K multi-modal recommendation performance (Section 6.2.10). In addition, we further investigate UGT's effectiveness with a cold-start analysis (Section 6.2.11) and an out-of-domain generalisation test (Section 6.2.12). Finally, we provide a qualitative study to understand how UGT leverages multi-modal knowledge to recommends items compared to the best-performing baseline.

6.2.4 Performance Comparison (RQ6.1)

Table 6.1 compares the performance of our UGT model to that of thirteen top-K multi-modal recommendation baselines, including both established methods and existing state-of-the-art multi-modal models. In the table, the top and second-best results are highlighted in bold and underlined, respectively. We also evaluate the statistical significance of the difference in performance between our UGT model and that of the used baselines according to the paired t-test with the Holm-Bonferroni correction (p-value < 0.05). From the results in Table 6.1, we have the following observations:

- For all three datasets, UGT outperforms all the baseline models on all metrics by a large margin, and the differences are statistically significant in all cases. In particular, our UGT model outperforms the strongest baseline, LightGT, in terms of Recall@20, showing improvements of 6.60%, 13.97% and 6.41% over the three used datasets, respectively. These significant improvements demonstrate the effectiveness of our graph transformer architecture in comparison to using an independent multi-modal recommender system alongside the corresponding pre-trained feature extractors. We attribute this improved performance to the combined effect of the multi-way transformer and the unified GNN, along with the used losses (i.e., BPR loss, ITC loss). These components enable a simultaneous optimisation for extracting effective multi-modal features, which are then fused to enhance the final user/item embeddings.

- Table 6.1 also shows that both the graph transformer models (PGMT, LightGT, UGT) and the GNN-based models (MMGCN, MMGCL, SLMRec, LATTICE, BM3, FREEDOM) outperform general established recommendation models (ItemKNN, NGCF, LightGCN, DGFCF) by large margins on all used datasets. This observation emphasises the usefulness of multi-modal features in enabling more accurate top-K multi-modal recommendations for target users.
- In Table 6.1, the comparison of the graph transformer models (PGMT, LightGT, UGT) with the GNN-based models (MMGCN, MMGCL, SLMRec, LATTICE, BM3, FREEDOM), shows that, with the exception of PGMT, the graph transformer models generally exhibit a performance on par with the GNN-based models. This result indicates that the application of the transformer network in the GNN-based models is more suitable for the extraction or distillation of multi-modal features, in comparison to using the transformer for aggregation. Similar observations have been made in other tasks such as VQA, as reported in (He and Wang, 2023).

Hence, in answer to RQ6.1, we conclude that our UGT model effectively leverages the graph transformer architecture, consisting of a multi-way transformer paired with a unified GNN, to significantly enhance the multi-modal recommendation performance in comparison to 13 strong baselines from the literature. UGT indeed offers a comprehensive end-to-end solution that effectively and seamlessly combines the extraction and fusion components, thereby addressing the issue of isolated feature extraction (Isolation 1) in multi-modal recommender systems.

6.2.5 Ablation Study (RQ6.2)

We now evaluate how each component of the UGT model, the multi-way transformer (referred to as Trans) and the unified GNN (UGNN), influences the recommendation performance. Recall that the unified GNN component incorporates an attentive-fusion (Attn-Fuse) as an internal component (see Figure 6.2). Specifically, to evaluate the impact of the multi-way transformer on recommendation effectiveness, we first replace it (Trans) with a CNN and Sentence Transformer extractor (Zhou et al., 2023a). Then, we replace the unified GNN (UGNN) component with LightGCN (He et al., 2020), which is commonly used in the baseline models. In addition, we simplify the attentive-fusion (Attn-Fuse) component by changing its attentive concatenation to a straightforward concatenation of the extracted multi-modal features so as to assess the effectiveness of the attentive-fusion component. Finally, we evaluate the effect of SSL-based contrastive learning (CL) within the UGT's loss function by removing the contrastive ITC loss (c.f. Equation (6.4)). Table 6.2 presents the performance outcomes of UGT's variants in multi-modal recommendation across the three datasets we used. We observe the following:

- Comparing UGT to its variant without the attentive concatenation (UGT w/o Attn-Fuse), we note a marked decrease in the effectiveness of UGT across all three datasets. This result indi-

Table 6.2: Ablation study on the key components of UGT. * indicates the significant differences according to the paired-t test.

Dataset	Sports				Clothing				Baby			
Methods	Recall@10	Recall@20	NDCG@10	NDCG@20	Recall@10	Recall@20	NDCG@10	NDCG@20	Recall@10	Recall@20	NDCG@10	NDCG@20
UGT w/o Attn-Fuse	0.0675*	0.0986*	0.0375*	0.0464	0.0564*	0.0823*	0.0310*	0.0373*	0.0568*	0.0894*	0.0308*	0.0404
UGT w/o UGNN	0.0667*	0.0980*	0.0371*	0.0452*	0.0540*	0.0811*	0.0302*	0.0356*	0.0551*	0.0865*	0.0298*	0.0383*
UGT w/o Trans	0.0628*	0.0929*	0.0357*	0.0440*	0.0483*	0.0784*	0.0269*	0.0323*	0.0535*	0.0834*	0.0290*	0.0364*
UGT w/o CL	0.0680*	0.0953*	0.0381*	0.0458*	0.0578*	0.0896*	0.0309*	0.0360*	0.0543*	0.0855*	0.0303*	0.0377*
UGT	**0.0705**	**0.1034**	**0.0391**	**0.0477**	**0.0603**	**0.0922**	**0.0330**	**0.0402**	**0.0602**	**0.0930**	**0.0325**	**0.0406**

cates the importance of our attentive-fusion component in effectively fusing the multi-modal features into the final user/item embeddings.

- The ablation of the unified GNN (UGNN) and its replacement with a LightGCN multi-stream processing on each modality[5] leads to a significant decrease of UGT's performance. This observation emphasises the essential role of applying a unified processing on the multi-modal inputs, in contrast to the traditional method that processes each modality input separately.
- Table 6.2 also shows that replacing our multi-way transformer (Trans) component with the CNN and the Sentence Transformer extractors markedly degrades the UGT's recommendation performance. This result highlights the advantage of replacing the pre-trained extractors, typically used in the existing methods, with our multi-way transformer, thereby enabling the simultaneous optimisation of the extraction process and the generation of the user/item embeddings for effective multi-modal recommendation.
- The UGT model, which uses both the ITC and BPR losses, significantly outperforms its variant without contrastive learning (i.e., UGT w/oCL). This suggests that the contrastive ITC loss, as introduced in Section 6.1.5, is beneficial for top-K multi-modal recommendation. Indeed, the use of contrastive ITC loss, simultaneously provides large gradients to optimise both the extraction (multi-way transformer) and fusion (UGNN) components, thereby improving the recommendation performance.

Hence, in answer to RQ6.2, we conclude that UGT successfully uses each of its key components as well as its loss function to provide an effective unified approach for learning effective user/item representations in top-K multi-modal recommendation. Specifically, our unified GNN (UGNN) component effectively fuses the multi-modal features using an attentive-fusion (Attn-Fuse) method, thereby addressing the problem of isolated modality encoding (Isolation 2) in the existing multi-modal recommenders. Furthermore, when we replace the CNN and the Sentence Transformer extractors with our multi-way transformer (Trans) component and pair it with the unified GNN, we observe an improvement in recommendation performance, thereby addressing the isolated feature extraction problem (Isolation 1). Finally, the contrastive ITC loss (CL) effectively optimises both the unified GNN (UGNN) and the multi-way transformer (Trans) components, thereby improving the recommendation performance.

[5] Indeed, most GNN-based models (MMGCL, SLMRec, LATTICE) apply LightGCNs in multi-stream processing.

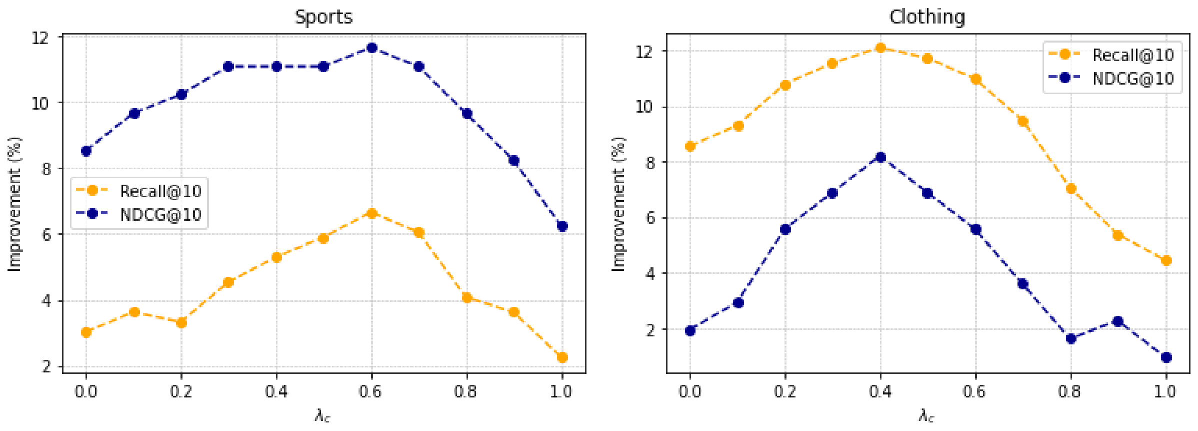


Figure 6.3: Performance of our UGT model with respect to different λ_c on the Sports and Clothing datasets.

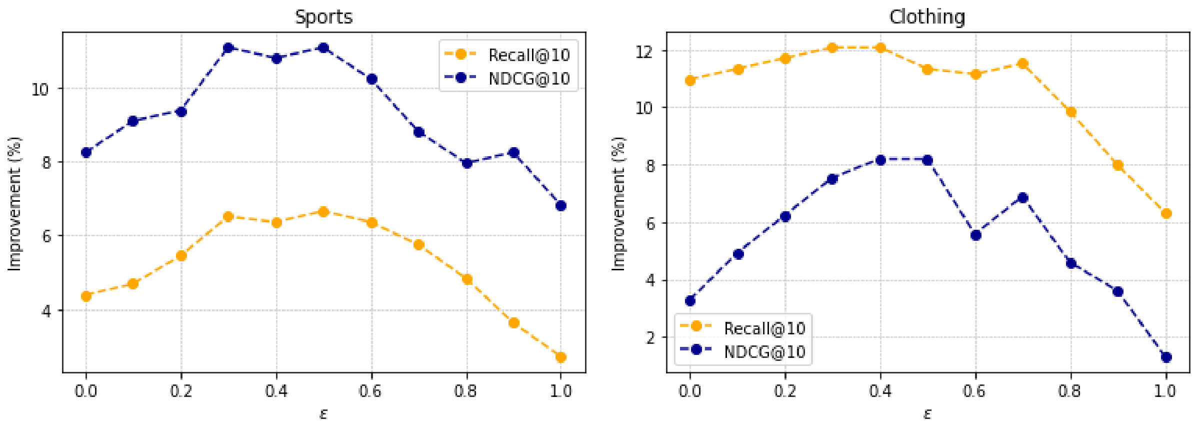


Figure 6.4: Performance of our UGT model using different ϵ values on the Sports and Clothing datasets.

6.2.6 Hyper-parameter Study (RQ6.3)

In this section, we investigate the sensitivity of our UGT model to the hyper-parameters. We focus on Recall@10 for reporting the recommendation performance, since we observe the same trends and conclusions with NDCG@10 across the used datasets. Figure 6.3 presents the effects of the hyper-parameters for the Sports and Clothing datasets, and we observe similar conclusions on the Baby dataset. We primarily analyse two important parameters in our UGT model, namely: (i) the scaling factor ϵ, which controls the influence of an item's intrinsic multi-modal features in the graph convolution operations; and (ii) the contrastive factor λ_c, regulating the strength of the Image-Text Contrastive (ITC) loss during training. Figure 6.3 and Figure 6.4 show the performance of UGT with different values of ϵ and λ_c, respectively.

6.2.6.1 Impact of the contrastive factor λ_c

The contrastive factor λ_c indicates the importance of the contrastively learned loss, balancing the contribution of the ITC loss within the joint training loss in Equation (6.4). From Figure 6.3, we observe that the best performance of our UGT model occurs at λ_c = 0.6, and at λ_c = 0.4 on the Sports and Clothing datasets, respectively, with a marked performance degradation at higher λ_c values. A high λ_c value indicates that UGT emphasises the item's image-text similarity as measured by the ITC loss, over the BPR pairwise ranking loss in the recommendation task. These results highlight the importance of carefully choosing λ_c to tailor the loss function to the task, thereby improving the model's performance.

6.2.6.2 Impact of the multi-modal scaling factor ϵ

To improve the multi-modal fusion within our unified GNN, we introduced a scaling factor, ϵ, which incorporates each item's own multi-modal features into the graph convolution operations, as detailed in Equation (6.1). To analyse the effect of incorporating an item's own multi-modal features in the graph convolution operations, we vary ϵ in the range {0.0, 0.1, ..., 0.9, 1.0} with a step size of 0.1, as previously described in Section 6.2.1. From Figure 6.4, we observe that our UGT model reaches its peak performance on the Sports and Clothing datasets when ϵ = 0.5 and ϵ = 0.4, respectively. This suggests that the best value ϵ differs slightly depending on the recommendation scenario. However, the performance drops markedly for a higher value of ϵ. These results indicate that the inclusion of an item's individual multi-modal features effectively enhances the user/item embeddings within a multi-modal recommendation system. On the other hand, excessively weighting the item's own features (indicated by a high value of ϵ) decreases the recommendation performance.

Hence, in answer to RQ6.3, we find that the performance of UGT is overall fairly sensitive to the changes in the ϵ and λ_c parameters. Indeed, while our UGT model is able to learn adequate values for different datasets, our parameter analysis shows that a careful selection of these parameters is in general needed for optimal results.

6.2.7 UGT Model Efficiency (RQ6.4)

In this section, we evaluate the efficiency of our UGT model in terms of computational complexity. In particular, we analyse the complexity of our UGT model, and compare it against the best-performing baselines (FREEDOM and LightGT). Table 6.3 presents a summary of the UGT model's computational complexity and effectiveness in comparison to the baselines. We begin by briefly introducing the notation used in our complexity analysis. Let $\mathcal{E}$ be the number of user–item edges (i.e., user–item interactions) in the bipartite graph $\mathcal{G}$, as defined in Section 6.1, while N_u and N_i denote the number of users and items, respectively. Let L_g represent the number of graph propagation layers, and L_t the number of transformer layers in our UGT model as well as in the baseline models (FREEDOM and LightGT). As introduced in Section 3.2, both FREEDOM and LightGT use Deeper CNN and Sentence-BERT in their feature extraction process. Let also L_{cnn} denote the CNN depth, R the spatial resolution, and C the average channel width in Deeper CNN. Similarly, let L_{txt} represent the number of layers and T_{txt} the token length of Sentence-BERT. In addition, we denote by q the number of modalities in the feature extraction process for both FREEDOM and LightGT. In contrast, UGT's multi-way transformer explicitly encodes the modalities into single, unified representations, thus it does not use q as a parameter in its modality fusion process. In UGT's multi-way transformer, we denote the combined sequence length – comprising image patches and text tokens – as T, while d represents the hidden dimensionality. In the following, we leverage these notations to provide a detailed breakdown of the computational efficiency analysis for UGT and the baselines, as summarised in Table 6.3.

6.2.7.1 Feature extraction

During the feature extraction process, FREEDOM and LightGT compute item features using the Deeper CNN model for visual data and a Sentence-BERT model for textual data, where $L_{\text{cnn}} = 50$, $L_{\text{txt}} = 12$ and $T_{\text{txt}} = 256$ (Szegedy et al., 2015). For UGT's multi-way transformer, we use T as the combined sequence length. We derive this combined length from image patches and text tokens, where $T = 197$ (image patches) $+ 128$ (text tokens) $= 325$. The 197 visual tokens refer to 196 image patches and an additional learnable class token. The 196 image patches are derived from an input image resolution of 224×224 pixels. By dividing the image's height and width by the 16×16 pixel patch size, we obtain $224/16 = 14$ patches along each dimension, resulting in 14×14 = 196 patches. Given that the layer counts are identical for UGT's multi-way transformer and the Sentence-BERT used by FREEDOM and LighGT ($L_{\text{txt}} = L_{\text{t}} = 12$), the textual sequence processed by the multi-way transformer (128 tokens) is smaller than Sentence-BERT (256 tokens). As shown in Table 6.3, UGT's textual feature extraction complexity is given as $\mathcal{O}\big(N_i\, L_t\, (128)^2\, d\big)$. This shows that the complexity of using a multi-way transformer reduces the attention cost for the textual modality during the feature extraction process in comparison to Sentence-BERT's complexity of $\mathcal{O}\big(N_i\, L_t xt\, (256)^2\, d\big)$. Due to the architectural differences between Deeper CNN and the multi-way transformer, a direct token-wise comparison for the visual modality is not feasible. Consequently, to provide a comprehensive assessment of the efficiency of UGT and the baselines in the feature extraction process, we can compare the total parameter counts of their respective feature extraction components as an alternative metric. The total parameter count of UGT's multi-way transformer is approximately 167 million (c.f. Section 6.2.2), which is comparable to the $\sim$170 million parameters associated with the combined Deeper CNN and Sentence-BERT used by the baselines (Yi et al., 2025). This demonstrates that the multi-way transformer used in UGT is computationally competitive with the combination of Deeper CNN and Sentence-BERT, while our UGT model also achieves a better effectiveness

Table 6.3: Comparative computational complexity per epoch and recommendation effectiveness. * indicates a statistically significant difference according to a paired t-test with the Holm-Bonferroni correction with $p < 0.05$. The best performance of each model is highlighted in bold.

Complexity	Feature Extraction (Textual)	Feature Extraction (Visual)	Modality Fusion	Recall@20	NDCG@20
FREEDOM	$\mathcal{O}\Big(N_i\, L_{\text{cnn}}\, R^2\, C^2\, d\Big)$	$\mathcal{O}\Big(N_i L_{\text{txt}}\, T_{\text{txt}}^2\, d\Big)$	$\mathcal{O}\big(L_g(\mathcal{E}+qN_i)d\big)$	0.0918*	0.0449*
LightGT	$\mathcal{O}\Big(N_i\, L_{\text{cnn}}\, R^2\, C^2\, d\Big)$	$\mathcal{O}\Big(N_i L_{\text{txt}}\, T_{\text{txt}}^2\, d\Big)$	$\mathcal{O}\big((L_g+L_t)(\mathcal{E}+qN_i)d\big)$	0.0953*	0.0440*
UGT	$\mathcal{O}\big(N_i\, L_t\, T^2\, d\big)$		$\mathcal{O}\big(L_g\, \mathcal{E}\, d\big)$	**0.1034**	**0.0477**

compared to these two strong baselines, as evidenced by its higher Recall@20 and NDCG@20 scores in Table 6.3.

6.2.7.2 Modality fusion

As described in Section 6.1, UGT aggregates all modalities in a single LightGCN pass, resulting in a per-epoch cost of $\mathcal{O}(L_g \mathcal{E} d)$. We can derive this complexity from the L_g graph propagation layers, where the model aggregates information across all $\mathcal{E}$ user-item interactions by passing embeddings of dimension d. In contrast, FREEDOM and LightGT perform separate propagation steps for each modality, leading to per-epoch costs of $\mathcal{O}\big(L_g(\mathcal{E} + kN_i)d\big)$ and $\mathcal{O}\big((L_g + L_t)\mathcal{E}d\big)$, respectively, as shown in Table 6.3. In terms of model size of the modality fusion components, UGT's unified GNN has 16 million parameters – markedly fewer parameters than FREEDOM's GNN (36 million parameters) and LightGT's graph transformer (51 million parameters). We empirically measured these parameter counts from the implementation of the baseline models. These parameter counts, along with the computational complexity analysis clearly demonstrate that UGT's fusion component is more efficient for the top-K multi-modal recommendation task.

In response to RQ6.4, we can conclude that UGT achieves a good balance between effectiveness and efficiency compared to the best-performing baselines, namely FREEDOM and LightGT, in both feature extraction and modality fusion processes.

6.2.8 Modality Alignment of UGT (RQ6.5)

Table 6.1 has shown that UGT achieves superior performance by addressing the challenges of Isolation 1 and Isolation 2. In this section, we analyse whether the improved performance is reflected in the quality of features produced by our UGT model in comparison to those generated by the baselines. In the following, we examine the most competitive baseline, namely FREEDOM. To assess the quality of the generated features, we measure the average Mean Squared Error (MSE) between the visual and textual embeddings produced by UGT and FREEDOM. This metric allows to assess how well the multi-modal embeddings align. We also use t-SNE visualisations (Van der Maaten and Hinton, 2008) to show the visual and textual embeddings from UGT and FREEDOM across the used datasets. Figure 6.5 shows the visualisations of the

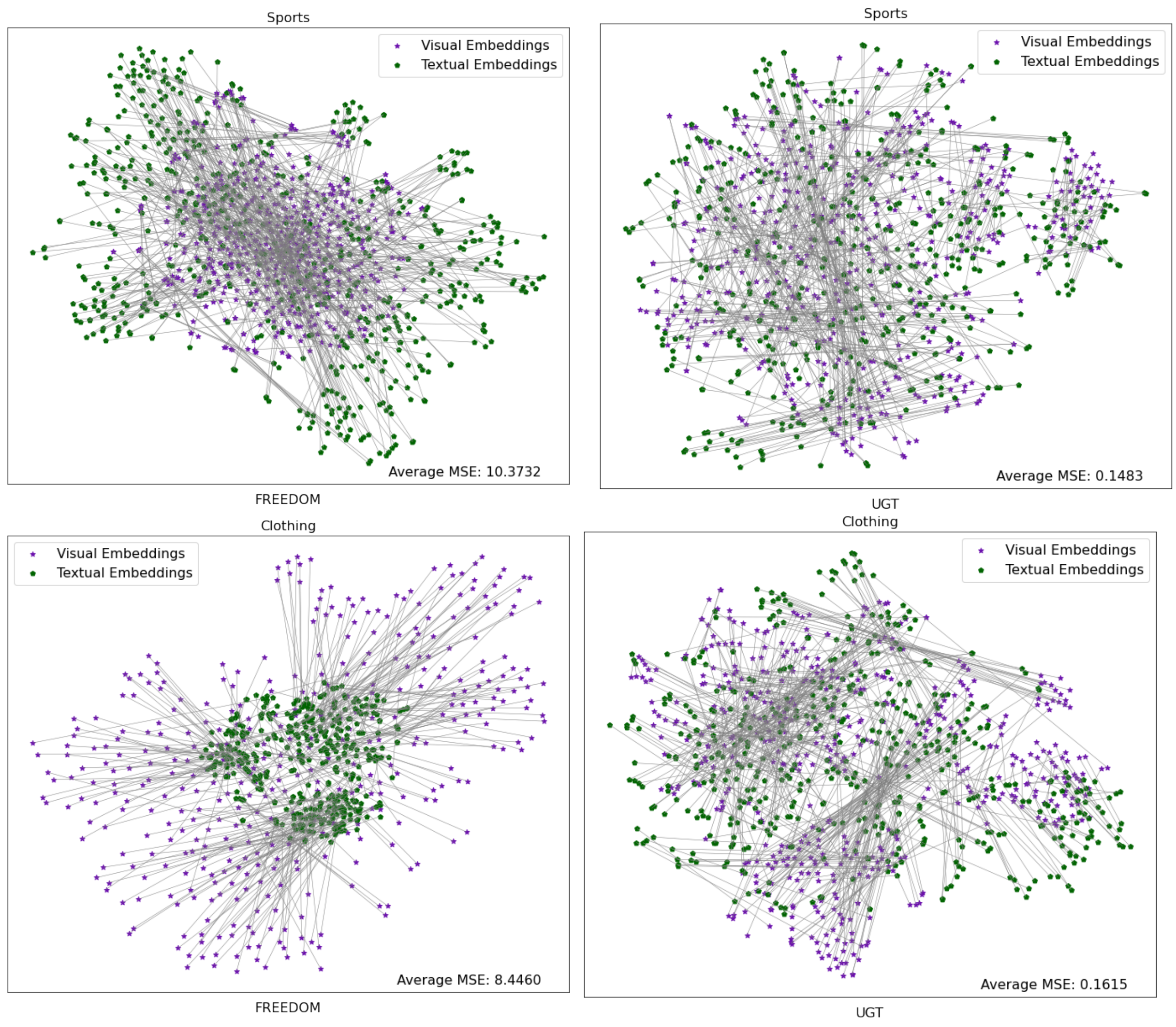


Figure 6.5: The t-SNE visualisation of the item embeddings on the Sports and Clothing datasets. The star refers to the visual embeddings while the pentagon represents the text embeddings. The average Mean Squared Error (MSE) value indicates the average distance between the visual and textual embeddings.

obtained visual and textual embeddings on the Sports and Clothing datasets, with stars and pentagons representing the visual and textual embeddings, respectively. We observe the same trends and conclusions on the Baby dataset. In Figure 6.5, we link the two types of embeddings with a light grey line to show their pairwise relationship. We anticipate that higher-quality multi-modal features exhibit cohesive distributions and reduced Mean Squared Error (MSE) values, indicating a successful integration of various modalities into a unified representation. Conversely, poorer-quality features likely appear more dispersed and have higher MSE values, suggesting an insufficient alignment between the modalities. From Figure 6.5, we observe that the visual embeddings of items in FREEDOM are densely clustered in the central space of the Sports dataset while their corresponding textual embeddings are more dispersed around this central cluster. In contrast, the UGT model produces visual and textual embeddings that are cohesively distributed, indicating a tighter semantic space on the Sports dataset. The markedly lower average MSE value (0.1483) for the UGT model compared to that of FREEDOM (10.3732) on the Sports dataset further demonstrates that UGT better aligns the visual and textual embeddings than FREEDOM. We observe similar trends in the Clothing dataset as shown in Figure 6.5, where the embeddings of UGT are cohesively distributed, in contrast to FREEDOM's embeddings, which are more widely scattered between the resulting visual and textual representations. Indeed, UGT exhibits a lower average MSE value (0.1615) compared to FREEDOM (8.4460), confirming that UGT more effectively unifies the multi-modal embeddings into a tighter semantic space. Overall, these observations indicate that UGT, by addressing the issues of isolated feature extraction and modality encoding through a unified approach, produces higher-quality multi-modal features than FREEDOM. The tighter alignment of the visual and textual embeddings within UGT is reflected in its improved performance, highlighting the importance of better aligning the multi-modal data for an effective multi-modal recommendation system.

Hence, in response to RQ6.5, our UGT model aligns the multi-modal embeddings more cohesively than the strongest baseline FREEDOM, resulting in higher-quality features and a more enhanced recommendation performance.

6.2.9 Modality Contribution Study (RQ6.6)

In this section, we examine the contribution of each modality to the recommendation performance of both UGT and the multi-modal recommendation baseline models. This analysis aims to evaluate whether our UGT model fuses the extracted multi-modal features more effectively into the final user/item representations compared with the baseline models. Figure 6.6 shows the recommendation performance using single modality data (i.e., textual or visual features) and combined modality data (i.e., textual & visual features) in both the UGT and baseline models on the Sports and Clothing datasets. We note that we observe similar trends on the Baby dataset. For the VBPR baseline, which only uses visual features, we report the results using only the visual features, since this model does not use any textual or combined features. We report the recommendation performance in terms of Recall@20 and NDCG@20. The results from Figure 6.6 show that while all tested baselines excepting VBPR (MMGCN, MMGCL, SLMRec, LATTICE, LightGT, BM3, FREEDOM) achieve their best performance using only textual or visual features on both metrics, our UGT model achieves the best performance with the combined textual & visual features. In particular, Figure 6.6 shows that even though UGT's performance with a single modality is comparable to BM3's performance on the Sports dataset – where BM3 is the strongest baseline model, which also combines modalities – our UGT model enhances its performance by effectively fusing both the visual and textual modalities into a combined representation. Similarly, from Figure 6.6 of the Clothing dataset, we observe that UGT not only outperforms the most effective baseline, namely FREEDOM, within each individual modality, but also achieves better results when combining the two modalities. This indicates that UGT

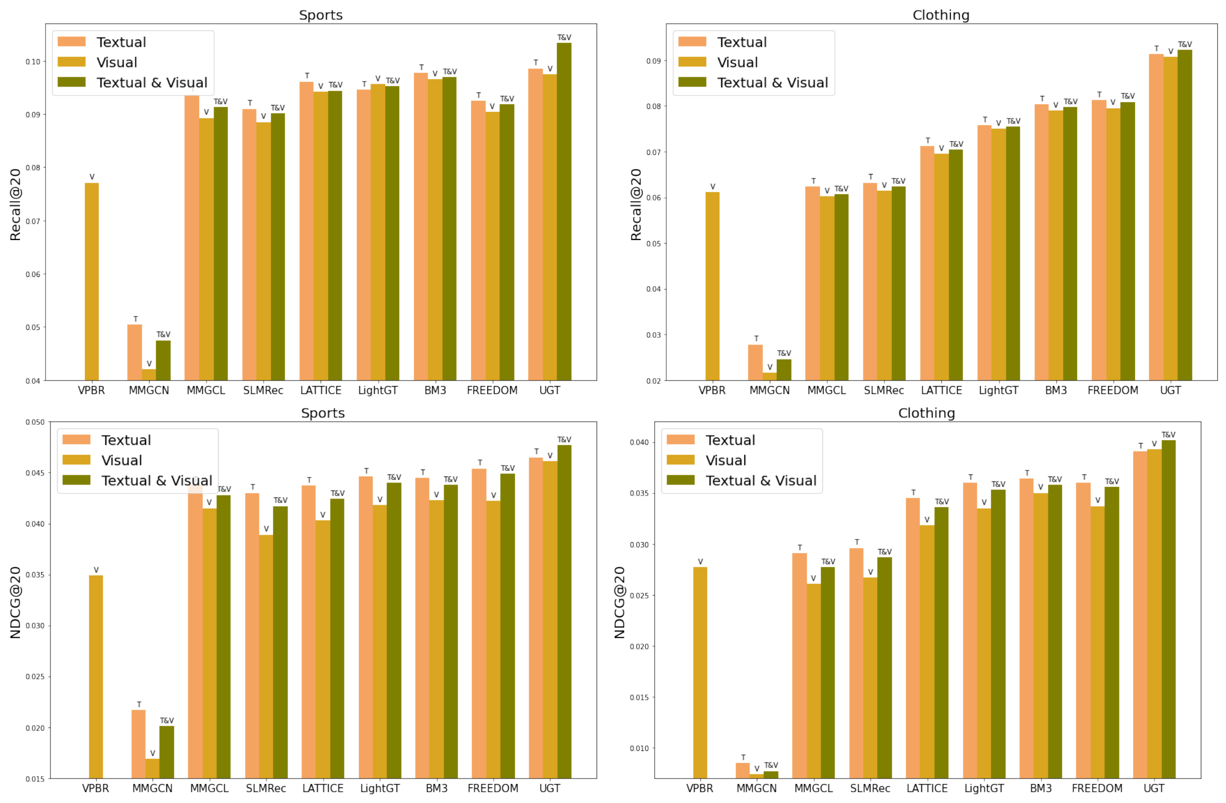


Figure 6.6: Comparative analysis of the Recall@20 and NDCG@20 scores of our UGT model and the used baseline models using different modality inputs across the used datasets. V and T are the abbreviations for Visual and Textual, respectively.

more effectively fuses the multi-modal features compared to all tested baselines, thereby enhancing the overall recommendation performance.

In response to RQ6.6, we conclude that our UGT model effectively leverages each individual modality, resulting in an improved performance over the existing top-K multi-modal recommendation baselines in terms of both the Recall@20 and NDCG@20 metrics when the used modalities are combined.

6.2.10 Cold-start Analysis (RQ6.7)

We now examine the effectiveness of the UGT model in a cold-start scenario, focusing on target users with fewer than 10 interactions. We perform this analysis across all used datasets to assess how effectively our UGT model estimates the profiles of the cold-start users in the top-K multi-modal recommendation task. Table 6.4 presents the performance of UGT for both the cold-start and regular users, in comparison to the best-performing baseline, FREEDOM, in terms of Recall@20 and NDCG@20. From Table 6.4, we observe that the UGT model shows a significantly enhanced performance for cold-start users compared to FREEDOM on both metrics across all three used datasets, according to the paired t-test ($p < 0.05$). This improvement suggests that UGT effectively uses the multi-way transformer's pre-trained and domain-specific knowledge

Table 6.4: Cold-start analysis results for UGT and the FREEDOM baseline. 'Cold-start' denotes the cold-start users and 'Regular' denotes the regular users in the used datasets. * denotes a significant difference between UGT and FREEDOM using the paired t-test with $p < 0.05$.

Datasets		UGT (Recall@20)	FREEDOM (Recall@20)	%Improv. (Recall@20)	UGT (NDCG@20)	FREEDOM (NDCG@20)	%Improv. (NDCG@20)
Sports	Cold-start	0.0856*	0.0764	12.04%	0.0393*	0.0341	15.25%
	Regular	0.1370*	0.1243	10.22%	0.0511*	0.0456	12.06%
Clothing	Cold-start	0.0741*	0.0639	18.92%	0.0320*	0.0266	20.30%
	Regular	0.1167*	0.1028	13.52%	0.0423*	0.0381	11.02%
Baby	Cold-start	0.0642*	0.0580	10.69%	0.0356*	0.0316	12.66%
	Regular	0.1304*	0.1197	8.94%	0.0455*	0.0413	10.17%

within the target datasets to refine the estimation of user preferences, thereby achieving a superior recommendation performance compared to FREEDOM. In addition, Table 6.4 shows that UGT achieves higher improvements for cold-start users compared to regular users across all three datasets. For example, on the Clothing dataset, our UGT model achieves a substantial improvement in recommendation performance for cold-start users, outperforming FREEDOM by 18.92% in terms of Recall@20 and 20.30% in terms of NDCG@20. In contrast, for regular users, the improvements are 13.52% and 11.02% on the same metrics, respectively. This result suggests that the UGT model successfully enables more effective multi-modal representation learning, thereby incorporating useful context and multi-modal information to the representations of the cold-start users. Consequently, our UGT model indeed benefits the target users with sparse interactions in multi-modal recommendation.

Hence, in response to RQ6.7, we conclude that our UGT model successfully leverages its pre-trained multi-modal knowledge as auxiliary information to significantly enhance the recommendation performance for cold-start users.

6.2.11 Out-of-distribution Performance (RQ6.8)

In this section, we assess the out-of-domain performance of our UGT model, focusing on its top-K recommendation capabilities in another recommendation scenario (i.e., video recommendation). This analysis allows us to investigate the generalisability of the UGT model by training exclusively on previously used Amazon e-commerce datasets. We then perform inference on different domain datasets, specifically those from micro-video recommendation datasets. In particular, we perform this generalisability study of UGT on three video recommendation datasets [6] – Douyin, Bilibili and Kuaishou – where each micro video is associated with a cover image (visual data) and its video description (textual data). Table 6.5 compares the performance of UGT against all multi-modal recommendation baselines (VBPR, MMGCN, MMGCL, SLMRec, LATTICE, BM3, FREEDOM, PMGT and LightGT). We aim to determine if our UGT model trained on e-commerce datasets still has a comparable recommendation performance in the video recommendation scenario. Table 6.5 reports the NDCG@20 and Recall@20 performance of the UGT model as well as the multi-modal baselines. From Table 6.5, we observe that our UGT model outperforms all the multi-modal recommendation baselines on all three used

[6] https://github.com/westlake-repl/NineRec

Table 6.5: Experimental results comparing our UGT model with the used baselines in multi-modal recommendation in the micro-video recommendation scenario. The best performance of each model is highlighted in bold. * denotes a significant difference compared to the indicated baseline performance using the paired t-test with the Holm-Bonferroni correction for $p < 0.05$.

Dataset	Douyin		Bilibili		Kuaishou	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
VBPR	0.0536*	0.0484*	0.0369*	0.0503*	0.0487*	0.0474*
MMGCN	0.0498*	0.0397*	0.0352*	0.0491*	0.0460*	0.0458*
MMGCL	0.0613*	0.0576*	0.0446*	0.0596*	0.0575*	0.0539*
SLMRec	0.0604*	0.0542*	0.0440*	0.0577*	0.0567*	0.0623*
LATTICE	0.0584*	0.0519*	0.0427*	0.0564*	0.0534*	0.0521*
BM3	0.0621*	0.0578*	0.0462*	0.0621*	0.0586*	0.0562*
FREEDOM	0.0616*	0.0573*	0.0448*	0.0612*	0.0604*	0.0566*
PMGT	0.0527*	0.0453*	0.0394*	0.0522*	0.0403*	0.0514*
LightGT	0.0608*	0.0551*	0.0421*	0.0562*	0.0577*	0.0530*
UGT	**0.0647**	**0.0620**	**0.0483**	**0.0657**	**0.0636**	**0.0613**
%Improv.	4.19%	7.27%	4.54%	5.80%	5.30%	8.30%

micro-video datasets. This result demonstrates that our UGT model successfully generalises multi-modal knowledge from e-commerce recommendation scenarios to video recommendation contexts. As mentioned in Section 6.1, the key component of UGT – the multi-way transformer – has been pre-trained on a large web corpus, and we further fine-tune UGT for the recommendation scenario as outlined in Section 6.1.5. Given the strong performance of UGT reported in Table 6.5, this suggests that the fine-tuning process, which integrates both general web knowledge and domain-specific information, enhances UGT's ability to generalise to out-of-domain scenarios. Consequently, the generalisability of UGT is likely the result of its inherent fine-tuning process leveraging a general web corpus and domain-specific knowledge tailored for the top-K multi-modal recommendation task.

In response to RQ6.8, we conclude that UGT effectively generalises from the e-commerce scenario to the micro-video recommendation scenario and significantly outperforms all multi-modal recommendation baselines. This successful generalisability in micro-video scenarios is likely the result of UGT's effective use of both the web corpus knowledge embedded in its parameters and the domain-specific knowledge gained from its fine-tuning stage on the e-commerce datasets.

6.2.12 Qualitative Analysis

We present a case study to illustrate the effectiveness of our UGT model in identifying useful multi-modal information from noisy user-item interactions and in making accurate recommendations accordingly. For this analysis, we randomly select 20 users from a subset of the total

Ground Truth of User 5387	Item 15190 (training)	Item 1530 (training)	Item 7191 (training)	Item 14156 (test)	Item 15219 (test)	Item 6656 (test)	Item 8773 (test)	Item 2307 (test)
Item Texts	Mad Engine Men's Mustang Emblem T-Shirt Clothing, Shoes & Jewelry Men Clothing Shirts T-Shirts	Casio Men's AW80V-1BV Casio Clothing, Shoes & Jewelry Men Watches Wrist Watches	Happy Feet Men's/Women's Original Foot Alignment Socks Sports & Outdoors Clothing	New Solid Winter Long Beanie TopHeadwear Clothing, Shoes & Jewelry Men Big & Tall	Fruit of the Loom Men's Crewneck Tee, Five-Pack Clothing, Shoes & Jewelry Men Clothing Underwear Undershirts	DR. WHO -- TARDIS -- Men's Tee Clothing, Shoes & Jewelry Men Big & Tall	Dr. Scholl's Diabetic Ankle Sock, 2-pk Health & Personal Care Medical Supplies & Equipment Diabetes Care Socks & Insoles	Orient Men's CEM65001M; Orange Mako; Stainless Steel Automatic Dive Watch Orient Sports & Outdoors Accessories Sport Watches
Item Image								
FREEDOM				×	×	-	-	-
UGT				-	×	×	×	×

Figure 6.7: Case study on User #5387 in the Sports dataset. × indicates a correct item recommendation for the given user.

35493 test users from the Sports dataset, who exhibit improved performance with UGT compared to FREEDOM (the strongest used baseline). We examine the products recommended to these users, along with their raw images and descriptions. From these users, we randomly select one user as an illustrative example. We ensure the representativeness of our results by selecting a user that shows a median level of performance improvement with respect to the baseline within our user pool. This approach allows to have a balanced view of a typical user's experience, while illustrating how the UGT model helps to enhance recommendation. Figure 6.7 shows the items recommended by the FREEDOM and UGT models for the selected target user, as well as the items the user has interacted with, with the corresponding images and texts presented in the second and third rows, respectively. From Figure 6.7, we observe that the products purchased by the user in the Sports dataset mainly fall into three categories of products: Men's t-shirts, Sport watches and Sport accessories. In the recommended products by FREEDOM in the test set, there is a men's t-shirt (item #15219) and a beanie hat (item #14156). Our UGT model predicts the same T-shirt item as FREEDOM, but not the beanie hat. However, UGT also recommends three more correct products to the user: a t-shirt (#6656), a sock (#8773) and a watch (#2037). All these three items are semantically relevant to the items in the training set (i.e., #15190, #7191, #1530). For example, in the Sports category, item #2307 in the test set shares the 'watch' keyword with the training item #1530 in their textual descriptions. This example shows how UGT, unlike FREEDOM, leverages detailed item characteristics to generate recommendations that are more useful and semantically relevant to the target user.

We also highlight another case study involving a different user, who is randomly selected from the same pool of 35,493 test users and who also exhibited an improved performance with UGT compared to another strong baseline, namely LightGT (see Figure 6.8). We observe trends similar to those illustrated with FREEDOM. These case studies indicate that our UGT model

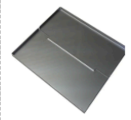
Ground Truth of User 826	Item 1122 (training)	Item 2362 (training)	Item 921 (training)	Item 2348 (test)	Item 247 (test)	Item 15219 (test)	Item 7146 (test)	Item 9168 (test)
Item Texts	Coleman Tent Light Coleman Sports & Outdoors Outdoor Gear Camping & Hiking Lights & Lanterns Lanterns	Schwinn Katana Road Bike (54cm Frame) Sports & Outdoors Cycling Bikes Road Bikes The Schwinn S2704 700c Men's Katana Bicycle	Victorinox Explorer Victorinox Sports & Outdoors Outdoor Gear Camping & Hiking Knives & Tools Folding Knives	Heininger Automotive SportsRack BedRack Heininger Sports & Outdoors Cycling Transportation & Storage Car Rack Accessories	Picnic Time Portable Reclining Camp Chair, Black/Gray Sports & Outdoors Outdoor Gear Camping & Hiking	Kershaw Amphibian - Kydex Sheath Knife Kershaw Sports & Outdoors Hunting & Fishing Hunting Hunting & Tactical Knives Hunting Knives	180 Tack Snow And Ash Pan 2 Piece 180ST-AP2P-s 180 Tack Sports & Outdoors Outdoor Gear Camping & Hiking Camp Kitchen Camp Stoves	Nite Ize SBP2-03-01BG S-Biner Plastic Size-2 Double Gated Carabiner, Black Nite Ize Sports & Outdoors
Item Image								
LightGT				-	-	×	×	-
UGT				×	×	×	×	-

Figure 6.8: Case study on User #826 in the Sports dataset. $\times$ indicates a correct item recommendation for the given user.

predicts similar items according to the visual/textual semantics of the candidate items by leveraging the visual and textual characteristics of items. As illustrated by the increased number of accurate recommendations to the target user in Figure 6.8, our UGT model successfully enhances the relevance of recommendations within the context of multi-modal scenarios. Hence, our UGT model identifies the visual and textual semantics of potential items more effectively than FREEDOM and LightGT, thereby enabling better recommendations for the target user.

6.2.13 Discussion

Unlike the graph augmentation and modality alignment approaches proposed in Chapter 5 that focus solely on the modality fusion stage within the top-K multi-modal recommendation pipeline, we introduce a unified model that integrates the entire process – from data ingestion and feature extraction, through multi-modal fusion, to representation learning and ranking – into an end-to-end model. This unified design enables our UGT model to more effectively mine self-supervised signals for top-K multi-modal recommendation through the SSL loss (i.e., ITC loss), aligning with our thesis statement in Section 1.2.

6.3 Conclusions

In this chapter, we overcome the longstanding isolated feature extraction and isolated modality encoding problems by proposing an SSL-enhanced Unified Multi-modal Graph Transformer (UGT) model for effective top-K mult-modal recommendation. Our UGT model first leverages a multi-way transformer for feature extraction and then integrates these features into a new unified GNN, which serves as the modality fusion component for top-K multi-modal recommendation. Our results on the three benchmark datasets demonstrate that UGT delivers substantial performance gains over thirteen high-performing baseline models, including both established methods and state-of-the-art multi-modal recommenders, achieving improvements of up to 13.97% (see Table 6.1). In addition, an ablation study confirmed the positive contribution of each UGT component to the overall recommendation performance (see Table 6.2). We further showed in Table 6.3 that UGT achieves computational and parameter efficiency comparable to the strongest baselines, namely FREEDOM and LightGT. Furthermore, we evaluated the quality of the learned representations and found that UGT produces more aligned visual–textual embeddings than FREEDOM, as evidenced by lower average MSE distances (see Figure 6.5), further confirming UGT's recommendation effectiveness. We also analysed the contribution of each modality and showed that UGT substantially enhances the utility of both visual and textual signals in multi-modal recommendation (see Figure 6.6). Furthermore, a cold-start analysis demonstrated that UGT enriches representations for cold-start users, thereby improving recommendation performance (see Table 6.4). Finally, UGT, originally trained for e-commerce scenarios, generalised effectively to an out-of-domain micro-video recommendation setting, outperforming multi-modal recommendation baselines (see Table 6.5).

In summary, we have validated one of the hypothesis of our proposed thesis statement in Section 1.2, namely that graph-based recommender systems can be effectively enhanced by effectively mining self-supervised signals from multiple modalities. However, despite the success in Chapters 5 and 6 in obtaining richer self-supervised signals from multiple modalities, we argue that multi-modal signals alone are not sufficient for fully enhancing graph-based recommender systems. Indeed, as discussed in Section 3.3, these systems can benefit further from self-supervised signals obtained across multiple domains. Therefore, in Chapter 7, we investigate how to effectively mine additional supervision signals for graph-based recommender systems across domains. In particular, we explore SSL-based pre-training combined with graph prompting to enable efficient knowledge transfer, thereby enhancing top-K recommendation performance in a cross-domain setting.

Chapter 7

Self-supervised Cross-domain Graph Prompting

In Chapters 5 and 6, we explored complementary approaches that improve modality fusion and establish a more effective unified pipeline for graph-based recommender systems using a self-supervised learning (SSL) paradigm for top-K multi-modal recommendation. These contributions empirically validated the thesis's argument that graph-based recommender systems can be enhanced by mining self-supervised signals from multiple modalities. However, as argued in Sections 1.1 and 3.3, mining supervision signals from multiple *domains* provides an additional and equally valuable direction for achieving further performance gains in graph-based recommender systems. In particular, as discussed in Section 3.3, SSL techniques in graph-based models can be extended to extract cross-domain supervision signals, thereby improving the transferability and generalisation capabilities of graph-based recommender systems for top-K recommendation tasks. In this chapter, we focus on top-K recommendation in a cross-domain setting, unlike Chapter 4, which focused on top-K general recommendation, and Chapters 5 and 6, which focused on top-K multi-modal recommendation. However, as mentioned in Section 3.3, graph-based recommender systems still lack sufficient methods for effectively transferring domain knowledge from source domains to target domains.

A key line of research that attempts to address this challenge is Cross-Domain Recommendation (CDR) (Tang et al., 2016; Zang et al., 2022), which relies on shared users over pairs of domains to transfer relevant information from the source domain to the target domain. By using such shared users, CDR alleviates the negative effect of sparse interactions, so as to improve the top-K recommendation performance when applied to a different target domain. As mentioned in Section 3.3, recent advances in natural language processing centred on the paradigm of first pre-training then fine-tuning. Following this paradigm, several cross-domain recommendation techniques (Wang et al., 2021a; Zhu et al., 2022) also considered the application of this pre-training and fine-tuning mechanism to facilitate more effective knowledge transfer across domains. In particular, a common development routine for shared users in CDR involves first the pre-training

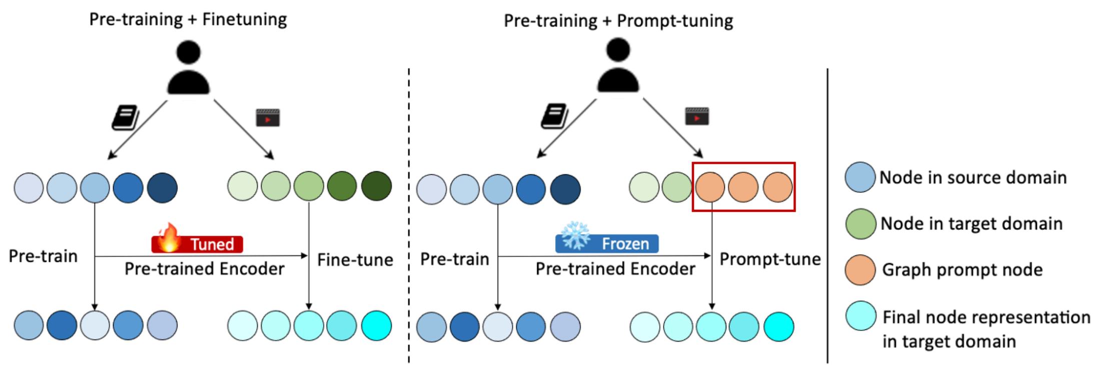


Figure 7.1: Overview of conventional graph fine-tuning and personalised graph prompt-tuning in Cross-Domain Recommendation (CDR).

of the recommendation model using data from the source domain to learn domain-invariant knowledge (e.g., user profiles). Then, the knowledge contained in the pre-trained model is used to initialise the user/item embeddings in the target domain. Such a strategy has been shown to be effective in alleviating the data sparsity issue (Xiao et al., 2021). Moreover, several graph-based recommender systems (Meng et al., 2021; Wang et al., 2021a) leveraged pre-training on the structural data from the source domain, which can capture collaborative filtering signals to yield better user/item representations through the high-order connectivity via graph convolutional operations (Welling and Kipf, 2016). However, as discussed in Section 3.3, the effectiveness of recommendation on the target domain can be negatively impacted if the pre-trained model learns from non-relevant domain-specific features in the source domain. On the other hand, in the Natural Language Processing (NLP) field, prompt-tuning (Brown et al., 2020; Lester et al., 2021) has been widely applied to address many NLP tasks and has shown its usefulness in extracting useful information from a pre-trained model, such that it can be adapted to a downstream task with less efforts. Several prompt-variants have been proposed, such as continuous embeddings (i.e. soft prompt) (Li and Liang, 2021; Qin and Eisner, 2021), which effectively bridges the general knowledge from the pre-training stage to a learned context of various downstream tasks.

Several studies (Cui et al., 2022; Geng et al., 2022; Wu et al., 2024) have investigated the application of prompt techniques within the sequential recommendation context, focusing on leveraging user and item information to generate customised prompts that can effectively address specific recommendation tasks. For example, Wu et al. (2024) proposed a soft prefix prompt based on users' profiles to tackle the cold-start problem in a sequential recommendation setting. Another line of work (Cui et al., 2022; Geng et al., 2022) attempted to convert the available data resources, such as the user-item interactions, the user descriptions, the item metadata, and the user reviews, into natural language sequences. Next, these approaches used a composition of such sequences as a prompt and the pre-trained models for addressing various downstream tasks of the recommender system (e.g., cold-start recommendation, few/zero-shot recommendation and user profile prediction). The aforementioned examples show that existing prompt tuning-based recommendation techniques rely on sequential patterns in the users' interacted items and mimic NLP models designed for sequential text. This demonstrates the feasibility of using the prompt-tuning techniques in the context of recommendation systems.

Inspired by the success of prompt-tuning in sequential recommendation, in this chapter, we aim to adapt the prompt-tuning paradigm into graph-based recommender systems. However, it is challenging to design a prompt-tuning technique that makes a full use of knowledge from structural data rather than from sequential data in graph-based recommender systems. Therefore, it is essential to bridge the graph-based recommender systems with the existing prompt-tuning techniques, which use a sequential modelling of data. In this section, we show that the language models in the NLP domain could be a special form of a GNN (as will be detailed in Section 7.1.3.1), which allows us to propose novel templates for personalised graph prompts. These templates could then be applied to graph-based recommenders. As an illustrative example, Figure 7.1 shows how our proposed prompt-tuning framework contrasts with conventional fine-tuning approaches in a CDR setting. As discussed in Section 3.3 and also at the start of this chapter, existing CDR solutions typically pre-train a graph encoder on a source domain and then fine-tune the trained encoder using interaction data from the target domain. Therefore, in this chapter, we propose a prompt-tuning framework as an alternative to the fine-tuning process, aiming to effectively and efficiently leverage personalised graph prompts while keeping the pre-trained parameters fixed. To be more specific, we build personalised graph prompts with a series of relevant items, which we call the neighbouring-items[1] and leverage these prompts to enhance the final user/item representations. To this end, we propose a novel Personalised Graph Prompt-based Recommendation (PGPRec) framework, which encapsulates our proposed personalised prompts as well as a Contrastive Learning (CL)-empowered graph-based recommender for cross-domain recommendation. Indeed, with the integration of a graph-based contrastive learning, the pre-trained model enforces the divergence of the learned user/item embeddings (Wu et al., 2021a), with the aim to make them generalisable to various target domains to achieve an an improved recommendation performance. The remainder of this chapter is structured as follows:

- Section 7.1 presents the tackled top-K cross-domain task and the Personalised Graph Prompt-based Recommendation (PGPRec) framework for transferring structural graph knowledge, using SSL-based pre-training and prompt-tuning, from source domains to target domains;

- Section 7.2 describes the experimental setup and our research questions in this chapter;

- Section 7.3 summarises the key insights derived from the model proposed in this chapter.

[1] Given an item, we denote by its "neighbouring-items" those items that share its same attributes in the target domain, i.e. the set of items with the same attributes. In this chapter, we define the neighbouring-items according to three types of available metadata – i.e. also_bought, also_viewed and bought_together – and refer to them as 'neighbouring-items' throughout this section.

7.1 Personalised Graph Prompting

In this section, we first present the cross-domain recommendation task (Section 7.1.1). Next, we describe the composition of our proposed personalised graph prompts and training process (Section 7.1.2). We also describe how to derive the personalised graph prompt-tuning paradigm by justifying the equivalence between the transformer and GNN (Section 7.1.3), along with the detailed implementations and objectives during the training phases (Section 7.1.5).

7.1.1 Task Definition

In this section, we focus on the top-K recommendation task in a cross-domain setting. Specifically, we extend the notations introduced in Chapter 4 from the single-domain top-K recommendation to the top-K cross-domain recommendation task. Conceptually, we consider two domains, the source domain D_{S} and the target domain D_{T}. Following the same notations defined in Section 4.1.2.1, the set of users in both domains is shared, and denoted by $\mathcal{U}$ (of size $n_u = |\mathcal{U}|$). Let the set of items in the source domain and target domains be $\mathcal{I}_{\mathrm{S}}$ (of size $n_s = |\mathcal{I}_{\mathrm{S}}|$) and I_{T} (of size $n_t = |\mathcal{I}_{\mathrm{T}}|$), respectively. Then, we aim to make effective and efficient recommendations to users in $\mathcal{U}$ with a ranked list of items from the target domain $\mathcal{I}_{\mathrm{T}}$, while leveraging the learned knowledge from the source domain D_{S}. Following the same definition of the user–item interaction graph $\mathcal{G}$ in Section 4.1.2.1, we construct two bipartite graphs $\mathcal{G}_S$ and $\mathcal{G}_T$ for both the source and target domains, where the nodes represent users/items and the edges indicate interactions between the users and items. Formally speaking, with the interaction graphs $\mathcal{G}_S$ and $\mathcal{G}_T$, we pre-train a graph encoder f in a selected source domain D_{S} and adapt it to the target domain D_{T} for an enhanced knowledge of the user preferences so as to effectively and efficiently recommend the top-K items related to their interests in the target domain.

7.1.2 The PGPRec Framework

First, we provide an overview of our proposed recommendation framework, called Personalised Graph Prompt-based Recommendation (PGPRec). In particular, Figure 7.2 shows and illustrates the major components included in our proposed framework. Specifically, for our cross-domain recommender, we adopt the GNN technique to model the data from each domain, in order to capture the high-order features and allowing the transfer of additional structural knowledge aside from the typical user/item features (e.g., the user profiles and item attributes). To instantiate the discussed GNN technique, we are not limited to the commonly used GNN model as introduced in Section 2.2. Indeed, PGPRec is also flexible in allowing to use other graph-based techniques, such as GAT, NFCG, or LightGCN, described in Section 3.1. Since prompt-tuning has shown strong performance in NLP and sequential recommendation, we extend this technique by introducing graph prompts for top-K cross-domain recommendation. In particular, as a rationale for

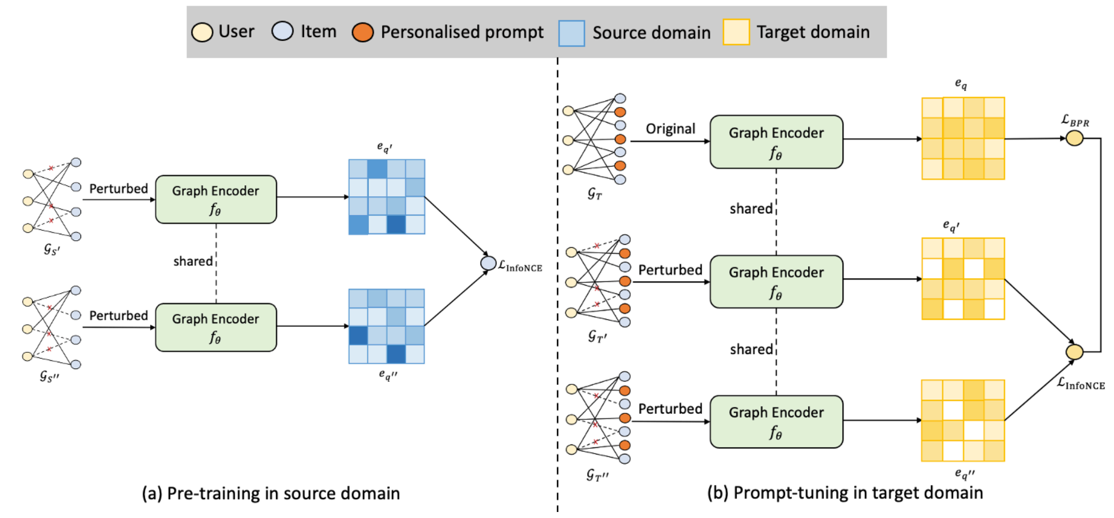


Figure 7.2: An illustration of the PGPRec framework using a specific example. The framework is mainly composed of two steps: (a) In the pre-training stage, PGPRec trains a GNN model to learn the source domain's transferable knowledge through a contrastive objective; (b) Then PGPRec uses the pre-trained GNN model and the personalised graph prompts to enable the prompt-tuning in the target domain through multi-task learning.

applying prompt-tuning to a graph model, we argue that the GNN technique can be considered as another type of a transformer model. We further explain and justify this argument in Section 7.1.3. In our PGPRec framework, we propose a mixture of hard and soft graph prompts for personalised recommendation, which we collectively denote as *personalised graph prompts*. We describe the hard graph prompts. For each user, we develop personalised prompts according to the associated relevant items (denoted as the neighbouring-items). Given an item, we denote by its "neighbouring-items" those items that share its same attributes in the target domain, i.e., the set of items with the same attributes.

In this chapter, we refer to the use of relevant items according to three available attributes in the metadata (i.e. also_bought, also_viewed and bought_together) of the used Amazon datasets, which are used in our experiments. However, such neighbouring-items are not necessarily fixed and can be naturally changed or extended depending on the used datasets. Next, we consider such neighbouring-items as adjacent nodes to a target user node. For example, consider a user u_0 that has interacted with items i_0 and i_1. Other users that interacted with item i_0 also bought items i_2 and i_3. Then, items i_2 and i_3 are considered as the adjacent nodes to the user node u_0. Afterwards, we use such neighbouring-items as a type of personalised graph prompts and call them as 'hard graph prompts'. Note that we ignore the users' interacted items as adjacent nodes when devising prompts to avoid the possible over-fitting of the final learned model. In addition, we also introduce a different type of personalised graph prompts, namely 'soft graph prompts', which consist of a pre-defined number of randomly initialised embedding vectors. Overall, we define both the hard graph prompts and the soft graph prompts as the personalised graph prompts of a given user. Such personalised graph prompts are leveraged to assist the graph encoder that is pre-trained from the source domain to improve the recommendation effectiveness. In particular, to guide the pre-trained model in learning richer user/item representations, we incorporate SSL-based contrastive learning technique in the pre-training stage. To further enhance the model tuning in the target domain, we combine the contrastive learning loss and the BPR loss (c.f. Equation (2.4)) in a multi-task manner. In the next section, we introduce the rationale and motivation for leveraging the personalised graph prompts in the tuning stage.

7.1.3 Prompting in Transformers and Graph Neural Networks

In this section, we first justify the equivalence between Graph Neural Networks (GNNs) and a transformer to explain the rationale of introducing our personalised graph prompts. Then, we further describe our proposed graph prompts and illustrate them with graph-based recommender systems. Furthermore, we explain the rationale behind the incorporation of graph prompts in top-K cross-domain recommendation, emphasising the benefits and implications of this approach within the context of recommendation systems.

7.1.3.1 Comparing a Transformer and GNNs:

- **Transformer:** The transformer architecture consists of a composition of transformer layers (Vaswani et al., 2017). Each transformer layer has two parts: a self-attention module and a position-wise feed-forward network (FFN). Let h^ℓ denote the input of the self-attention module on a certain layer. The propagation rule of the transformer updates the hidden feature h at position i of a sentence S from layer ℓ to layer $\ell+1$ as follows:

$$h_i^{\ell+1} = \sum_{j \in \mathcal{S}} w_{ij} \left(W_V^\ell h_j^\ell \right) \tag{7.1}$$

$$w_{ij} = \operatorname{softmax}_j \left(W_Q^\ell h_i^\ell \cdot W_K^\ell h_j^\ell \right) \tag{7.2}$$

  where $j \in S$ denotes the set of words in the sentence and W_Q, W_K, W_V are learned linear weights. These weight matrices are used to project the input features (h^ℓ) into the three distinct vectors that define the attention mechanism: the Query (Q), the Key (K), and the Value (V). It is important to note that the 'Query' in this context is an internal, learned vector, distinct from the 'query' in Information Retrieval, which typically refers to user-generated search input. In natural language processing, the transformer sums over all the words in a sentence and outputs the next hidden feature by applying a weighted summation on the values.

- **Graph Neural Network:** Modern GNNs (c.f. Section 2.2) follow a learning schema that iteratively updates the representation of a node by aggregating the representations of its first or higher-order neighbours. Let h^ℓ denote the representation of a node v_i at the l-th layer. Recall from Equation (2.10) in Section 2.2 that GNNs update the hidden features h of node i at layer ℓ via a non-linear transformation combining the node's own features added to the aggregation of features from each neighbouring node $j \in N(i)$:

$$h_i^{\ell+1} = \sigma\left(W_U^\ell h_i^\ell + \sum_{j \in \mathcal{N}(i)} w_{ij}\left(W_V^\ell h_j^\ell\right)\right) \tag{7.3}$$

where W_U, W_V are learned weights of the GNN layer and σ is a non-linear transformation. GNNs exploit high-order connectivity by summing over the feature vectors h_j^ℓ from the adjacent nodes with w_{ij} set to 1 and output the next hidden feature. Given that w_{ij} is a learned weight that is dependent on nodes i and j, the GNN layer can be interpreted as a layer in a single head-based GAT (Veličković et al., 2018) and is similar to the calculation of the hidden features in a transformer layer (see Equation (7.1)).

To conclude, a transformer sums over all words in a sentence and a GNN sums over the local neighbourhood. Moreover, a transformer uses self-attention to have a weighted sum for feature transformation since self-attention can be seen as the inference of a soft adjacency matrix. Indeed, compared to the transformer architecture, a GNN can be considered as a simple transformer that applies a linear-weighted sum on a randomly permuted sentence, since the neighbouring nodes compose a 'sentence' and the 'words' are in random order without the positional embeddings. Inversely, the transformers can also be viewed as a special case of GNNs on a fully connected graph of words (Joshi, 2020).

7.1.3.2 Prompt-tuning in a Transformer and GNNs:

Equation (7.1) and Equation (7.2) describe the dot-product attention mechanism, which provides a weighted summation over the words in a sentence. Here, we derive an equivalent form of Equation (7.1) as follows:

$$\begin{aligned} h_i^{\ell+1} &= \mathrm{Attn}\left(h_i^\ell W_Q, h_j^\ell W_K, h_j^\ell W_V\right) \\ &= \mathrm{Attn}\left(QW_Q, KW_K, VW_V\right) \end{aligned} \tag{7.4}$$

where W_Q, W_K, W_V are learned weights on the layer input Q and the key-value pairs. The mechanism of prompt-tuning changes the attention module through prepending the tunable vectors to the original attention keys and values at every layer (He et al., 2022; Li and Liang, 2021). Specifically, two sets of prefix vectors $P_K, P_V \in \mathbb{R}^{l \times d}$ are concatenated with the original attention key K and value V. Here, we provide an alternative view of prompt-tuning in a transformer:

$$
\begin{aligned}
h_i^{\ell+1} &= \operatorname{Attn}\left(QW_Q, \left(P_K \parallel KW_K\right), \left(P_V \parallel VW_V\right)\right) \\
&= \operatorname{softmax}\left(QW_Q\left(P_K \parallel KW_K\right)^{\top}\begin{bmatrix} P_V \\ VW_V \end{bmatrix}\right) \\
&= (1-\lambda)\operatorname{softmax}\left(QW_Q W_K^{\top} K^{\top}\right) VW_V + \lambda \operatorname{softmax}\left(QW_Q P_K^{\top}\right) P_V \\
&= (1-\lambda)\operatorname{Attn}\left(QW_Q, KW_K, VW_V\right) + \lambda \operatorname{Attn}\left(QW_Q, P_K, P_V\right)
\end{aligned}
\tag{7.5}
$$

where $\parallel$ is the concatenation operator, and λ is a scalar that represents the sum of normalised attention weights on the prefixes:

$$
\lambda = \frac{\sum_i \exp\left(QW_Q P_K^{\top}\right)_i}{\sum_i \exp\left(QW_Q P_K^{\top}\right)_i + \sum_j \exp\left(QW_Q W_K^{\top} K^{\top}\right)_j}
\tag{7.6}
$$

where the index i corresponds to the l prefix vectors, and the index j corresponds to the original input sequence tokens. Note that the first term in Equation (7.5), $\operatorname{Attn}\left(QW_Q, KW_K, VW_V\right)$, is the original attention without prefixes, whereas the second term is the attention modification term, which changes the attention module through linear interpolation on the original output of a transformer:

$$
h_i^{\ell+1} \leftarrow (1-\lambda) h_i^{\ell+1} + \lambda \Delta h_j^{\ell+1}, \quad \Delta h_j^{\ell+1} = \operatorname{softmax}\left(QW_Q P_K^{\top}\right) P_V
\tag{7.7}
$$

where $\Delta h_i^{\ell+1}$ represents the attention output computed only over the learnable prefix vectors P_K and P_V.

Inspired by the success of prompt-tuning methods, which interpolate between an original model's output and a prompt, we adapt this interpolation to the GNN aggregation step for the CDR scenario. We design a new message-passing function that, instead of a single aggregation, computes and fuses two distinct representations from the same set of neighbouring nodes, $N(i)$. The first representation, the 'original' message, is calculated using a standard projection matrix W_V. The second, the 'prompted' message, is calculated using a new, learnable 'prompt' matrix P'_V. These two messages of information are then combined using an item-related weight factor λ. This design, which directly applies the interpolation from prompting to the GNN's message passing, results in an updated propagation rule on top of Equation (7.3):

$$
h_i^{\ell+1} = \sigma\left(W_U h_i^{\ell} + (1-\lambda)\sum_{j\in N_{(i)}}\left(W_V h_j^{\ell}\right) + \lambda \sum_{r\in N_{(i)}}\left(P_V{}' h_r^{\ell}\right)\right)
\tag{7.8}
$$

where r is a neighbouring-items node such as $r \in N(i)$. r is used to act as a new adjacency node of the given node, thereby creating a new edge connection in the graph; $\lambda = \frac{|N_r|}{|N_j|+|N_r|}$ is an item-related weight factor. $|N_j|$ and $|N_r|$ denote the first-hop neighbours item j and the neighbouring-item r, respectively. Similar to W_V, P'_V is a learned weight matrix of the GNN layer. It is worth noting that during the tuning stage, the second term of Equation (7.8), without $(1-\lambda)$, is the output from a pre-trained graph encoder with frozen parameters. Furthermore, the third term of Equation (7.8), specifically the learned matrix P'_V, is the only learned matrix within the context of the target domain D_{T}. The number of tuning parameters is therefore determined solely by the learned embeddings associated with the P'_V matrix and the new hard/soft prompt nodes (r). As a result, tuning only on a small set of parameters is expected to improve efficiency in the tuning stage. By freezing the extensive parameters of the pre-trained graph encoder (W_U and W_V), we ensure the resulting GNN model using the personalised graph prompts is capable of effectively leveraging the useful source domain knowledge while significantly reducing the total number of parameters required for adaptation in the target domain.

7.1.3.3 Hard Prompts and Soft Prompts:

In this section, we provide a comprehensive overview of both hard and soft prompts in our PGPRec framework, detailing their selection, initialisation, and integration within the framework.

Hard Prompts. Informative personalised prompts necessitate the inference of correlations among items. Indeed, the implementation of a straightforward yet efficient approach for determining item correlations is essential for generating personalised hard prompts that enable sufficient learning within a target domain. Following Liu et al. (2021d), we introduce a model-based method to infer correlations between the interacted items and their corresponding neighbouring items within the target domain. Liu et al. (2021d) suggested that a model-based correlation measurement is more desirable than alternative methods such as relying solely on the user frequency counts of the given items. Since the item representations are jointly learned with the graph encoder during the prompt-tuning phase, this correlation score is inherently model-based. In this chapter, we use the dot-product to measure the correlation between items, as it serves as the standard similarity metric in latent factor and graph-based models (He et al., 2017). Let $\mathbf{e_j}$ be the representations of all the $\mathbf{j}$ items that a given user has interacted with, and similarly $\mathbf{e_r}$ denotes the representations of all of the neighbouring-items $\mathbf{r}$ in the target domain. It follows that the model-based correlation score can be defined as follows:

$$\mathrm{Cor_e}(\mathbf{j}, \mathbf{r}) = \mathbf{e_j} \cdot \mathbf{e_r} \tag{7.9}$$

In our proposed PGPRec framework, we select the top-m items from the neighbouring-items $\mathbf{r}$, determined by the correlation score, as hard prompts where m represents the number of hard prompts integrated within the framework. The hard prompts, which originate from the actual nodes within the graph and which are updated independently, directly affect the total number of parameters. As mentioned in Section 7.1.3.2, since the number of these hard prompts (m) is a fixed hyper-parameter, the contribution of the hard prompts to the total model size results

in a fixed and consistent number of tunable parameters. This selection approach ensures that the most relevant and informative neighbouring items are used for generating the hard prompts, thereby enhancing the overall recommendation performance in the target domain.
Soft Prompts. Different from the hard prompts, the soft prompts adopt a flexible and adaptive approach to capture the underlying item relationships within the target domain. In particular, the soft prompts are randomly initialised and subsequently embedded into trainable vectors. These soft prompts serve as auxiliary nodes that establish connections with the target users within the target domain. This process facilitates the collection of contextual information from the user-item interactions, thereby enabling our PGPRec framework to generate more effective user/item representations. Since these embeddings are learned during the tuning stage, the number of tunable parameters increases proportionally to the chosen number of soft prompts. Furthermore, since the soft prompts are trainable, they can adapt to the data during the training process, thereby aligning the user/item embeddings with the underlying structure of the target domain data. As such, this adaptability results in smoother user/item representations, since the soft prompts can better capture more fine-grained representations compared to the hard prompts.

7.1.4 Why Prompt-tuning in GNNs?

We argue that applying prompt-tuning in GNNs offers several potential advantages for cross-domain recommendation tasks, such as improved tuning efficiency, the provision of additional collaborative signals for the target domains, and an easier deployment in large-scale graphs:

- **Tuning efficiency:** Tuning a smaller set of parameters within GNNs using both hard and soft prompts is essential, as it allows for the customisation of the model, enabling it to capture the unique characteristics specific to the target domain. As illustrated in Equation (7.8), prompt-tuning in GNNs leverages pre-trained graph encoders and concentrates on a limited set of parameters by tuning solely with the personalised prompts. Subsequently, the obtained embedding is combined with that generated by the pre-trained graph encoder, therefore leveraging the previously pre-trained knowledge while maintaining the model efficiency.

- **Additional collaborative signals:** Prompt-tuning in GNNs offers a flexible solution capable of incorporating supplementary collaborative signals for the target domain (Fang et al., 2022; Sun et al., 2022). In our proposed PGPRec framework, the personalised graph prompts, which include both the hard and soft prompts, can indeed serve as auxiliary side information for the target domain. These prompts can be tailored to enhance the recommendation performance in specific cases, thereby likely improving the accuracy and effectiveness of the cross-domain recommender system.

- **Easier deployment:** Applying personalised graph prompts on adequately pre-trained

GNNs leads to less parameters to store (Sun et al., 2022), rendering it a more convincing option for large-scale graph applications where the training of GNNs necessitates significant computational resources and memory. In our proposed PGPRec framework, the GNN parameters are frozen after pre-training, allowing only the graph prompts to be tuned (c.f. Section 7.1.3.2). This parameter-efficient strategy ensures PGPRec can potentially be deployed for various applications without the need for extensive training.

To validate these theoretical efficiency and effectiveness claims, we provide a detailed investigation of the parameter efficiency and time efficiency of our proposed graph prompting technique in Section 7.2.6.

7.1.5 Self-supervised Learning (SSL) in PGPRec

Unlike the existing approaches, we adopt the Contrastive Learning (CL) scheme (c.f. Section 2.3.3) within SSL to ensure an effective training in both the pre-training and fine-tuning phases. In the pre-training stage, we employ contrastive learning to optimise the graph encoder with the data from the source domain D_{S}. Specifically, the contrastive pre-training stage comprises three main components: 1) a graph augmentation tool, which builds distinct sub-graphs for generating the representation variants of identical nodes, 2) a graph encoder to model the developed sub-graphs, and 3) a corresponding contrastive loss function for the graph encoder optimisation. On the other hand, to further enhance the recommendation performance in the tuning stage, we also leverage the contrastive learning technique and introduce a joint-learning loss that comprises both the contrastive loss and the BPR loss (c.f. Equation (2.4)). To better illustrate the training process of PGPRec, Algorithm 7.1 provides the training pseudo-code.

7.1.5.1 Contrastive Pre-training of PGPRec

We now describe the contrastive learning component in our PGPRec framework. Following (Wu et al., 2021a), we generate two augmented sub-graphs via the edge dropout technique and feed them into the graph encoder f. As mentioned in Section 7.1.2, PGPRec is a model-agnostic framework that is flexible to accommodate the commonly used GNN models. In this section, we adopt a special case of LightGCN [2] (as introduced in Section 3.1) as the graph encoder, which is a simple yet efficient graph-based recommender (Mao et al., 2021). Specifically, we use the edge dropout augmentation with the best reported performance in SGL (as described in Section 3.1.2), which is an effective graph contrastive learning method. Then, we can obtain

[2] Since the only trainable model parameters in LightGCN are the user/item embeddings at the 0-th layer, LightGCN cannot transfer the structure knowledge with the feature transformation matrices from the source domain. Hence, following (Yang et al., 2022b), we adopt LightGCN with the attention aggregator to explicitly learn transferable knowledge.

two augmented sub-graphs as follows:

$$\mathcal{G}_{S'} = (\mathcal{V}, M \odot \mathcal{E}), \;\; \mathcal{G}_{S''} = (\mathcal{V}, M' \odot \mathcal{E}) \tag{7.10}$$

where $\mathcal{V}$ is the node set, while M and M' are two different masking vectors on the edge set $\mathcal{E}$.

By using the augmented sub-graphs and a graph encoder, we can generate different embedding representations of an identical node (i.e. positive pair). For example, consider a given node q. We can obtain its embeddings e_q and $e_{q'}$ after encoding two sub-graphs with the graph encoder f. Then, to obtain a negative pair of sub-graphs, we apply an in-batch random sampling strategy (Wang et al., 2021b), pairing the target representation e_q with the embedding e_k of a distinct node k. By contrasting the positive and negative pairs, we enable the model to be invariant to local neighbourhood changes, which in turn produce the generalisable and transferable representations required to effectively improve recommendation performance when applied to the target domain. As such, the learned feature embeddings from the source domain allow the model to start from a better initialisation point and provide promising results after a efficient fine-tuning (Gouk et al., 2021) in various target domains.

Next, after obtaining the positive and negative pairs for the user side, we adopt the InfoNCE loss (c.f. Equation (2.11)) to obtain a contrastive loss on user: $\mathcal{L}_{cl}^{user}$. Analogously, we obtain the contrastive loss of the item side $\mathcal{L}_{cl}^{item}$. Combining these two losses, we obtain an objective function for the contrastive task as follows:

$$\mathcal{L}_{cl} = \mathcal{L}_{cl}^{user} + \mathcal{L}_{cl}^{item} \tag{7.11}$$

7.1.5.2 Contrastive Tuning of PGPRec

Once we obtain the pre-trained GNN model from the source domain, we transfer the learned model to the target domain. Recall the discussion in Section 7.1.2 where we introduced two variants of the personalised graph prompt, namely the soft and hard graph prompts in order to improve the recommendation effectiveness in a CDR setting. Differently from the pre-training phase, we optimise both a pairwise ranking task objective and a contrastive learning objective $\mathcal{L}_{cl}$ to further integrate the prediction signals during the tuning phase:

$$\mathcal{L} = \mathcal{L}_{bpr} + \lambda_1 \mathcal{L}_{cl} + \lambda_2 \left\|\Theta\right\|_2^2 \tag{7.12}$$

where $\mathcal{L}_{bpr}$ is a BPR loss as introduced in Section 2.4, λ_1 is the hyper-parameter to control the strengths of the contrastive loss $\mathcal{L}_{cl}$. The $\lambda_2\|\Theta\|_2^2$ is the L2 regularisation to penalise large parameter values in Θ, so as to prevent the model from overfitting.

Algorithm 7.1 The overall training process of the PGPRec framework

```
Input:
        The user-item interaction graphs $\mathcal{G}_S$ and $\mathcal{G}_T$
        The batch size $B$
        The hyper-parameters $\lambda_1, \lambda_2$
        The ID representations of neighbouring-item $\boldsymbol{e_{p_h}}$ as hard prompts
Output:
        The final user/item embeddings $e_u$, $e_i$
Initialise epoch $t = 0$  Initialise the model parameters $\Theta$ with the default Xavier distribution
    // Pre-training parameters in the source domain $\mathcal{D}_S$
repeat
    $t = t + 1$
     Obtain the original ID representations $\boldsymbol{e_u}$, $\boldsymbol{e_i}$ for each user $u$ and item $i$ based on $\mathcal{G}_S$
     Obtain the sub-graphs $\mathcal{G}_{S'}$ and $\mathcal{G}_{S''}$ with a graph perturbation on $\mathcal{G}_S$ with Eq.(7.10)
     Obtain the perturbed user/item representations $e_q$, $e_{q'}$ based on sub-graphs $\mathcal{G}_{S'}$ and $\mathcal{G}_{S''}$ with
     Eq.(7.3)
     Calculate the InfoNCE losses $\mathcal{L}_{cl}^{user}$ and $\mathcal{L}_{cl}^{item}$ with $e_q$ and $e_{q'}$, according to Eq.(2.11)
     Calculate the combined InfoNCE loss $\mathcal{L}_{cl}$ with Eq.(7.11)
     Backpropagation and update model parameters $\Theta$ with $\mathcal{L}$
until convergence;
// Tuning parameters in the target domain $\mathcal{D}_T$
Initialise epoch $t = 0$  repeat
    $t = t + 1$
     Obtain the original ID representations $\boldsymbol{e_u}$, $\boldsymbol{e_i}$ for each user $u$ and item $i$ based on $\mathcal{G}_T$
     Acquire the hard prompt representations $\boldsymbol{e_{p_h}}$ through neighbouring-items
     Acquire the soft prompt representations $\boldsymbol{e_{p_s}}$ with random initialised embedding vectors
     Encode the user/item representations $\boldsymbol{e_u}$, $\boldsymbol{e_i}$ through the pre-trained model parameters $\Theta$
     with Eq. (7.3)
     Combine the embeddings of $\boldsymbol{e_{p_h}}$ and $\boldsymbol{e_{p_s}}$ into the user/item representations $\boldsymbol{e_u}$, $\boldsymbol{e_i}$ with Eq.
     (7.8)  Calculate the BPR loss $\mathcal{L}_{BPR}$ with Eq. (7.12)
     Obtain the perturbed user/item representations $e_q$, $e_{q'}$ based on sub-graphs $\mathcal{G}_{T'}$ and $\mathcal{G}_{T''}$
     with Eq. (7.10)
     Calculate the InfoNCE loss $\mathcal{L}_{cl}$ for contrastive learning with $e_q$ and $e_{q'}$, according to Eq.
     (2.11)
     Calculate the joint learning loss $\mathcal{L}$ with Eq. (7.12)
     Backpropagation and update the model parameters $\Theta$ with $\mathcal{L}$
until convergence;
return the final user/item embeddings $e_u$, $e_i$ in the target domain $\mathcal{D}_T$
```

Table 7.1: Statistics of the used Amazon Review datasets. The column 'Users' donates the number of overlapping users and the column 'Cold-start users' donates the number of cold-start users in the test set.

Dataset	Users	Cold-start users	Items	Interactions	Density
Elec	12,739	2,014	36,183	465,545	0.101%
Phone	12,739	2,014	12,772	178,973	0.110%
Sports	6,755	893	31,514	93,666	0.044%
Cloth	6,755	893	35,131	87,805	0.037%
Sports	4,017	617	19,396	52,202	0.067%
Phone	4,017	617	12,517	40,225	0.080%
Elec	11,611	1,803	49,730	202,095	0.035%
Cloth	11,611	1,803	47,265	137,198	0.025%

7.2 Experiments

To evaluate the effectiveness of our PGPRec framework, we conduct experiments to answer the following four research questions:
RQ7.1: How does the PGPRec framework perform in cross-domain recommendation compared with existing baselines?
RQ7.2: How do different number of hard graph prompts and soft graph prompts and their combination impact the recommendation performance?
RQ7.3: Are the personalised graph prompt-tuning more efficient than fine-tuning in cross-domain recommendation?
RQ7.4: Are the personalised graph prompts effective in a cold-start scenario?

7.2.1 Datasets and Experimental Settings

To evaluate the effectiveness of our PGPRec framework, we use the same Amazon Review datasets as per caption of Table 7.1 described in Section 5.2.3.1, but under a cross-domain setting. Specifically, we conduct experiments on four datasets, namely Electronics (Elec for short), Cell Phones (Phone for short), Accessories, Sports and Outdoors (Sport for short) & Clothing Shoes and Jewelry (Cloth for short). Since the Amazon Review datasets include information from items in different domains, they are suitable for evaluating top-K cross-domain recommendation (Wang et al., 2020d). In particular, these datasets are considered into 4 pairs of source-target datasets as shown in Table 7.1. Each pair of datasets shares common users between the source and target datasets. The exact statistics of the used pairs of datasets are listed in Table 7.1. By comparing the statistics across all datasets in Table 7.1, it is clear that each source-target dataset pair exhibits a similar density level. When processing the datasets, we transform the ratings into 1 or 0, indicating whether the user has rated the item or not. Following (Liu et al., 2020a), we denote those users that have more than 5 interactions in a given dataset as the regular users while the cold-start user are those users with less than five ratings.

Table 7.2: Summary of compared approaches across different aspects.

Method	CMF	CDMF	CoNet	PPGN	BiTGCF	NGCF	LightGCN	SGL	PGPRec
Single-domain	×	×	×	×	×	✓	✓	✓	×
Joint-learning	✓	✓	✓	✓	✓	×	×	×	×
Pre-training & Fine-tuning	×	×	×	×	×	✓	✓	✓	×
Pre-training & Prompt-tuning	×	×	×	×	×	×	×	×	✓

7.2.2 Baselines

To evaluate the effectiveness of our proposed approach, we compare PGPRec with the following existing state-of-the-art baselines. The baselines can be roughly categorised into four groups:

- NGCF$_{SD}$, LightGCN$_{SD}$, and SGL$_{SD}$ are the graph-based models introduced in Section 3.1 (NGCF, LightGCN, SGL). The SD (Single-Domain) notation indicates that we treat them as single-domain baselines trained directly and only on the target domain.

- CMF, CoNet and PPGN are joint-learning approaches for cross-domain recommendation, as discussed in Section 3.3;

- NGCF$_{PT}$, LightGCN$_{PT}$, and SGL$_{PT}$ are graph pre-training and fine-tuning approaches, where PT denotes pre-training and full fine-tuning strategy.

Table 7.2 summarises the baselines as well as PGPRec across different aspects, namely single-domain, joint-learning, and pre-training followed by fine-tuning.

7.2.3 Implementation Details and Hyper-parameter Settings

For a fair comparison between our PGPRec framework and the used baselines, we conduct all experiments on the same machine with a GeForce RTX 2080Ti GPU. Moreover, we use the learned parameters at the pre-training stage to initialise the model parameters at the tuning stage. In the pre-training stage, the dropout rate ρ of nodes and edges are set to 0.1 and the softmax temperature τ in the contrastive learning loss is set to 0.2, which are reported with the best performance of the original SGL paper (Wu et al., 2021a). Furthermore, in the tuning stage, we tune the hyper-parameters of our PGPRec framework on the validation set. The learning rate is selected from $\{10^{-2}, 10^{-3}, 10^{-4}\}$. For those hyper-parameters unique to PGPRec, we tune λ_1, λ_2, τ and ρ within the ranges of $\{0, 0.1, 0.2, ..., 1.0\}$, $\{0, 0.1, 0.2, ..., 1.0\}$, $\{0, 0.1, 0.2, ..., 1.0\}$ and $\{0, 0.1, 0.2, ..., 0.9\}$, respectively. We adopt Recall@K and NDCG@K (c.f. Section 2.4) to evaluate the performance of top-K recommendations in a cross-domain setting. We follow (Simmons-Mackie et al., 2017) setting K = 10 and report the average performance achieved for all users in the test set. Following (He et al., 2020), the dimensions of user and item embedding are set to 64 to achieve a trade-off between performance and time cost, which is determined by a grid search in the range of $\{16, 32, 64, 128, 256\}$. We tune NGCF$_{PT}$, LightGCN$_{PT}$, SGL$_{PT}$

Table 7.3: Experimental results for PGPRec and the used baselines. The best performance is highlighted in bold and the second best result is highlighted with an underline. * and $^{\triangle}$ mean p <0.05 in the t-test and TOST test (AP=0.05) compared to the result of PGPRec with the Holm–Bonferroni correction. SD/PT are the abbreviations for Single-Domain and Pre-Training, respectively.

Dataset	Elec-Phone		Sport-Cloth		Sport-Phone		Elec-Cloth	
Methods	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
CMF	0.4015*	0.2543*	0.4309*	0.2685*	0.3348*	0.1927*	0.3054*	0.1836*
CDMF	0.4457*	0.2612*	0.4561*	0.2874*	0.3808*	0.2242*	0.3524*	0.2356*
CoNet	0.4514*	0.2696*	0.4912*	0.3128*	0.3854*	0.2331*	0.3546*	0.2353*
PPGN	0.4464*	0.2629*	0.4678*	0.2930*	0.3677*	0.2240*	0.3404*	0.2184
BiTGCF	0.5064*	0.2999*	0.5360*	$\underline{0.3315}^{\triangle}$	$\underline{0.4688}^{*}$	0.3072*	0.4209	0.2783*
NGCF$_{SD}$	0.4501*	0.2643*	0.4889*	0.3027*	0.3955*	0.2431*	0.3878*	0.2483*
LightGCN$_{SD}$	0.4776*	0.2993*	0.5327*	**0.3358**$^{\triangle}$	0.4557*	0.3006*	$\underline{0.4261}$	0.2699*
SGL$_{SD}$	0.4722*	0.2941*	0.5296*	0.3219*	0.4486*	0.2749*	0.4012*	0.2569*
NGCF$_{PT}$	0.4665*	0.2969*	0.4983*	0.3253$^{\triangle}$	0.4251*	0.2681*	0.3723*	0.2359*
LightGCN$_{PT}$	0.4821*	0.3044*	0.5196*	0.3268$^{\triangle}$	0.4543*	0.2986*	0.4137	0.2727*
SGL$_{PT}$	$\underline{0.5071}$	$\underline{0.3282}^{\triangle}$	$\underline{0.5398}$	0.3119*	**0.4721**	**0.3114**$^{\triangle}$	**0.4301**$^{\triangle}$	**0.2967***
PGPRec	**0.5213**	**0.3386**	**0.5678**	0.3281	0.4607	$\underline{0.3086}$	0.4233	$\underline{0.2854}$
%Diff.	2.80%	3.17%	3.27%	- 2.29%	- 2.41%	- 0.10%	- 1.58%	- 3.80%

and PGPRec-BPR on the validation set. For the other cross-domain recommendation baselines corresponding to joint-learning approaches (CMF, CDMF, CoNet, PPGN, BiTGCF), we follow the reported optimal parameter settings by the authors of these baselines. For our PGPRec framework, we tune the hyper-parameters (i.e., λ_1, λ_2, τ and ρ) on the validation set. We use the two one-sided equivalence test (TOST) with the Holm–Bonferroni correction to justify the effectiveness equivalence between PGPRec and baselines. Moreover, following SGL, we adopt the Xavier initialisation to initialise all the models' parameters and use the Adam optimiser for the model optimisation with a batch size of 1024. We apply early-stopping during training, terminating the training when the validation loss does not decrease for 50 epochs.

In the following sections, we first compare PGPRec again all used baseline in Section 7.2.4. Next, we analyse the impact on different numbers of soft and hard prompts along with the use of contrastive loss in Section 7.2.5. We then examine the parameter and tuning efficiency of PGPRec compared with the identified strongest baseline in Section 7.2.6. In Section 7.2.7, we further examine PGPRec's effectiveness in transferring knowledge to cold-start users.

7.2.4 PGPRec Effectiveness Evaluation (RQ7.1)

Table 7.3 reports the empirical results of our PGPRec approach in comparison to all of the baselines, which were described in Section 7.2.1. We evaluate our PGPRec framework in comparison to three distinct recommendation approaches: graph-based recommenders, single-domain recommenders and joint-learning recommenders. Specifically, we compare our PGPRec framework to the graph-based recommenders (NGCF, LightGCN, SGL), which are trained in both the single domain (SD for short) setting as well as in the pre-training (PT for short) setting. In addition, we compare our PGPRec framework to the joint-learning approaches (CMF, CDMF, CoNet, PPGN, BiTGCF). Among the evaluated baselines, LightGCN$_{SD}$ generally outperforms the joint-learning CDR approaches, with the exception of BiTGCF. This observation highlights the importance of investigating the nonlinear interaction relationship between users and items through graph neural networks, as mentioned in Section 7.2.1. Notably, BiTGCF is a tailored approach that builds upon LightGCN. The comparison between LightGCN$_{SD}$ and BiTGCF further supports the effectiveness of the graph-based approaches in capturing complex user-item relationships within the cross-domain recommendation context. Furthermore, when comparing SGL$_{PT}$ with NGCF and LightGCN, as well as their single-domain (NGCF$_{SD}$, LightGCN$_{SD}$) and pre-trained variants (NGCF$_{PT}$, LightGCN$_{PT}$), SGL$_{PT}$ exhibits a superior performance in 88% of the cases (28 out of 32 instances), with a significant difference. This result emphasises the advantages of employing contrastive learning during both the pre-training and tuning stages, since it facilitates the generation of richer user representations, ultimately enhancing the recommendation performance. Comparing PGPRec and SGL$_{PT}$, we observe that PGPRec significantly outperforms SGL$_{PT}$ in 3 out of 4 instances, thereby demonstrating the general effectiveness of personalised graph prompts. However, PGPRec shows minor performance degradation compared to SGL$_{PT}$ yet it remains equivalent in 2 out of 4 instances on the Sport-Phone & Elec-Cloth datasets, as confirmed by the TOST test in Table 7.3.. This may be due to PGPRec being more sensitive to differences between domains, such as variations in user-item interaction patterns or preferences, compared to SGL-PT. As a result, such sensitivity may affect the PGPRec's performance when transferring knowledge between distant domains. This observation warrants further investigation to better identify the factors influencing PGPRec's performance in such scenarios. We leave such an investigation to future work.

Hence, in answer to RQ7.1, we conclude that PGPRec successfully leverages the contrastive learning loss to effectively pre-train a graph encoder and further enhances the recommendation performance by leveraging the personalised graph prompts. As a result, our PGPRec framework successfully the SSL-enhanced graph encoder and the personalised graph prompt to enable effective knowledge transfer from the source domain to the target domain.

7.2.5 Ablation Study (RQ7.2)

To investigate the impact of each component of our PGPRec framework, in Table 7.4, we compare the results of PGPRec to its variants that employ different types of personalised prompts (soft/hard) as well as PGPRec variants with varying pre-training loss functions (BPR/contrastive loss). In particular, we aim to examine the impact of these components on the overall performance of the framework. Table 7.4 presents the effect of these components on the overall performance of the framework. In particular, we evaluate the statistical significance of the difference in performance between PGPRec and its variants with the paired t-test ($p < 0.05$). We first compare

Table 7.4: PGPRec performance in terms of Recall@10 and NDCG@10 on the used datasets. The best performance is highlighted in bold and the second best result is highlighted with an underline. * denotes a significant difference compared to the result of PGPRec using the paired t-test with the Holm-Bonferroni correction for $p < 0.05$.

Dataset	Elec-Phone		Sport-Cloth		Sport-Phone		Elec-Cloth	
Methods	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
PGPRec-BPR	0.4851^*	0.3006^*	0.5332^*	0.3149^*	0.4361^*	0.2593^*	0.3918^*	0.2603^*
PGPRec-soft	0.4716^*	0.2845^*	0.5216^*	0.3063^*	0.4468^*	0.2610^*	0.4033^*	0.2662^*
PGPRec-hard	0.5172^*	0.3244^*	0.5532^*	0.3193^*	0.4537^*	0.2856^*	0.4231	0.2814^*
PGPRec	**0.5213**	**0.3386**	**0.5678**	**0.3281**	**0.4607**	**0.3086**	**0.4233**	**0.2854**

PGPRec to its variants with different types of personalised prompts (PGPRec-soft and PGPRec-hard), which sorely use soft and hard prompts, respectively. From the table, we observe that for all four used datasets, PGPRec significantly outperforms PGPRec-soft and PGPRec-hard in 94% of the cases (15 out of 16 instances). This observation indicates the effectiveness of combining both hard and soft prompts for better user/item representations by transferring the knowledge from a source domain and tailoring the user/item representations to a target domain. On the other hand, comparing the variants PGPRec-soft and PGPRec-hard, we observe a noticeable performance gap between the two, with PGPRec-hard outperforming PGPRec-soft on all datasets and metrics. This finding indicates that the neighbouring-items carry more informative content than the randomly initialised learned vectors. Hence, these hard prompts can better assist user representations in adapting to the target domain. This comparison between PGPRec-soft and PGPRec-hard highlights the importance of leveraging the appropriate item-level information, as provided by hard prompts, to enhance the recommendation performance in cross-domain settings. Furthermore, we also compare PGPRec to PGPRec-BPR, which uses the BPR loss as a training loss function during the pre-training stage while PGPRec employs a contrastive loss when pre-training the graph encoder. From Table 7.4, we observe that PGPRec significantly outperforms PGPRec-BPR by a large margin across all used datasets, demonstrating the rationality of incorporating contrastive learning to ensure an effective pre-training. The observed enhanced performance highlights the importance of leveraging contrastive learning to extract valuable knowledge from the source domain, and ultimately benefiting the downstream target domain.

To further investigate the impact of different combinations of personalised prompts, Figure 7.3 shows how the performance of PGPRec changes as we use different numbers and combinations of hard and soft graph prompts. For ease of illustration and visual clarity, in addition to the performance of PGPRec as we vary the number of used hard and soft prompts, the figure only reports the best-performing combination of soft and hard prompts, which was consistently the same across the used datasets. Note that we use Recall@10 to report the performance of different PGPRec variants, since using NDCG@10 also leads to the same conclusions across

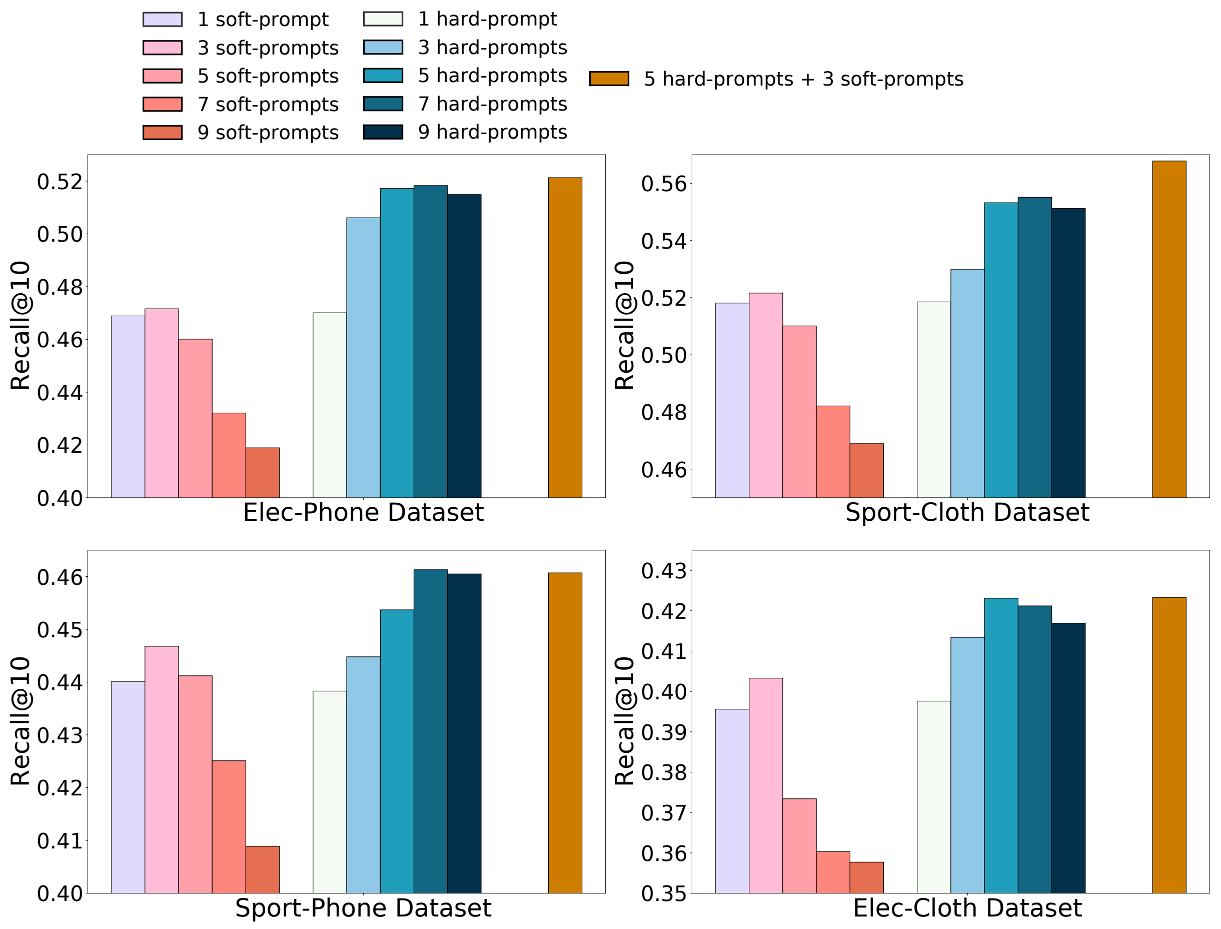


Figure 7.3: Performance of the PGPRec variants in terms of Recall@10 on four Amazon Review datasets.

the used datasets. Figure 7.3 shows that the PGPRec variant (the bar in *orange*) that combines five hard prompt nodes and three soft prompt nodes achieves the best performance in both the Elec-Phone and Elec-Cloth datasets. This promising performance is due to the supplement of the neighbouring-items from the available metadata in the used datasets as well as the use of a continuous vector as a prompt. For the PGPRec variants that only use the hard prompts (bars in *blue*) in Figure 7.3, the performance of PGPRec improves with more neighbouring-items, which can be viewed as if more adjacent nodes of a target node are added/taken into consideration. However, the improvement becomes marginal when the number of hard prompt nodes increases to over five. In particular, we can observe that the performance of the variant with seven hard prompts is on a par with the variant in *orange* on the Sport-Phone and Elec-Cloth datasets, which combines both hard and soft prompts. This further supports that the hard neighbouring-items contribute most in the personalised graph prompts while the contribution of the soft prompts is marginal. By comparing the results observed on the Elec-Phone and Elec-Cloth datasets, we find that the best number of hard prompts depends on the scale of the target domain dataset, which indicates that a larger scale target dataset always requires more prompts to facilitate more divergent user/item representations. Moreover, we observe from Figure 7.3 that the performance of CDR is further enhanced by the addition of soft prompt nodes. One possible reason is that embedding soft prompt nodes provides an additional supervised signal when optimising through contrastive learning. Indeed, through contrastive learning, the collaborative signal is further enhanced by mining graph data in a self-supervised manner. However, as observed in Figure 7.3, a large number of soft prompt nodes impedes the CDR recommendation performance, which shows the difficulty for the graph model to train a useful user/item embedding from scratch.

Hence, in answer to RQ7.2, we conclude that PGPRec successfully uses hard and soft graph prompts as well as the SSL-based contrastive loss for learning effective user/item representations in the target domain. Specifically, the SSL-based contrastive loss effectively optimises PGPRec to produce generalisable representations, thereby improving the top-K cross-domain recommendation performance. Furthermore, the PGPRec-hard variant consistently and significantly outperformed PGPRec-soft, showing that transferring a user's actual neighbourhood data is more effective than learnable, randomly-initialised vectors. Moreover, we investigate the impact of the prompt number. We find that, while performance with hard prompts improves as the number increases, the gains become marginal after approximately five to seven prompts. The optimal configuration was a combination of five hard prompts and three soft prompts, which suggests that hard prompts provide the core transferable knowledge while a small, fixed number of soft prompts offer additional effectiveness in the target domain.

7.2.6 Efficiency of Graph Prompt-tuning (RQ7.3)

To answer RQ7.3, we investigate the superiority of personalised graph prompts in terms of parameter efficiency (Section 7.2.6.1) and tuning efficiency (Section 7.2.6.2).

7.2.6.1 Parameter Efficiency

First, we investigate how many tuned parameters can be reduced by using our personalised graph prompt-tuning on all four used datasets. Specifically, we examine the parameter efficiency by comparing the number of tuned parameters of the personalised graph prompts with the number of fine-tuned parameters of the GNN baselines. In particular, we use the two most effective PGPRec variants as reported in Section 7.2.5 and calculate the number of tuned parameters. Since efficiency and effectiveness are both critical in our study, we first compare the calculated number of tuned parameters in PGPRec as well as in the conventionally fine-tuned GNN baselines (NGCF$_{PT}$, LightGCN$_{PT}$ and SGL$_{PT}$)[3] in Section 7.2.2. Figure 7.4 shows a comparison of the effectiveness and efficiency of the fine-tuned GNN baseline models in comparison to the variants of PGPRec. For conciseness, we use Recall@10 to report the effectiveness of PGPRec and

[3] Recall, that in cross-domain recommendation, all the pre-trained parameters of the graph-based recommenders are fine-tuned in the target domain, while in PGPRec, we only fine-tune the parameters from the personalised graph prompts in the target domain.

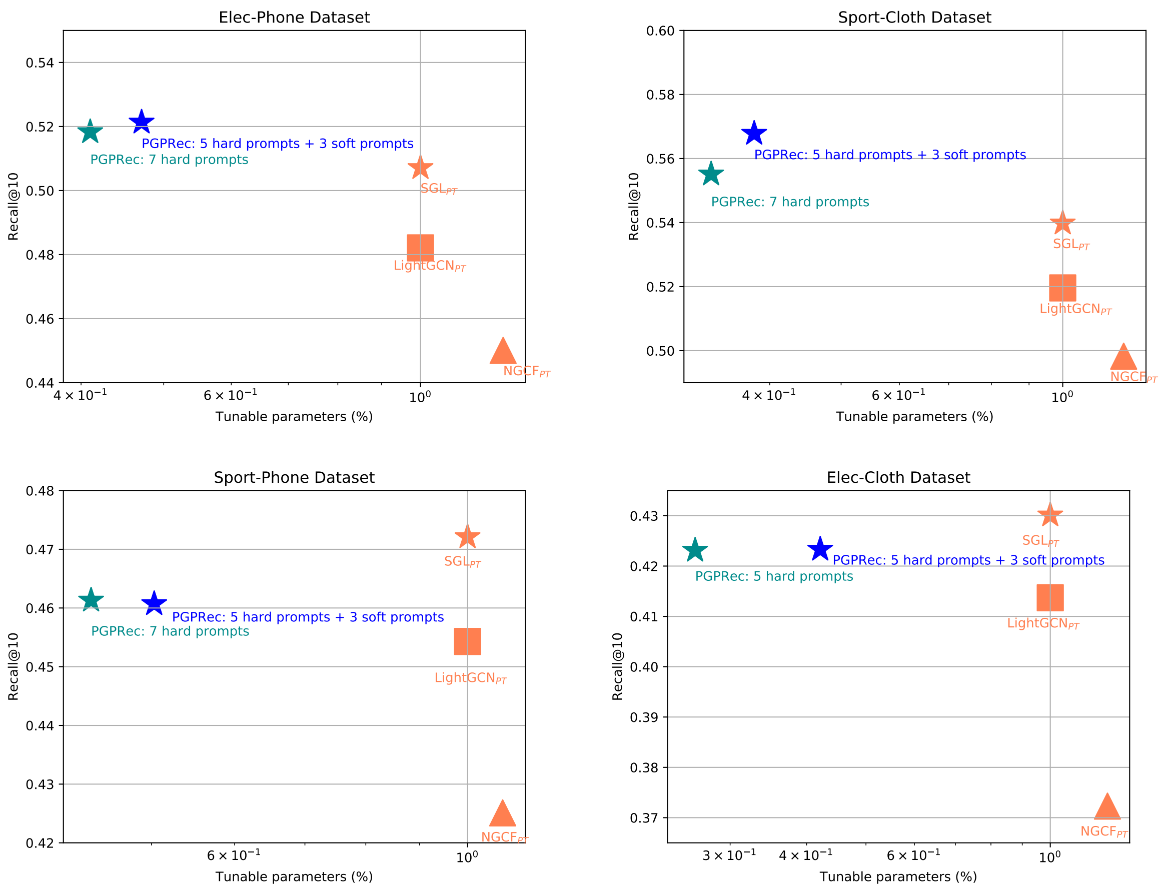


Figure 7.4: Performance of different tuning approaches on the four used datasets with the pre-trained GNN models

the used baselines in Figure 7.4, since the use of NDCG@10 also leads to the same conclusions across the used datasets. As shown in Figure 7.4, LightGCN$_{PT}$ and SGL$_{PT}$ have the same number of tuned parameters but less so than NGCF$_{PT}$, which has additional learned weight matrices compared to LightGCN$_{PT}$ and SGL$_{PT}$ (see Section 7.2.1). Note also that SGL$_{PT}$ has the best effectiveness among the baselines. We observe that the two PGPRec variants need only 41% and 47% of the required fine-tuned parameters of SGL$_{PT}$ to achieve an even higher effectiveness on the Elec-Phone dataset. Similarly, the two PGPRec variants only need 33% and 38% of the tuned parameters required by SGL$_{PT}$ on the Sport-Cloth dataset. Next, we also compare our PGPRec variants with NGCF$_{PT}$. Consider the Elec-Phone dataset as an example. Figure 7.4 shows that the two PGPRec variants need only 34% and 39% of the required fine-tuned parameters of NGCF$_{PT}$, respectively. Similarly on the Sport-Cloth dataset, the two PGPRec variants need 27% and 31% of the fine-tuned parameter required by NGCF$_{PT}$, respectively. Similar conclusions can also be observed on the Sport-Phone and Elec-Cloth datasets, demonstrating the parameter efficiency of PGPRec. Overall, the results in terms of both effectiveness and efficiency on both datasets demonstrate that our personalised graph prompt-tuning is more parameter-efficient than the conventional fine-tuning when transferring pre-trained knowledge to a target domain. Meanwhile, our PGPRec framework can achieve a competitive performance compared to the fine-tuned GNN baselines on both datasets.

To investigate the parameter-efficiency impact of both hard and soft graph prompts within our PGPRec framework, we conduct a comparative analysis on the number of tuned parameters, taking into account the different types of graph prompts used. This comparative analysis encompasses two types of PGPRec variants - one exclusively employs hard prompts (indicated by the red line) while the other uses a combination of hard and soft prompts (indicated by the green line) - as well as the fine-tuned SGL$_{PT}$ baseline. Note that we use SGL$_{PT}$ for comparison in this experiment because it is the most effective baseline. For a fair comparison, both the PGPRec variants and SGL$_{PT}$ use the same GNN encoder (i.e LightGCN). Figure 7.5 shows the results of the two PGPRec variants in comparison to the fine-tuned SGL$_{PT}$ baseline on all four used datasets. From Figure 7.5, we observe that the effectiveness of the variant that uses seven hard prompts (in red) is on a par with the variant that combines five hard prompts with three soft prompts (in green) while reducing the number of tuned parameters on the x-axes of all four used datasets. This result suggests that the hard prompts are overall more efficient in estimating the users' preferences in the target domain. In particular, the PGPRec variant that leverages seven hard prompts only needs 41% and 33% of the tuned parameters of SGL$_{PT}$ to achieve an even better performance on both the Elec-Phone dataset and the Sport-Cloth dataset, respectively. This means that the personalised prompt-tuning allows to reduce 59% and 67% of the parameters in comparison to using SGL$_{PT}$, while also attaining an improved effectiveness. Similarly, the PGPRec variant that uses seven hard prompts can reduce by 54% the number of tuned parameters of SGL$_{PT}$ on the Sport-Phone dataset, and the PGPRec variant that uses five hard prompts can reduce by 74% the number of tuned parameters of SGL$_{PT}$ on the Elec-Cloth dataset, while ensuring a competitive performance on both datasets. To verify that the reduction in the number of tuned parameters does not lead to a significant degradation of effectiveness in comparison to SGL$_{PT}$ on all 4 used datasets, we apply a two one-sided equivalence test (TOST). This test is performed to ascertain an effectiveness equivalence with the SGL$_{PT}$ model. In Figure 7.5, a point marker (corresponding to a green or red dot) indicates that the corresponding performance is equivalent with SGL$_{PT}$ using the TOST test. The TOST results validate our argument that only tuning on the parameters of graph prompts with all other pre-trained parameters remaining unchanged can achieve a competitive performance in comparison to SGL$_{PT}$ on all used datasets.

7.2.6.2 Efficiency of Tuning Time

We now analyse the time efficiency of PGPRec. As described in Section 7.2.3, for a fair comparison, we use one Geforce RTX 2080Ti GPU to conduct the time efficiency experiments. For conciseness, we report the tuning times for PGPRec and the two most effective baselines (LightGCN$_{PT}$ and SGL$_{PT}$) on the Elec-Phone dataset, since the conclusions on the

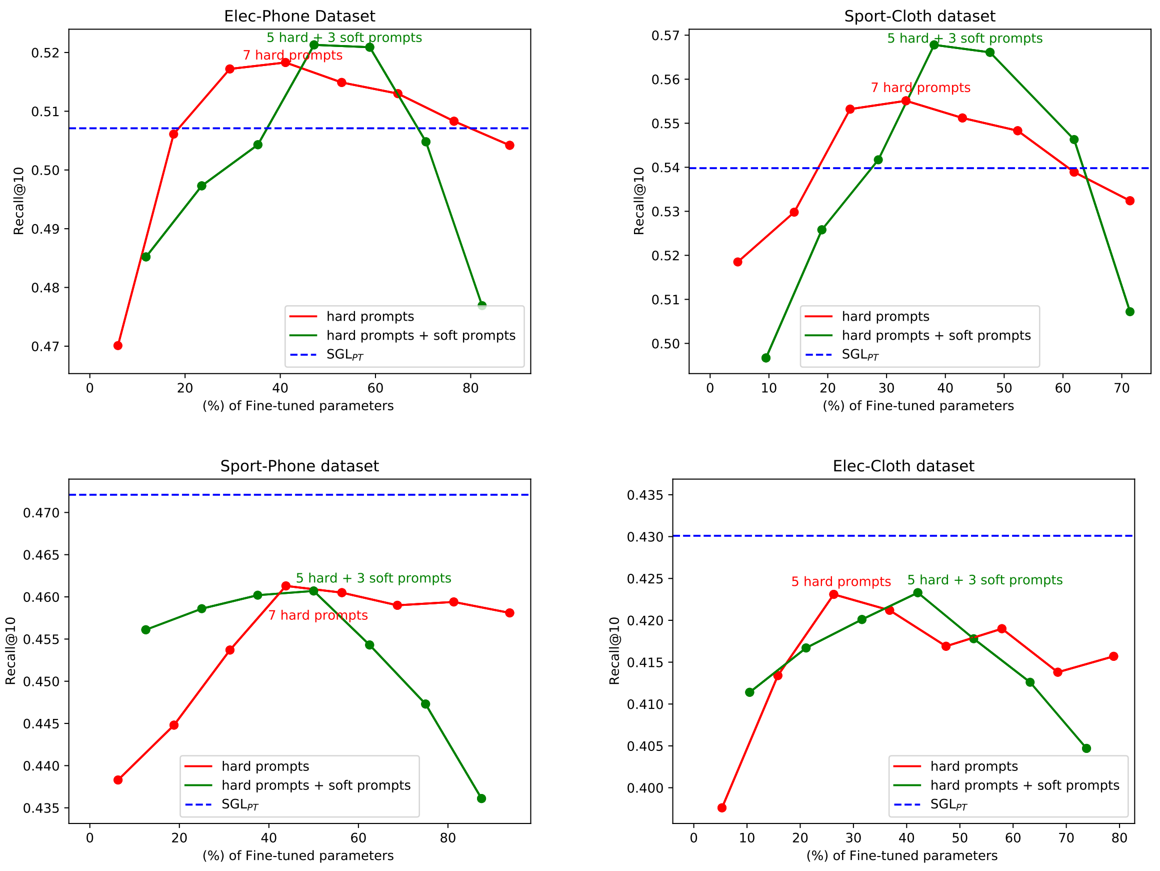


Figure 7.5: A parameter comparison between two types of PGPRec variants: one with only hard prompts and another that combines hard and soft prompts. The peak performances of the two types of PGPRec variants, denoted by the red and green lines respectively, are highlighted with text annotations. A point marker, represented as a red dot for the PGPRec variant with hard prompts and a green dot for the PGPRec variant that combines hard and soft prompts, indicates that the corresponding performance is significantly equivalent over the SGL$_{PT}$ baseline using the TOST test.

other datasets are similar. Table 7.5 compares the efficiency of various PGPRec variants on the Elec-Phone dataset in terms of tuning time. Specifically, we compare the effective PG-PRec variants from Section 7.2.5, namely the variants with seven hard prompts (denoted by PGPRec$_{hard}$), three soft prompts (denoted by PGPRec$_{soft}$) and the mixture of five hard prompts along with three soft prompts (denoted by PGPRec$_{mixture}$) to LightGCN$_{PT}$ and SGL$_{PT}$. Note that we do not compare with NGCF$_{PT}$ as it has additional learned weight matrices compared to LightGCN$_{PT}$ and SGL$_{PT}$. From Table 7.5, we observe that all PGPRec variants have a faster training speed (seconds per epoch) than the LighGCN$_{PT}$ and SGL$_{PT}$ baselines, which indicates the efficiency of using prompts to tune on a small portion of parameters. Moreover, PGPRec$_{hard}$ takes the least training time while achieving a competitive performance compared with SGL$_{PT}$ and PGPRec$_{mixture}$. This demonstrates the efficiency of prompt-tuning compared with conventional fine-tuning as well as the effectiveness of hard prompts in enhancing the pre-trained

Table 7.5: Efficiency comparison on the Elec-Phone Dataset in terms of tuning time. PGPRec$_{hard}$, PGPRec$_{soft}$ and PGPRec$_{mixture}$ refer to the PGPRec variants with hard prompts, soft prompts and mixture prompts, respectively.

Model	Time/Epoch	Nbr of Epochs	Training Time	Recall@10
LightGCN$_{PT}$	63s	294	309m	0.4821
SGL$_{PT}$	68s	138	157m	0.5071
PGPRec$_{mixture}$	57s	423	402m	0.5213
PGPRec$_{soft}$	26s	367	159m	0.4716
PGPRec$_{hard}$	44s	147	108m	0.5172

Table 7.6: Cold-start analysis results for PGPRec and the SGL$_{PT}$ baseline, where PT is the abbreviation for Pre-Training. 'Cold-start' denotes the cold-start users and 'Regular' denotes the regular users in the used datasets. * denotes a significant difference between PGPRec and SGL$_{PT}$ using the paired t-test with p<0.05.

Datasets		PGPRec (Recall@10)	SGL$_{PT}$ (Recall@10)	%Improv.
Elec-Phone	Cold-start	0.4671*	0.4314	8.27%
	Regular	0.7501*	0.7343	2.15%
Sport-Cloth	Cold-start	0.5185*	0.4653	11.41%
	Regular	0.6735*	0.6564	2.60%
Sport-Phone	Cold-start	0.4224*	0.4042	4.5%
	Regular	0.5483	0.5687	-3.59%
Elec-Cloth	Cold-start	0.3876*	0.3764	2.98%
	Regular	0.6192	0.6354	-2.62%

embeddings. However, Table 7.5 also shows that PGPRec$_{soft}$ and PGPRec$_{mixture}$ take more epochs to converge, which indicates the difficulty of the graph encoder (LightGCN) in PGPRec to train useful soft graph prompts from scratch.

To answer RQ7.3, according to Figure 7.4 and Figure 7.5, we conclude that PGPRec empowered by personalised graph prompts is more parameter-efficient than fine-tuned GNNs such as NGCF$_{PT}$, LightGCN$_{PT}$ and SGL$_{PT}$. Moreover, as shown in Table 7.5, we find that only using hard graph prompts helps to further improve the efficiency of tuning time while maintaining a competitive effectiveness.

7.2.7 Cold-start Analysis (RQ7.4)

To investigate the effectiveness of transferring knowledge to the cold-start users using our personalised graph prompts, we examine our PGPRec framework in a cold-start scenario. Specifically, we use the most effective PGPRec variant from Section 7.2.5, namely the PGPRec variant with five hard prompts and three soft prompts, and perform a cold-start analysis on four Amazon Review datasets. Table 7.6 shows the performances of PGPRec for both the cold-start and regular users, in comparison to the best used baseline, SGL$_{PT}$, in terms of Recall@10 (similar trends can be observed with NDCG@10). The results show that PGPRec benefits the cold-start users more than SGL$_{PT}$ and significantly according to the paired t-test on all four datasets. This observation suggests that PGPRec successfully leverages the neighbouring-items and the learned vectors as graph prompts to bring useful information to estimate a user's preferences. In particular, PGPRec outperforms SGL$_{PT}$ by 8.27% and 11.41% on the Elec-Phone dataset and the Sport-Cloth datasets, respectively, thereby demonstrating the successful feature transfer from the source domain and the added-value of personalised graph prompts as additional information. Significant improvements on the cold-start users in comparison to SGL$_{PT}$ are also obtained on the Sport-Phone and Elec-Cloth datasets, although their scale is smaller (4.5% and 2.98%, respectively. This latter observation indicates that the transferred features from the source domain are not sufficiently informative to enhance the prediction of cold-start users' preference in a distant target domain. In fact, given the scale of improvements in Table 7.6, we observe that our PGPRec framework actually benefits the cold-start users more than the regular users. For example, on the Sport-Cloth dataset, PGPRec improves the performance by 11.41% for the cold-start users in comparison to SGL$_{PT}$, while it only improves the performance by just 2.60% for the regular users. This suggests that PGPRec is particularly helpful in cold-start scenarios. This result suggests that the PGPRec framework successfully leverages the personalised graph prompts to act as additional adjacent items to a cold-start user, thereby enriching the representations of users with sparse interactions in the target domain.

Overall, in response to RQ7.4, we conclude that our PGPRec framework successfully leverages the pre-trained user representations and the personalised graph prompts as auxiliary information to effectively enhance the recommendation performance on cold-start users.

7.2.8 Discussion

Chapters 5 and 6 have validated one of the thesis's arguments that graph-based recommender systems can be enhanced by mining self-supervised signals from multiple modalities. In this chapter, we explored an additional and equally valuable direction for achieving further performance gains. We showed that graph-based models can be extended to extract cross-domain supervision signals, thereby improving the transferability and generalisation capabilities of graph-based recommender systems for top- K recommendation. In particular, we used SSL and item-wise graph prompts to facilitate the effective transfer of structural knowledge from the source domain to the target domain.

7.3 Conclusions

In the thesis statement (c.f. Section 1.2), we postulated that the graph-based recommender systems can obtain more effective user/item representations by effectively mining self-supervised signals across domains. Therefore, we proposed personalised and item-wise graph prompts based on relevant items to those items the user has interacted with in the source domain, and used SSL-based pre-training (c.f. Section 7.1.5.1) and SSL-based prompt-tuning (c.f. Section 7.1.5.2) to ensure effective adaptation of graph-based recommenders to the target domain. In particular, PGPRec effectively leveraged the proposed graph prompts to ensure effective knowledge transfer with promising performance from one source domain to the target domain (see Table 7.3). Moreover, we conducted an ablation study investigating different combinations of the personalised graph prompts. As shown in Table 7.4, we showed that the hard prompts make a key and marked contribution in the tuning stage while the soft prompts can provide limited improvement to top-K recommendation in the target domain. Furthermore, in Figure 7.4, we showed that a PGPRec variant with five hard prompts achieves comparable effectiveness to the strong SGL$_{PT}$ baseline, while markedly reducing the number of required tuned parameters by a large margin (e.g., a 74% reduction in the number of tunable parameters of the strongest baseline SGL$_{PT}$ on the Elec-Cloth dataset). By comparing the tuning time of the existing fine-tuned GNNs with that of several PGPRec variants in Table 7.5, we showed that our PGPRec framework (using only hard prompts) tunes faster (e.g., 31% faster on the Elec-Phone dataset) while achieving an improved recommendation effectiveness. We further conducted a cold-start analysis to investigate whether our proposed PGPRec framework benefits the cold-start users. The results obtained on Table 7.6 showed that PGPRec successfully alleviates the cold-start problem, and achieves effective cross-domain recommendations (e.g., an 11.41% performance improvement on the Sport-Cloth dataset in comparison to the strongest baseline SGL$_{PT}$) for cold-start users.

In this chapter, we have validated one of the hypotheses of our thesis statement, namely that graph-based recommender systems can be effectively enhanced by mining self-supervised signals across domains. However, as discussed in Section 3.3, existing cross-domain models are predominantly ID-based approaches. This poses a significant limitation, as they rely heavily on user and item ID identifiers to model common knowledge dependencies across domains. To reduce this reliance on ID-based features and further improve the generalisability of graph-based models, we propose leveraging the multi-modal semantics of items (e.g., text and images) as a source of universal representations. In particular, we explore the use of SSL in enabling multi-modal LLMs to generate highly generalisable representations across multiple domains. We then investigate how to combine the parameters of these MLLMs – trained on different domains – via a weighted average method (i.e., the "model soup" technique), so as to improve the effectiveness of graph-based recommender systems for the top-K recommendation task in a multi-domain setting.

Chapter 8

Self-supervised Multi-domain Universal Representation Learning

In Chapter 7, we proposed the PGPRec approach to leverage SSL-enhanced pre-training and item-wise graph prompts to enhance the domain transfer capabilities in graph-based recommender systems. However, PGPRec shares a fundamental limitation with many graph-based models due to its reliance on ID-based features, which produce abstract representations derived solely from sparse interaction data. This limitation is not unique to cross-domain approaches. As discussed in Section 3.3, existing Multi-domain Recommendation (MDR) approaches (Ma et al., 2018; Tang et al., 2020; Zhang et al., 2022a) also rely on using IDs as user/item features. Different from the cross-domain approaches in Chapter 7 (which transfers from one source to one target), MDR aims to leverage overlapping user interests and item associations from multiple source domains to benefit a target domain, typically using an ID-based universal model. This reliance on IDs might lead to an inadequate item representation learning because of the sparsity of interaction data (Luo et al., 2023; Meng et al., 2024b; Yi et al., 2023b, 2024). This sparsity prevents the MDR models from sharing transferable knowledge between multiple similar domains, thereby constraining the effectiveness of IDs-based models. This leads to user/item representations with limited semantic meaning and high domain dependence, which restricts their transferability compared to multi-modal or semantic-based features. To address this reliance on ID-based features and better mine effective supervision signals across domains, we propose a paradigm shift from ID-based to semantic-based universality. We argue that multi-modal large language models (MLLMs) can enable better generalisation by generating universal representations for the multi-modal items' contents across multiple domains.

On the other hand, with recent advances in computer vision (Kolesnikov et al., 2022) and natural language processing (NLP) (Vaswani et al., 2017), large-scale vision and language models now possess the ability to extract semantic content from items with a high degree of generalisation. Inspired by the recent success of the *model soup* technique (Chronopoulou et al., 2023), we use the expressive power of MLLMs to model the item multi-modal representations in MDR.

This technique improves out-of-domain performance in NLP by averaging the parameters (or "weights") of multiple model configurations that were fine-tuned separately (e.g., on different data or with different settings). To better leverage the diverse but disparate user/item data available in MDR, we propose AdapterSoupRec, a novel approach that applies the model soup technique (Wortsman et al., 2022) to MLLMs (namely, CLIP (Radford et al., 2021) and LLaVA (Liu et al., 2024a)) in order to enhance top-K multi-domain recommendation performance. In particular, we pre-train these 'soup-enhanced' MLLMs across various source domains with their respective losses (an SSL loss for CLIP, a cross-entropy loss for LLaVA), so as to generate more generalisable user/item representations and effective model configurations. We then apply the model soup technique, performing a weighted average on these configurations to obtain the final set of MLLM parameters. This resulting "soup-enhanced" model is then used to encode universal representations across domains. To ensure that the resulting model retains the domain-specific knowledge, we additionally integrate an adapter with the MLLM to effectively enrich the obtained generalised item features, enhancing their adaptability to each target domain. The remainder of this chapter is structured as follows:

- Section 8.1 presents the related work of applying model soup techniques in LLMs;
- Section 8.2 describes the tackled top-K multi-domain task and the methodology of modelling universal representations across domains using soup-enhanced MLLMs;
- Section 8.3 presents the experimental setup and our research questions in this chapter;
- Section 8.5 summarises the key findings presented in this chapter and position them in relation to the thesis statement.

8.1 Model Soups of LLMs

The linear interpolation of neural network weights has recently garnered attention (Alayrac et al., 2022; Bolya et al., 2022; Matena and Raffel, 2022). Such an interpolation results in an improved accuracy across various domains (Jaiswal et al., 2023). The model soup technique, first proposed by Wortsman et al. (Wortsman et al., 2022), includes either a uniform or greedy mechanism for averaging weights of multiple large language models with varying hyper-parameter configurations. The *greedy soup* selectively combines the best-performing model parameters based on the validation set's accuracy, whereas the *uniform soup* averages parameters across various model configurations. These model soup techniques have been shown to significantly improve tasks such as image classification and text summarisation. Later works (Croce et al., 2023; Jaiswal et al., 2023) found that averaging fine-tuned language models improves out-of-domain generalisation in the text summarisation task. The model soup technique involves blending multiple model variants, each variant optimised under different conditions, to generate a more adaptable model ensemble. This model soup technique, which aggregates the best aspects of different configurations, prevents overfitting to specific domains and enhances the used model's ability to generalise to a new domain. Hence, we argue that the model soup technique can be particularly suitable for MDR scenarios, since it helps prevent overfitting within specific source domains and improves the MLLMs' capacity to adapt to a new target domain. In this chapter, to reduce the reliance on ID-based features in graph-based recommender systems (as used in Chapter 7), we first use an SSL loss (c.f. Equation (2.11)) or a cross-entropy (c.f. Equation (2.14)) loss to pre-train two MLLMs, namely (CLIP (Radford et al., 2021) and LLaVA (Liu et al., 2024a)), to obtain diverse model configurations across multiple domains. We then apply the model soup technique to average these model configurations to obtain a final set of model parameters, so as to enable generalisable item representation learning in the MDR task. Additionally, we leverage a graph neural network as an adapter with the MLLM to incorporate personalised information (i.e., user-item interactions), thus retaining the domain-specific knowledge for each target domain. To the best of our knowledge, we propose the first soup-enhanced MLLMs for enabling effective top-K multi-domain recommendations.

8.2 Universal Representations via Soup-enhanced MLLM

In this section, we present our AdapterSoupRec approach, designed for the top-K multi-domain recommendation task in Section 8.2.1. Unlike the single-source cross-domain task in Chapter 7, the multi-domain setting leverages multiple source domains to benefit a target domain. First, we state the multi-domain recommendation problem. Next, in Section 8.2.2, we present how SSL can generate universal item representations using two distinct MLLMs, so as to reduce the reliance on traditional ID-based approaches. Finally, Section 8.2.3 describes AdapterSoupRec with different model soups and the personalised adapter for each target domain.

8.2.1 Task Definition

Given a multi-modal large language model f_θ adapted to n source domains $\{D_1, \ldots, D_n\}$, the Multi-Domain Recommendation (MDR) task aims to return the top-K items for a target user u in a target domain D_{n+1}. Following the same notation in Section 5.1.1.1, we use $m \in \mathcal{M} = \{v, t\}$ as the modality indicator for each domain D, where v and t represent the visual and textual modalities [1] of each item, respectively. In contrast to the ID-based approaches of Chapter 7, which relied on overlapping users to obtain representations ($\mathcal{I}_{id}$, $\mathcal{U}_{id}$), we devise a universal MDR approach to learn multi-modal item representations $\mathcal{I}_m$ and user preferences $\mathcal{U}_m$ from all the domains. This ensures that AdapterSoupRec effectively functions with both overlapping and non-overlapping users, thereby allowing its applicability in diverse domains.

[1] We use these two modalities, namely item images and textual descriptions, since most available datasets have only these two types of raw data.

8.2.2 Universal Multi-modal Item Representation Learning

In this section, we extract universal and shared knowledge from the source domains. Hence, we train a Multi-modal Large Language Model (MLLM), such as CLIP or LLaVA, with multi-modal item contents from the source domains. More specifically, we obtain the textual item representations as follows:

$$x_i^{(l)} = \text{LN}\left(x_i^{(l-1)} + \text{MSA}(x_i^{(l-1)}, x_i^{(l-1)}, m_i)\right), \tag{8.1}$$

where $x_i^{(l)}$ is the hidden state of the i-th item in the l-th layer, while LN (layer normalisation) stablise the hidden states and MSA (Multi-head Self-Attention) captures the contextual relationships between items (Vaswani et al., 2017). m_i corresponds to the textual embedding. Similarly, we obtain the visual item representations. As such, the used MLLM learns the semantic representations of items from multiple source domains, thereby mapping all learned items into a unified space. Following (Liu et al., 2024a; Wang et al., 2023a), we select two configurations of our AdapterSoupRec model: CLIP ViT-B/16 and LLaVA-7b. We encode the item images along with the item descriptions using these two configurations and train each configuration using a loss function that linearly combines the MLLM's own loss function with a BPR loss (see Equation (2.4)): $\mathcal{L} = \mathcal{L}_{bpr} + \lambda_1 \mathcal{L}_{mllm} + \lambda_2 \|\Theta\|_2$, where $\mathcal{L}_{\text{MLLM}}$ corresponds to the MLLM's own training loss (i.e., LLaVA's cross-entropy loss, c.f. Equation (2.14), or CLIP's contrastive ITC loss, c.f. Equation (6.3)), λ_1 is the hyper-parameter to control the strengths of the contrastive loss $\mathcal{L}_{cl}$. The $\lambda_2\|\Theta\|_2^2$ is the L2 regularisation to penalise large parameter values in the set of model's parameters Θ, so as to prevent the model from overfitting. As introduced in Section 2.3, both the ITC loss and the cross-entropy loss are foundational objectives in SSL. Indeed, both losses are widely used in pre-training the LLMs with strong generalisability required for downstream tasks (Gunel et al., 2020). As such, in this pre-training process, we leverage these SSL objectives to generate highly generalisable MLLM configurations. In the next section, we will describe how the model soup technique is applied in AdapterSoupRec to the obtained model configurations pre-trained by SSL, so as to obtain a highly generalisable MLLM for top-K multi-domain recommendation.

8.2.3 AdapterSoupRec: A Model Soup Approach

In this section, we present a two-stage approach for our AdapterSoupRec model. First, we describe how to combine the MLLM configurations using the model soup technique. Second, we describe how to adapt the resulting model to the target domain using a graph adapter.

Algorithm 8.1 Greedy Interpolation Soup Procedure

```
1: **Input**: Configurations of Model Weight $\Theta = \{\theta_1, \ldots, \theta_n\}$
2: # sort the ingredients in decreasing order of validation accuracy $\mathrm{ValAcc}\,(\theta_i)$
3: $\Theta_{\text{sorted}} \leftarrow \mathrm{SORT}_{\mathrm{ValAcc}}(\Theta)$
4: soup $\leftarrow \Theta_{\text{sorted}}[0]$ **for** $i = 1$ *to* $n$ **do**
5:
        **end**
        **if** $\mathrm{ValACC}\,($interpolate(soup, $\Theta_i, \alpha)) \geq \mathrm{ValACC}$ (soup) **then**
6:      soup $\leftarrow$ interpolate(soup, $\Theta_i, \alpha$ )
7: **Return** soup
```

8.2.3.1 Model Soups

We now describe the recipes for the uniform and greedy soups using the obtained MLLMs' parameters. After pre-training with the multi-modal item contents from n source domains, we obtain an MLLM model $f(x, \theta)$ with data x and parameters $\theta \in \mathbb{R}^d$. Let $\theta =$ PreTrain (θ_0, p) denote the parameters obtained with a pre-trained initialisation θ_0 and a hyper-parameter configuration p. The hyper-parameter configuration includes the learning rate, training iterations, and a random seed, which determine the data order (Wortsman et al., 2022). To obtain multiple configurations of MLLMs' parameters, we define $\theta_i =$ PreTrain (θ_0, p_i) where each p_i denotes a different hyper-parameter configuration. In traditional machine learning practices, we select the parameters θ_i that achieve the highest accuracy on a held-out validation set and discard the remaining parameters (Wortsman et al., 2022). Instead, the model soup technique averages the MLLM's configurations with all parameters θ_i – the so-called uniform soup – as follows:

$$\mathrm{AdapterSoupRec}(x) = f\left(x, \frac{1}{n}\sum_{i=1}^{n}\theta_i\right), \tag{8.2}$$

where n is the number of the MLLM's configurations, and x is the resulting user/item embedding. As such, we obtain an MLLM by interpolating different configurations of the MLLM's parameters with the original model size. Another recipe for enhancing the model soup involves creating a greedy soup by sequentially adding the MLLM with different parameters as potential ingredients. Algorithm 8.1 presents the procedure of the greedy soup. The greedy soup only retains the configurations that show an improved performance on a held-out validation set in the soup, which is disjoint from the training and test sets: $\mathrm{AdapterSoupRec}(x) = f\left(x, \frac{1}{n'}\sum_{i=1}^{n'}\theta_i\right)$, where n' is the number of selected MLLM's parameters to obtain a new set of parameters for an MLLM in the target domain and $n' <= n$. If $n' = n$, this model becomes akin to a uniform soup. As such, the greedy soup prevents overfitting to specific source domains and enhances the used model's ability to generalise to a new target domain by using diverse configurations of MLLM parameters.

8.2.3.2 Personalised Adapter

Adapters (Chronopoulou et al., 2023; Gururangan et al., 2020) and mixture of experts (Chronopoulou et al., 2022; Stickland et al., 2021) have been shown by (Diao et al., 2023) to be effective in enhancing domain adaptation. In this work, we enhance domain adaptation by adding an adapter into the used MLLM in each target domain. Specifically, we use a lightweight graph neural network (He et al., 2020) as a graph adapter to incorporate the user-item interactions in a target domain. This graph adapter enables the MLLM to learn domain-specific knowledge, thereby enriching the multi-modal item features produced by the MLLM. We define a graph adapter with parameters θ_α and fine-tune these parameters in a target domain. Notably, we only fine-tune the adapters' parameters θ_α without updating the MLLM's parameter θ, ensuring an efficient adaption to the target domain. We leverage the resulting graph adapter to tailor the MLLM for a target domain by aggregating the multi-modal features obtained from the MLLM:

$$x_{u-m}^{(l_\alpha)} = \sum_{i \in \mathcal{N}_u} \frac{x_{i-m}^{(l_\alpha - 1)}}{\sqrt{|\mathcal{N}_i| \, |\mathcal{N}_u|}}, \tag{8.3}$$

where l_α is the number of layers. $\mathcal{N}_i$ and $\mathcal{N}_u$ denote the set of neighbours for user u and item i, respectively, while $|\mathcal{N}_u|$ and $|\mathcal{N}_i|$ represent the size of $\mathcal{N}_u$ and $\mathcal{N}_i$, respectively. As a result, we aggregate the multi-modal features according to the interactions between the users and items, thereby incorporating personalised information in the obtained features from the used MLLM. We optimise the graph adapter using the BPR loss (c.f. Equation (2.4)) to effectively capture domain-specific knowledge.

Once we obtain both visual and textual user embeddings from the graph adapter for the target domain, we then concatenate them for the later top-K multi-domain recommendation task:

$$x_{u-vt} = x_{u-v} \parallel x_{i-t}, \tag{8.4}$$

where $\parallel$ represents the concatenation operation. Analogously, we obtain the multi-modal embeddings of the items. By applying a graph adapter on top of the MLLM, we incorporate domain-specific knowledge from the user-item interactions of the target domain, thereby enhancing the recommendation accuracy.

8.3 Experiments

To demonstrate the effectiveness of our AdapterSoupRec approach, and illustrate the reasons for its effectiveness, we conduct experiments to answer the following research questions:
RQ8.1: Does AdapterSoupRec outperform the baselines in top-K multi-domain recommendation, and which variant – MLLMs, uniform or greedy soup – is more effective?
RQ8.2: What is the performance impact of our AdapterSoupRec model's key components – the

Table 8.1: Statistics of the used datasets.

Datasets	#Users	#Items	#Interactions	Sparsity
Food	115,349	39,670	$1,027,413$	99.99%
Home	731,913	185,552	$6,451,926$	99.99%
Clothing	39,387	23,033	237,488	99.97%
Office	87,436	25,986	684,837	99.97%
Pantry	13,101	4,898	126,962	99.80%
Electronics	192,403	63,001	$1,689,188$	99.99%
Sports	87,436	25,986	684,837	99.97%

model soup method and the graph adapter – along with its pre-training strategy?
RQ8.3: Do the uniform and greedy model soup techniques enhance multi-modal fusion in item representations more effectively than non-soup variants of AdapterSoupRec?
RQ8.4: Does AdapterSoupRec demonstrate a superior out-of-domain performance compared to the best-performing multi-modal baselines?

8.3.1 Datasets and Experimental Settings

To evaluate our AdapterSoupRec model, we conduct experiments on the same Amazon Review datasets introduced in Section 7.2.1, but under a multi-domain setting involving seven domains. Table 8.1 shows the statistics of the used datasets. Specifically, we pre-train the Food, Home, Clothing, and Office datasets as the source domain datasets and evaluate the effectiveness of multi-domain recommendation on the Pantry, Electronics and Sports datasets as target datasets. For each dataset in the source domains, we pre-train the MLLMs using the URLs of the items' images as well as the detailed items' descriptions including their titles, categories and brands.

8.3.2 Baselines

We compare our proposed AdapterSoupRec [2] approach with three major groups of recommendation baselines:

- Multi-modal methods: VBPR, MMGCL and BM3 (c.f. Section 3.2);
- Cross-domain methods: PPGN and BiTGCF (c.f. in Section 3.3);
- Multi-domain methods: **MOME** (Ma et al., 2018) uses multiple expert networks and a gating network that selects a relevant subset of experts for each target recommendation domain, thereby enhancing the recommendation accuracy in those target domains; **PLE** (Tang et al., 2020) distinguishes between the task-shared and task-specific experts and uses a progressive routing mechanism to dynamically route the target domain recommendations through the appropriate experts; **MGFN** (Zhang et al., 2022a) uses Graph

[2] Our source code is available at: `https://github.com/zxy-ml84/SoupRec`

Attention Networks to learn both intra-domain and inter-domain knowledge, enhancing the model's ability to facilitate recommendations across multiple domains.

8.3.3 Evaluation Protocol and Hyper-parameter Settings

We randomly split the given datasets into training, validation, and testing sets with a 7:1:2 ratio. We adopt Recall@K and NDCG@K (c.f. Section 2.4) to evaluate the performance of top-K multi-domain recommendation. Following (Yu et al., 2022), we set K = 20 and report the average performance achieved for all users in the testing set. To ensure a fair comparison between AdapterSoupRec and the baselines, we use the Xavier initialisation and the Adam optimiser with a batch size of 2048 for all tested models. We apply a grid search to tune the hyper-parameters on both the AdapterSoupRec's CLIP variant (i.e., AdapterSoupRec (CLIP)) and all used baselines on the validation set, with the learning rate ranging within $\{10^{-2}, 10^{-3}..., 10^{-6}\}$ and the batch size varying within $\{32, 64, ..., 512, 1024\}$. For tuning the hyper-parameters specific to the AdapterSoupRec's LLaVA variant (i.e., AdapterSoupRec (LLaVA)), which configures a Large Language Model (e.g., Llama) with a higher number of specific parameters that are not used in the baseline models, we use the following settings: we used a batch size of 8, a token length of 2048, varied the learning rate within $\{0.001, 0.002, ..., 0.005\}$, the number of epochs within $\{1, 2, 3\}$, the warm-up ratio within $\{0.01, 0.02, ..., 0.10\}$ and the weight decay within $\{0, 0.01, ..., 0.05\}$. In addition, we use the Multi-Layer Perceptrons to reduce the dimensionality of the MLLMs' multi-modal item embeddings from 768 to 128, so as to match the multi-modal baselines, thereby ensuring a fair comparison.

In the following sections, we compare AdapterSoupRec against all baselines (Section 8.3.4), and analyse the contribution of each component within AdapterSoupRec (Section 8.3.5). We further examine the alignment of intermediate visual and textual embeddings to indicate a better universal representation for top-K multi-domain recommendation performance (Section 8.3.6). In addition, we further evaluate AdapterSoupRec's out-of-domain performance with generalisation study in Section 8.3.7.

8.3.4 Performance Comparison (RQ8.1)

We compare our proposed AdapterSoupRec method with all the baselines. Table 8.2 presents the comparison results. In Table 8.2, "Uni" and "Grd" denote the uniform and greedy soup, respectively. From the observed results on the three datasets, we observe that all AdapterSoupRec variants (CLIP-Uni, CLIP-Grd, LLaVA-Uni, LLaVA-Grd) significantly outperform all the baselines in most instances (except in 2 out of 54 instances) on both used metrics as confirmed by a paired t-test with the Holm-Bonferroni correction. These results indicate that AdapterSoupRec is a promising approach for enabling an effective transfer of both domain-specific and common shared knowledge in MDR. Furthermore, as shown in Table 8.2, the CLIP-based variants achieve

Table 8.2: Comparison of AdapterSoupRec (greedy) with the baselines. * denotes a significant difference with a baseline using the corrected paired t-test with $p < 0.05$.

Dataset	Pantry		Electronics		Sports	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
VBPR	0.0723^*	0.0326^*	0.0442^*	0.0196^*	0.0771^*	0.0349^*
MMGCL	0.0907^*	0.0377^*	0.0627^*	0.0304^*	0.0913^*	0.0428^*
LATTICE	0.0918^*	0.0409^*	0.0618^*	0.0296^*	0.0944^*	0.0424^*
BM3	0.0932^*	0.0417^*	0.0638	0.0310^*	0.0970^*	0.0438^*
PPGN	0.0774^*	0.0341^*	0.0425^*	0.0173^*	0.0836^*	0.0366^*
BiTGCF	0.0823^*	0.0360^*	0.0552^*	0.0247^*	0.0870^*	0.0383^*
MOME	0.0797^*	0.0352^*	0.0573^*	0.0261^*	0.0749^*	0.0318^*
PLE	0.0862^*	0.0384^*	0.0595^*	0.0278^*	0.0866^*	0.0367^*
MGFN	0.0891^*	0.0413^*	0.0623^*	0.0305^*	0.0894^*	0.0383^*
AdapterSoupRec (CLIP-Uni)	0.1021^*	0.0441^*	0.0633^*	0.0304^*	0.0975^*	0.0450^*
AdapterSoupRec (LLaVA-Uni)	0.1061^*	$\underline{0.0455}^*$	0.0642	0.0317^*	$\underline{0.1023}^*$	$\underline{0.0464}^*$
AdapterSoupRec (CLIP-Grd)	$\underline{0.1074}^*$	0.0452^*	$\underline{0.0646}$	$\underline{0.0319}^*$	0.0985^*	0.0461^*
AdapterSoupRec (LLaVA-Grd)	**0.1103**	**0.0477**	**0.0656**	**0.0328**	**0.1068**	**0.0480**

performance comparable to their LLaVA-based variants. This indicates that an SSL-enhanced MLLM provides similar performance benefits compared to a cross entropy-trained MLLM, further validating the usefulness of SSL in top-K multi-domain recommendation. Comparing the multi-modal baselines (VBPR, MMGCL, LATTICE, BM3), which combine multi-modal features with ID features against the cross-domain baselines (PPGN, BiTGCF) that only use IDs, we observe that the former group of models incorporating multi-modal item contents are overall more effective, suggesting a more successful knowledge transfer. Notably, AdapterSoupRec only use multi-modal features without IDs, simplifying the model structure by avoiding the complexity of aligning multi-modal and ID embeddings. These approaches consistently demonstrate a superior performance against the multi-modal baselines. We also evaluate the performance of the multi-domain methods (MOME, PLE, MMGFN, AdapterSoupRec) in comparison to the cross-domain methods. The improved performance observed in Table 8.2 for the multi-domain methods indicates that training on multiple source domains, as opposed to one source domain in cross-domain setting, indeed enhances the top-K recommendation performance on the target datasets. Comparing all the AdapterSoupRec variants in Table 8.2, we observe that the AdapterSoupRec (LLaVA) variants outperform the CLIP variants by a large margin across all used datasets. These results suggest that on the used datasets, LLaVA is more effective than CLIP as an MLLM for modelling the universal user/item multi-modal representations in the MDR task. Table 8.2 also shows that the greedy variants of AdapterSoupRec, i.e., LLaVA-Grd and CLIP-Grd, outperform all their uniform (Uni) variants on the used datasets. These results suggest that the greedy soup, which selectively interpolates effective weights based on the recommendation accuracy on the validation set, is a better choice for the MDR task on the used datasets. This is also consistent with the model soup's observed generalisation in the text summarisation task (Jaiswal et al., 2023).

Table 8.3: Results for ablating the key components of CLIP/LLaVA AdapterSoupRec. * indicates a significant difference using a paired t-test with $p < 0.05$.

Dataset	Pantry		Electronics		Sports	
Variants	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
AdapterSoupRec (CLIP-Uni) w/o ModelSoup	0.0987*	0.0421*	0.0620	0.0287*	0.0954*	0.0417*
AdapterSoupRec (CLIP-Uni) w/o Adapter	0.0794*	0.0336*	0.0521*	0.0223*	0.0817*	0.0357*
AdapterSoupRec (CLIP-Uni) w/o PreTrain	0.0940*	0.0426*	0.0618*	0.0292	0.0890*	0.0383*
AdapterSoupRec (CLIP-Uni)	**0.1021**	**0.0441**	**0.0633**	**0.0304**	**0.0975**	**0.0450**
AdapterSoupRec (CLIP-Grd) w/o ModelSoup	0.0946*	0.0434*	0.0624*	0.0306	0.0891*	0.0389*
AdapterSoupRec (CLIP-Grd) w/o Adapter	0.0823*	0.0350*	0.0534*	0.0232*	0.0803*	0.0323*
AdapterSoupRec (CLIP-Grd) w/o PreTrain	0.0914*	0.0398*	0.0621*	0.0303*	0.0883*	0.0386*
AdapterSoupRec (CLIP-Grd)	**0.1074**	**0.0452**	**0.0646**	**0.0319**	**0.0924**	**0.0410**
AdapterSoupRec (LLaVA-Uni) w/o ModelSoup	0.1042*	0.0438*	0.0624*	0.0303	0.0975*	0.0442*
AdapterSoupRec (LLaVA-Uni) w/o Adapter	0.0806*	0.0330*	0.0534*	0.0245*	0.0841*	0.0368*
AdapterSoupRec (LLaVA-Uni) w/o PreTrain	0.0917*	0.0396*	0.0626*	0.0293*	0.0968*	0.0440*
AdapterSoupRec (LLaVA-Uni)	**0.1061**	**0.0455**	**0.0642**	**0.0317**	**0.1023**	**0.0464***
AdapterSoupRec (LLaVA-Grd) w/o ModelSoup	0.0971*	0.0443*	0.0634*	0.0311	0.0942*	0.0421*
AdapterSoupRec (LLaVA-Grd) w/o Adapter	0.0814*	0.0336*	0.0552*	0.0265*	0.0824*	0.0354*
AdapterSoupRec (LLaVA-Grd) w/o PreTrain	0.0933*	0.0414*	0.0610*	0.0288*	0.0945*	0.0426*
AdapterSoupRec (LLaVA-Grd)	**0.1103**	**0.0477**	**0.0656**	**0.0328**	**0.1068**	**0.0461**

In answer to RQ8.1, we conclude that our AdapterSoupRec approach, which includes a soup-enhanced MLLM and a graph adapter for each target domain, is overall promising in enabling an effective knowledge transfer across all target domains. In particular, we found that an SSL-trained MLLM (CLIP) shows comparable effectiveness to a cross-entropy-trained MLLM (LLaVA) when used along with AdapterSoupRec. Moreover, we showed that the greedy soup is the most effective model soup method for MDR on the used datasets.

8.3.5 Ablation Study (RQ8.2)

In this section, we conduct an ablation study to assess the impact of the main components in our AdapterSoupRec method, namely the model soup (ModelSoup) and the adapter (Adaptor). We also evaluate the SSL-based pre-training strategy (PreTrain), which refers to pre-training AdapterSoupRec with multi-modal item contents from the source datasets, using SSL objectives such as the ITC and cross-entropy losses. We assess the performance of each component within four AdapterSoupRec variants (CLIP-Uni, CLIP-Grd, LLaVA-Uni and LLaVA-Grd). First, to gauge the effectiveness of the model soup technique using MLLM, we conduct a comparative analysis by removing the model soup procedure from each AdapterSoupRec's CLIP or LLaVA variant, in both their 'Uni' and 'Grd' configurations. From Table 8.3, we observe that all AdapterSoupRec variants (CLIP-Uni, CLIP-Grd, LLaVA-Uni and LLaVA- Grd) significantly outperform their "w/o ModelSoup" variants, where the model soup technique has been removed, in 20 out of 24 instances. This result confirms that blending the parameters of multiple MLLMs results in an improved multi-domain recommendation performance. Next, we ablate the adapter in AdapterSoupRec, so as to examine its usefulness. By comparing the CLIP/LLaVA variants in both their uniform (Uni) and greedy (Grd) configurations with their corresponding variants without the adapter (w/o Adapter) in Table 8.3, we observe that there is a significant performance

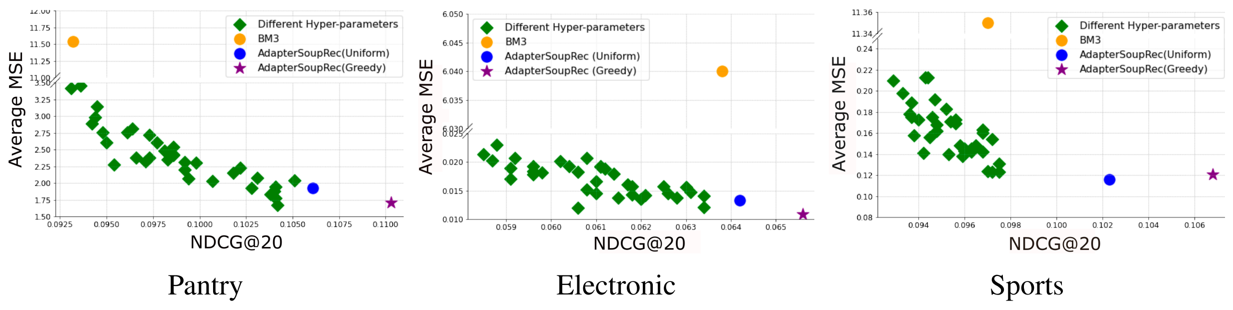


Figure 8.1: Effectiveness of the AdapterSoupRec model compared to multi-modal recommendation baselines.

decrease when removing the graph adapter. This confirms the necessity of including personalised information such as user-item interactions in order to enable domain-specific knowledge in the target domain. Moreover, Table 8.3 shows that pre-training these MLLMs on recommendation data enables a more effective item modelling than their variants without pre-training (w/o PreTrain). Such an observation indicates the MLLMs acquire strong generalisability for the target domains when trained with SSL on multiple source domain datasets.

Overall, in response to RQ8.2, we conclude that our AdapterSoupRec method successfully leverages each of its two key components – the model soup and graph adapter. In addition, SSL-based pre-training is an effective strategy that enhances the generalisability of AdapterSoupRec across multiple domains.

8.3.6 Multi-modal Fusion (RQ8.3)

We have already shown in Table 8.3 that the model soup variants outperform the non-model soup variants (i.e., their single hyper-parameter configurations). In this section, we analyse whether the improved performance is reflected in the quality of features produced by the model soup techniques (i.e., uniform soup and greedy soup) in comparison to those single hyper-parameter configuration variants. To assess the quality of the generated item features, we measure the average Mean Squared Error (MSE) between the visual and textual embeddings produced by the AdapterSoupRec variants. This metric allows us to assess the quality of the item embeddings, with smaller MSE values of visual and textual features (Yi and Ounis, 2024) indicating that the corresponding model enables an effective fusion of item representations. Figure 8.1 shows the MSE values and NDCG@20 scores of each AdapterSoupRec variant. From Figure 8.1, we observe that a lower average MSE value (y-axis) always leads to a better recommendation performance (x-axis) for all AdapterSoupRec variants across all three used datasets. In particular, AdapterSoupRec (Greedy) achieves the best recommendation performance while maintaining a lower average MSE value (e.g., 1.71 on the Pantry dataset). AdapterSoupRec (Uniform) also exhibits comparable recommendation performance with lower average MSE values across all datasets used. Moreover, most AdapterSoupRec variants outperform the best-performing baseline BM3 with markedly lower MSE values (e.g., 11.54 on the Pantry dataset). These consistent performances of AdapterSoupRec (Greedy) and AdapterSoupRec (Uniform) indicate that our proposed model soup techniques effectively integrate multi-modal features into item representations, thereby improving performance in the top-K recommendation task.

In response to RQ8.3, we conclude that the model soup techniques, namely uniform and greedy soups, result in a lower average MSE value between the MLLM's generated visual and textual embeddings compared to the best-performing baseline, BM3, thereby improving the top-K recommendation performance.

8.3.7 Out-of-domain Performance (RQ8.4)

In this section, we evaluate our AdapterSoupRec model's out-of-domain performance, particularly its top-K recommendation performance in another recommendation scenario. This analysis allows us to investigate the generalisability of AdapterSoupRec by training only on previously used Amazon e-commerce datasets and performing inference on other domain datasets, specifically video recommendation datasets. In particular, we perform this generalisability study of AdapterSoupRec on three video recommendation datasets, namely Douyin, Bilibili and Kuaishou, where each video is associated with a cover image and its video description. We compare the performance of AdapterSoupRec (LLaVA-Grd), our best-performing variant as shown in Table 8.2, against multi-modal recommendation baselines, which are selected due to their demonstrated stronger generalisation in the cross-domain recommendation task (Liu et al., 2024b). Our aim is to determine if our AdapterSoupRec model trained on e-commerce datasets still has a comparable recommendation performance on another domain. Figure 8.2 reports the NDCG@20 performance of both AdapterSoupRec and the multi-modal baselines since we observe the same conclusion using Recall@20 across all three video recommendation datasets. From Figure 8.2, we observe that AdapterSoupRec outperforms all the multi-modal recommendation baselines on both the Douyin and Kuaishou datasets while maintaining a comparable performance on the Bilibili dataset. This result indicates that our AdapterSoupRec model effectively generalises the multi-modal knowledge from an e-commerce recommendation scenario to a video recommendation scenario. This generalisability is attributed to the MLLM's training on a large web corpus and our model souping strategy in adapting the MLLM to the top-K multi-modal recommendation task.

In response to RQ8.4, we conclude that AdapterSoupRec effectively generalises from the e-commerce scenario to the video recommendation scenario when compared to the multi-modal recommendation baselines.

8.4 Discussion

In this section, we have demonstrated that our AdapterSoupRec model, a soup-enhanced MLLM trained with SSL, can generate effective universal representations across multiple domains.

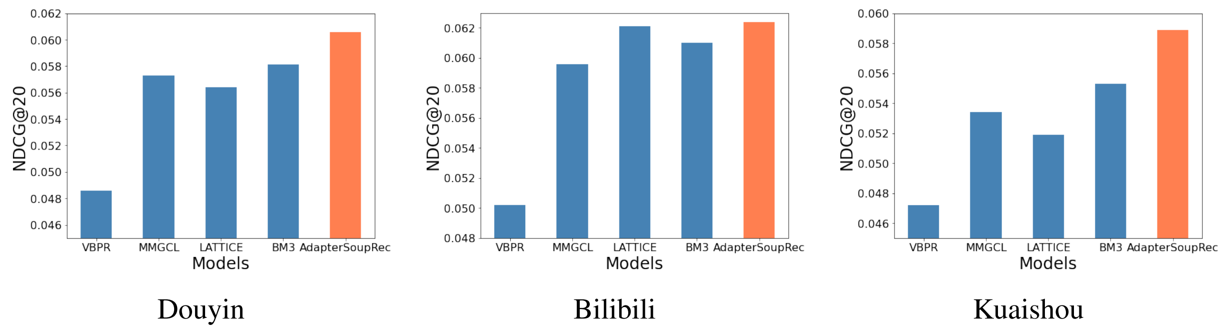


Figure 8.2: Effectiveness of the AdapterSoupRec model compared with multi-modal recommendation baselines.

This approach reduces the reliance of ID-based features in graph-based recommender systems in Chapter 7 by using multi-modal semantics to model universal representations, thereby improving the top-K recommendation in a multi-domain setting. In addition, using multi-modal data (such as item descriptions and images) overcomes the reliance on the overlapping users paradigm (c.f. Section 3.3) – a critical requirement of the ID-based approaches in Chapter 7. This shift from ID-based to multi-modal semantic-based transfer presents a new paradigm for top-K multi-domain recommendation. As a result, we have validated one of the arguments in our thesis statement that graph-based recommender systems can be enhanced by mining richer supervision signals from multiple domains through universal representation learning.

8.5 Conclusions

In this chapter, we address the limitations of the ID-based approach from Chapter 7. Such approaches rely on abstract representations learned from sparse interaction data, which have limited semantic meaning and high domain dependence, thereby restricting their transferability. We introduced the AdapterSoupRec model, which leverages SSL to ensure multi-modal semantic representations are generalisable, thereby enhancing transferability across multiple recommendation domains. Specifically, AdapterSoupRec integrates an MLLM with a graph adaptor and leverages the "model soup" technique to improve top-K recommendation performance in a multi-domain setting. This integration enables an effective architecture that encodes universal representations across multiple source domains and can adapt the domain-invariant knowledge to the target domain. This chapter demonstrated a further step in using SSL to enhance effective knowledge transfer within graph-based recommender systems across multiple domains, providing additional empirical support for our thesis that leveraging richer supervision signals from multiple domains can enhance the performance of graph-based recommender systems. Our extensive experiments on four source domains and three target datasets showed that AdapterSoupRec significantly outperformed nine strong existing baselines, including the state-of-the-art multi-modal graph-based recommender systems (see Table 8.2). In particular, we showed that an SSL-trained MLLM exhibits similar performance benefits to a cross-entropy-trained MLLM in the used multi-domain recommendation datasets. We also showed that the greedy soup effectively enhanced top-K multi-domain recommendation performance compared to the uniform soup in Table 8.2. Furthermore, LLaVA proved to be a more effective MLLM within AdapterSoupRec compared to CLIP. Our ablation study showed that the model soup, the graph adapter and the pre-training strategy significantly contribute to a more effective multi-domain recommendation (see Table 8.3). In addition, we measured the average Mean Squared Error (MSE) distances between the visual and textual embeddings of items from AdapterSoupRec and its non-soup variants. As shown in Figure 8.1, we found that the uniform and greedy soups produce lower average MSE values, further confirming the effectiveness of the model soup techniques in recommendations. Finally, we found that our AdapterSoupRec model, initially trained for e-commerce, still performs more effectively than the multi-modal graph-based recommendation baselines in an out-of-domain video recommendation scenario (see Figure 8.2).

In summary, we have validated one of the hypotheses of our thesis statement, namely that graph-based recommender systems can be effectively enhanced by mining self-supervised signals from multiple domains. However, despite that we have successfully developed more expressive graph neural architectures (Chapter 4) enriched with SSL-derived supervision signals from multiple modalities (Chapter 5 and Chapter 5) and multiple domains (Chapter 7 and Chapter 8), we further investigate a unified, advanced model in Chapter 9. This chapter demonstrates how the core concepts from this thesis can be synthesised and extended within a new, more powerful Large Language Model (LLM) paradigm, while preserving the strengths of each preceding method. We introduce a recommendation model that adapts the core concepts of previous approaches into more advanced forms, thereby achieving improved top-K recommendation performance in graph-based recommender systems.

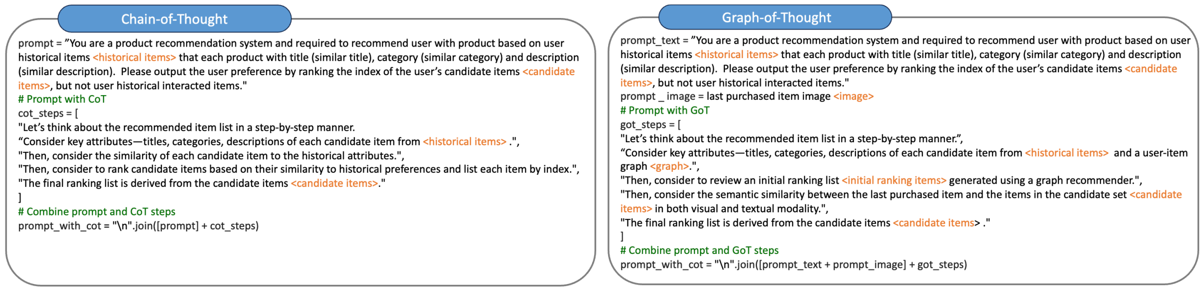
Chapter 9

A Unified SSL-based Recommender System with Multiple Modalities and Domains

In the previous chapters, we have presented all of the contributions supporting our thesis statement proposed in Section 1.2. In particular, Chapter 4 introduced graph positional encodings at the message-passing level and a graph transformer at the architectural level. Both were trained with SSL losses to enhance the expressiveness of graph neural architectures. Chapter 5 then presented modality-specific graph augmentations and deep modality alignment techniques using an SSL paradigm to enhance modality fusion. This was further enhanced by Chapter 6, which introduced the Unified Graph Transformer (UGT). UGT uses SSL to unify the modality fusion and extraction processes, projecting multi-modal representations into a cohesive semantic space for improved top-K recommendation. Next, in Chapter 7, we proposed an ID-based approach that combines SSL-based pre-training with a novel graph prompting technique to enable effective knowledge transfer across different domains. Finally, to reduce the reliance on ID-based features, Chapter 8 demonstrated the use of SSL to pre-train multi-modal LLMs, producing universal representations for improved top-K recommendation across multiple domains.

Although our proposed SSL approaches have successfully enhanced the effectiveness of graph-based recommender systems, they represent a suite of specialised models, each tailored to the top-K recommendation task within a specific setting (i.e., general, multi-modal, or multi-domain). Building on these proposed advances, this chapter presents a unified step that synthesises and extends the core principles from all preceding chapters into a more advanced graph-based recommender system for top-K recommendation. We introduce GollaRec, which leverages a new Graph-of-Thought (GoT) prompting technique within a multi-modal large language model (MLLM), specifically LLaVA, to integrate both visual and textual information into a unified graph-structured prompt. This design enables a single, unified model to apply its multi-modal understanding of items (such as images and descriptions) across the various top-K recommendation settings (i.e., general, multi-modal, and multi-domain). In particular, GollaRec represents a substantial expansion of the graph prompting technique from Chapter 7. The newly proposed GoT prompting technique uniquely incorporates the user-item interaction graph and user-interacted items with both images and textual descriptions. First, the user-item interaction graph is represented as special graph tokens that encode positional and neighbourhood information into the used LLM. This is a direct conceptual extension of the graph positional encodings introduced in Chapter 4. Second, the user's historical interactions with items – including both their images and textual descriptions – are combined sequentially and fed into the MLLM for joint multi-modal reasoning. This inclusion of multi-modal data extends the fusion and extraction techniques from Chapters 5 and 6. The multi-modal features are no longer treated as external inputs but serve as integral components of the reasoning 'thoughts' themselves, allowing the LLM to process visual and textual data simultaneously. As a result, GoT prompting enables multi-step, structured, and context-aware reasoning within the LLM, so as to improve the inference quality in top-K recommendation. Furthermore, GollaRec extends prior graph transformer designs from a specialised graph-transformer architecture (Chapter 4) to a graph-MLLM architecture on domain transfer. It extends the graph prompting strategy from Chapter 7 by integrating it with the universal, multi-modal representation learning paradigm established in Chapter 8. In this chapter, we build upon recent advances in MLLM, leveraging the SSL paradigm to extend and enhance the proposed techniques from Chapters 4 to Chapter 8, resulting in a more effective solution to the top-K recommendation task.

LLMs have demonstrated a remarkable capability in language understanding in various real-world scenarios (Jiang et al., 2024; Touvron et al., 2023). However, their training on unstructured data often limits their capability in handling complex tasks necessitating complex, multi-step reasoning or a precise contextual understanding (Lei et al., 2023). This limitation becomes particularly important with graph-structured data, which is essential in recommender systems (Guo et al., 2023). Indeed, graph data, with its complex relational structures between node entities, poses a unique challenge for LLMs, which typically do not encounter structured data formats like column-indexed records during their pre-training, leading to difficulties in handling domain-specific knowledge inherent to such data (Yu et al., 2023a). As discussed in Section 3.3, recent advances in prompting techniques have enhanced the LMMs' capability to address complex reasoning tasks (Jin et al., 2022). For example, Brown et al. (2020) employed few-shot in-context learning to enhance the reasoning capabilities of an LLM by using input and output examples as prompts. Wei et al. (2022) and Wang et al. (2022) enhanced the LLMs' effectiveness by using Chain-of-Thought (CoT) prompting, which involves a series of demonstrations where each step of the detailed, step-by-step explanation, serves as an instructive example to guide reasoning processes. Despite its effectiveness in linear textual reasoning, CoT does not inherently extend to tasks involving structural graph data, which necessitate mining complex relational structures (Jiang et al., 2023). This identified limitation, henceforth denoted as *insuffi-*



```
prompt = "You are a product recommendation system and required to recommend user with product based on user
historical items <historical items> that each product with title (similar title), category (similar category) and description
(similar description).  Please output the user preference by ranking the index of the user's candidate items <candidate
items>, but not user historical interacted items."
# Prompt with CoT
cot_steps = [
"Let's think about the recommended item list in a step-by-step manner.
"Consider key attributes—titles, categories, descriptions of each candidate item from <historical items> .",
"Then, consider the similarity of each candidate item to the historical attributes.",
"Then, consider to rank candidate items based on their similarity to historical preferences and list each item by index.",
"The final ranking list is derived from the candidate items <candidate items>."
]
# Combine prompt and CoT steps
prompt_with_cot = "\n".join([prompt] + cot_steps)
```

```
prompt_text = "You are a product recommendation system and required to recommend user with product based on user
historical items <historical items> that each product with title (similar title), category (similar category) and description
(similar description).  Please output the user preference by ranking the index of the user's candidate items <candidate
items>, but not user historical interacted items."
prompt _ image = last purchased item image <image>
# Prompt with GoT
got_steps = [
"Let's think about the recommended item list in a step-by-step manner.",
"Consider key attributes—titles, categories, descriptions of each candidate item from <historical items>  and a user-item
graph <graph>.",
"Then, consider to review an initial ranking list <initial ranking items> generated using a graph recommender.",
"Then, consider the semantic similarity between the last purchased item and the items in the candidate set <candidate
items> in both visual and textual modality.",
"The final ranking list is derived from the candidate items <candidate items> ."
]
# Combine prompt and GoT steps
prompt_with_cot = "\n".join([prompt_text + prompt_image] + got_steps)
```

Figure 9.1: Chain-of-thought prompting and our proposed graph-of-thought prompting in the recommendation task.

cient graph mining, emphasises the need for enhancing LLMs to effectively tackle graph-related tasks. The graph data, especially user-item interaction graphs in recommender systems, encapsulates unique patterns that contain domain-specific knowledge. To bridge this gap, we integrate graph data into the prompt by linearising the structured data into textual sentences, thereby addressing the insufficient graph mining problem in CoT prompting.

In addition to the problem of insufficient graph mining in CoT, the language-based reasoning process is often overly complex and abstract (Huang and Chang, 2023). Instead, the use of the item images can be an intuitive medium in recommendation tasks. Indeed, integrating an image in a prompt not only enriches the modelling of the user profiles but also enables the used model to generate more coherent outputs. In this chapter, we use an MLLM, namely LLaVA (Liu et al., 2024a), to ensure that the user/item embeddings are semantically aligned across modalities. To prompt effective reasoning in the presence of multi-modality and a graph structure, we propose a multi-modal GoT prompting technique, as illustrated in Figure 9.1. In particular, the item images are embedded within the prompt sequence, serving as visual context alongside the textual descriptions to drive the reasoning process. To enable an effective understanding of the graph knowledge in the used MLLM, we leverage a two-step alignment strategy. First, we perform a text-graph alignment using an SSL loss to align the item textual embeddings and the corresponding item node embeddings in the semantic space. Second, we perform a graph instruction tuning on the MLLM so as to match each graph token with its textual description. In addition, we propose an adaptive truncation method that selectively maintains the most relevant interactions within our devised GoT prompt, so as to address the constraint of fixed input token length in the used MLLM. The remainder of this chapter is structured as follows:

- Section 9.1 describes our GollaRec model and the tackled top-K recommendation tasks in a multi-modal setting (top-K general and top-K multi-domain tasks);
- Section 9.2 presents the experimental setup and our research questions in this chapter;
- Section 9.4 summarises the key findings and insights derived from our results.

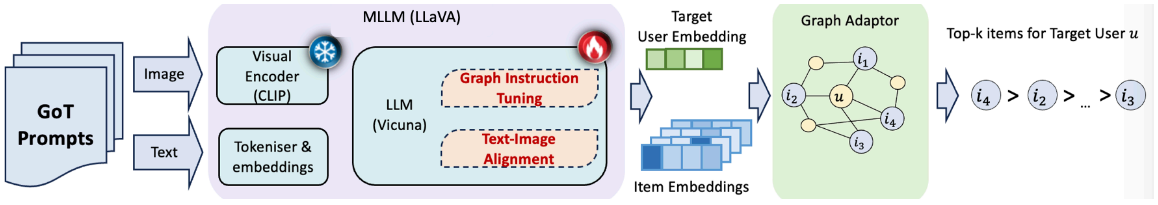


Figure 9.2: The architecture of our GollaRec model.

9.1 GollaRec Model

We first describe in Section 9.1.1 the top-K recommendation tasks we tackle in this chapter, namely a general recommendation task and a multi-domain recommendation task, both of which are conducted within a multi-modal setting. Section 9.1.2 presents our proposed GoT technique, which includes three key parts (adaptive graph truncation, text-graph alignment and text-image alignment). Next, we introduce the architecture of GollaRec in Section 9.1.3. The model is illustrated in Figure 9.2.

9.1.1 Top-K Recommendation Tasks

In this chapter, we address top-K general and top-K multi-domain recommendation tasks in a multi-modal setting that leverage the capabilities of a Multi-modal Large Language Model (MLLM) alongside a graph adaptor to process and integrate diverse data types. Following the same definition provided in Section 4.1.2.1, each task includes user and item sets, denoted as $\mathcal{U} = \{u\}$ and $I = \{i\}$ respectively, with embeddings $\mathrm{X} \in \mathbb{R}^{d\times(|U|+|I|)}$ where d is the dimensionality of these embeddings. Following the same notations defined in Section 5.1.1.1, we denote the items' multi-modal embeddings as $\mathbf{X}_{i,m} \in \mathbb{R}^{d_m\times|I|}$, where d_m is the dimension of that modality's embedding, $m \in \mathcal{M}$ is the set of modalities where $\mathcal{M} = \{v, t\}$, with v and t representing the visual and textual modalities[1], respectively. The users' historical behaviour data is denoted by $\mathcal{R} \in \{0,1\}^{|U|\times|I|}$, where each entry $\mathcal{R}_{u,i} = 1$ if the user u clicked item i, otherwise $R_{u,i} = 0$. Using the historical interaction data, we construct an interaction graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with $\mathcal{V} = \{\mathcal{U} \cup I\}$ and $\mathcal{E} = \{(u,i)|u \in \mathcal{U}, i \in I, \mathcal{R}_{ui} = 1\}$. Based on these notations, we formally define the top-K general recommendation task and top-K multi-domain recommendation task within the multi-modal setting addressed in this chapter:

Task 1 - General Recommendation: For the general recommendation task, given an interaction graph $\mathcal{G}$ enriched with the item descriptions and the item images, we aim to estimate the user preferences through an MLLM f_{θ_1} and an adaptor f_{θ_2} that recommend the top-K items for a target user u.

Task 2 - Multi-domain Recommendation: The multi-domain recommendation task extends the application of the MLLM f_{θ_1} and the adaptor f_{θ_2} across multiple source domains $\{D_1, \dots, D_n\}$

[1] We focus on these two modalities, since the used datasets only provide raw item images and descriptions.

to a new target domain D_{n+1}. The goal of this task is to rank the top-K items for a target user u in a new target domain D_{n+1}. Note that, in this task, we do not require overlapping users between the domains.

9.1.2 SSL-enhanced Graph-of-Thought

As discussed at the start of this chapter, LLMs often struggle with unfamiliar patterns and structures in graph data, thereby impeding these LLMs from generating accurate and coherent responses in graph-related tasks. To tackle this problem of insufficient graph mining, it is important to enhance the LLM's ability to interpret the interactions with the user-item graph. This necessitates a step-by-step demonstration of reasoning within the task. In this chapter, we propose a Graph-of-Thought (GoT) prompting technique, specifically designed to provide a structured rationale that outlines the reasoning steps needed for the recommendation tasks. Specifically, this GoT technique prompts the MLLM to reason about the potential candidates on the user-item interaction graph, taking into account the semantic similarity derived from both the visual and textual item embeddings in order to determine the final ranking list of the target user. Figure 9.1 (right) shows our proposed GoT technique in action when addressing two multi-modal recommendation tasks. However, effectively incorporating graph and multi-modal data for an MLLM empowered with a GoT technique presents several challenges:

- **C9.1**. How to integrate a large volume of textual nodes within a fixed input token length?
- **C9.2**. How to enable the MLLM to effectively model the relationship between the item images and their descriptions?
- **C9.3**. How to facilitate the MLLM's understanding of graph patterns in the user-item interaction graph?

In the following, we propose solutions to address these challenges, namely adaptive graph truncation, text-image alignment and text-graph alignment.

9.1.2.1 Adaptive Graph Truncation

In our initial experiments, we found that the MLLM used in GollaRec, LLaVA (Vicuna-7B), configures 576 tokens within its overall input token length limit (2048). We note that similar conclusions were observed across other LLaVA variants using different LLM backbones. As outlined in challenge **C9.1**, such amount of the total input token length markedly reduces the remaining available token length, thereby impeding the model's capability to encode richer user-item graph information. To address this limitation and ensure an effective user/item modelling, we leverage a pre-trained recommender to produce an initial candidate ranking list of items for the target user. We aim to maintain the most potential candidate items in both recommendation

Algorithm 9.1 Adaptive Graph Truncation in GoT

```
1: **Input:** User ID, Item ID and Descriptions, Max Tokens = 2048
2: **Output:** Truncated Item List
3: Initialise a pre-trained recommender (e.g., LightGCN)
4: # Generate initial ranking
5: items_list ← Recommender.RankItems(User ID, Item ID)
6: # Reserve tokens for visual data
7: total_tokens ← 576
8: Initialise initial_list as empty **for** *each item in items_list* **do**
 |9:
        **end**
        description ← GetDescription(item)
10: tokens ← Tokenise(description) **if** *total_tokens + length(tokens)* $\leq$ *Max Tokens* **then**
 |11:
        **end**
        append description to initial_list
12: total_tokens ← total_tokens + length(tokens) **else**
 |13:
        **end**
        **break**
14: **return** initial_list
```

tasks. In particular, we append the descriptions of these highly potential candidate items to the GoT prompt, and adaptively truncate this list so that it fits within the restricted token limit. Algorithm 9.1 presents our method for addressing the limited input token length of LLaVA. Figure 9.3 illustrates our adaptive truncation method applied within our GoT prompt.

9.1.2.2 SSL-based Text-image Alignment

To address challenge **C9.2** and enhance the MLLM's understanding of the relationships between multiple modalities, we use a contrastive pre-training method to pre-train LLaVA using all available item image-text pairs. Yi et al. (2025) also showed that fine-tuning MLLMs with image-text pairs significantly enhances the recommendation performance. Therefore, we use item images and their descriptions as inputs to pre-train LLaVA with an Image-Text Contrastive (ITC) loss (c.f. Equation (5.5) of Section 5.2). This pre-training method aims to maximise the similarity between the items' image and description pairs while minimising the similarity between the mismatched pairs, thereby facilitating a unified joint embedding space for multi-modal inputs within GoT. As such, LLaVA learns the relationships between the item descriptions and their corresponding images, hence enriching GoT with contextualised information.

9.1.2.3 SSL-based Text-graph Alignment

To address challenge **C9.3** – enhancing the understanding of graph structural information by the MLLM – we focus on aligning the encoding of the graph structures with the natural language space. This alignment enables the used MLLM to effectively capture the structural patterns using their language understanding capabilities. Inspired by prior works about aligning text and graph data for the node classification task (Tang et al., 2024; Wen and Fang, 2023), we employ a text-graph grounding method and a graph instruction tuning method to maintain the graph's structural context within the used MLLM in recommendation scenarios:

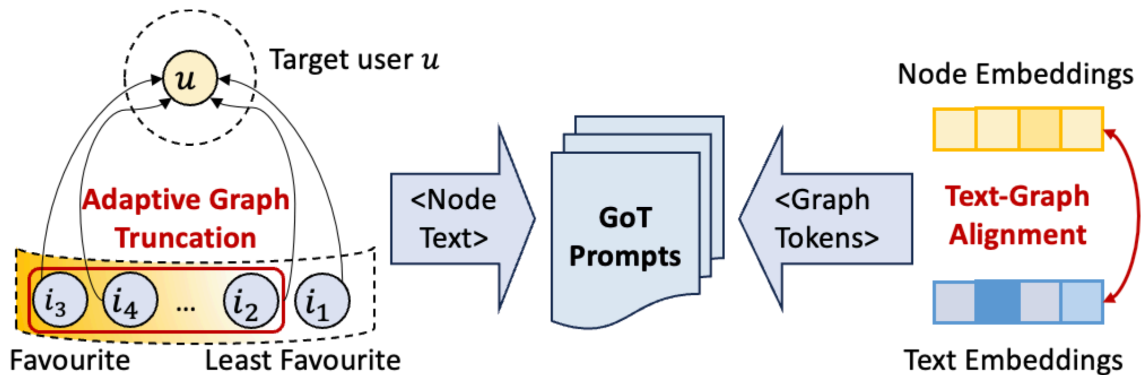


Figure 9.3: Our proposed GoT for recommendation.

(1) ***Text-graph grounding***: Following (Wen and Fang, 2023), we use a text encoder (namely BERT (Kenton and Toutanova, 2019b)) and a graph encoder (namely a graph transformer (Yun et al., 2019)) to align their resulting item node embeddings z_1 and the item textual embeddings z_2. We input item descriptions into both encoders to generate these embeddings, aligning them within a unified semantic space. Similar to the ITC loss (c.f. Equation (5.5)), we use a text-node SSL loss by (Wen and Fang, 2023) to differentiate between the node-text matching pair (z_{1p}, z_{2p}) as a positive pair and the non-matching pair (z_{1p}, z_{2q}) as a negative pair. As a result, we use this contrastive loss to refine this alignment, thereby preparing the well-trained graph encoder for the following instruction tuning.

(2) ***Graph instruction tuning***: Following the text-graph grounding phase, we use the pre-trained graph encoder to project the node embeddings into graph tokens using a Multi-Layer Perceptron (MLP): $\hat{z_1} = \text{MLP}\,(z_1)$, where are z_1 the node embeddings derived from the graph encoder and $\hat{z_1}$ represents the resulting graph tokens. These graph tokens, which represent graph structures, allow the MLLM to process and interpret the graph-structured data, thereby enabling the MLLM's understanding of graph patterns in the interaction graph. Inspired by GraphGPT's (Tang et al., 2024) methodology, we adapt a graph matching task to the context of recommender systems during this instruction tuning phase. Specifically, we construct the instruction by selecting a central item node and its l neighbouring nodes, presenting these nodes as a sequence of graph tokens (<graph_start>, <graph_token>$_1$, <graph_token>$_2$, ..., <graph_token>$_l$, <graph_end>). The goal of this matching task is to differentiate and match graph tokens with the corresponding language tokens using an MLLM. We input the projected graph tokens $\hat{z_1}$ and the instruction's textual embeddings z_3, for a given sequence of length l. We then compute the probability of generating the target output x_o as follows:

$$\psi\left(x_o \mid \hat{z_1}, z_3\right) = \prod_{j=1}^{l} \psi_{\theta_2}\left(x_j \mid \hat{z_1}, z_3\right) \tag{9.1}$$

where θ_2 are the learnable parameters within GollaRec, and ψ_{θ_2} is the probability of the j-th token x_j. Moreover, we optimise the MLLM's performance in matching the shuffled list of language tokens to the ordered sequence of graph tokens using a cross-entropy loss. As such, we enhance the understanding of graph structural information by the MLLM, thereby addressing the insufficient graph mining problem.

9.1.3 The GollaRec Model Architecture

Figure 9.2 provides a detailed overview of the GollaRec model architecture, including the GoT technique to prompt the LLaVA model as input and an adapter, namely LightGCN (He et al., 2020), to propagate the resulting embeddings from the MLLM's last layer and output final embeddings for ranking. We obtain the final user embeddings using the adapter as follows: $h_u = \sum_{i \in \mathcal{N}_u} \frac{h_i}{\sqrt{|\mathcal{N}_i||\mathcal{N}_u|}}$, where $\mathcal{N}_i$ and $\mathcal{N}_u$ denote the set of neighbours for user u and item i, respectively, while $|\mathcal{N}_u|$ and $|\mathcal{N}_i|$ represent the size of $\mathcal{N}_u$ and $\mathcal{N}_i$. Analogously, we also obtain the item embeddings. Our GollaRec model leverages two types of input, with the textual input encompassing the item descriptions and the visual input including the item images. During the training stage, the SSL-based text-graph and SL-based text-image alignment methods enable LLaVA to capture graph structures and the relationships between the item images and descriptions. For the inference stage, we append the resulting GoT to our GollaRec model's input so as to generate the corresponding user/item embeddings, as shown in Figure 9.2. Then, we use a graph adapter to integrate the user-item interactions, thereby refining the final user/item embeddings for the top-K multi-modal recommendation tasks. As such, by leveraging an SSL-based ITC loss (c.f. Equation 5.5) to train GollaRec on text-graph and text-image pairs (with and without multi-domain training data), we can mine additional supervision signals from multiple modalities and domains with a graph-MLLM architecture. In the next section, we conduct extensive experiments to justify the effectiveness of both GollaRec and its key components, so as to validate the overall argument of our thesis statement in Section 1.2: that graph-based recommender systems can be enhanced by mining self-supervised supervision signals from multiple modalities and domains using a more advanced graph neural architecture.

9.2 Experiments

We conduct experiments to validate the effectiveness of our GollaRec model on three public datasets, in comparison to nine strong baselines, including established and existing state-of-the-art recommender systems. To examine and analyse the effectiveness of our GollaRec model, we conduct experiments to answer the following four research questions:

- **RQ9.1**: How does our proposed GollaRec model perform compared with existing recommendation models include the models proposed in this thesis?
- **RQ9.2**: How do the key components of GollaRec affect the performance of the model?

Datasets	#Users	#Items	#Interactions	Sparsity
General Recommendation				
HM	27,883	2,742	185,297	99.76%
Clothing	39,387	22,499	185,297	99.99%
Baby	19,445	7,037	271,001	99.99%
Multi-domain Recommendation				
Food	115,349	39,670	$1,027,413$	99.99%
Home	731,913	185,552	$6,451,926$	99.99%
Clothing	39,387	23,033	237,488	99.97%
Office	87,436	25,986	684,837	99.97%
Pantry	13,101	4,898	126,962	99.82%
Electronics	192,403	63,001	$1,689,188$	99.99%
Sports	87,436	25,986	684,837	99.95%

Table 9.1: Statistics of the used datasets.

- **RQ9.3**: Does GollaRec exhibit a better integration of the item descriptions and images compared to the strongest baseline?

- **RQ9.4**: How do the length of GoT and the position of demonstration steps affect the performance of our model?

9.2.1 Datasets and Experimental Settings

In order to evaluate the effectiveness of our GollaRec model across top-K general and multi-domain recommendation tasks in a multi-modal setting, we conduct experiments on six commonly used datasets – three are focused on general recommendations and three on multi-domain recommendations. For general recommendations, we use Amazon Clothing and Baby datasets (He and McAuley, 2016a) from Section 5.2, and additionally include the HM fashion dataset (Xian et al., 2023) to further validate our model's effectiveness in a distinct recommendation scenario. For multi-domain recommendations, we conduct experiments across seven domains derived from the same Amazon Review datasets introduced in Section 8.3. We use the Food, Home, Clothing, and Office datasets as the source domain datasets and evaluate GollaRec and the used baselines on three target datasets, namely Pantry, Electronics and Sports. We choose these datasets for their extensive user-item interactions and rich multi-modal data, which include images and detailed textual descriptions such as titles, categories, and brands (Zhou et al., 2023c). Table 9.1 presents the statistics of the used datasets.

9.2.2 Evaluation Protocol and Implementation Details

Following the evaluation setting in (Zhang Jinghao et al., 2021; Zhou et al., 2023c), we randomly split the datasets into training, validation, and testing sets using an 8:1:1 ratio. We op-

Method	General	Multi-modal	MLLM	Multi-domain	Language-based
LightGCN	✓	×	×	×	×
VBPR	✓	✓	×	×	×
MMGCL	✓	✓	×	×	×
BM3	✓	✓	×	×	×
CLIP	×	×	✓	×	×
BEiT-3	×	×	✓	×	×
LLaVA	×	×	✓	×	×
MOME	×	×	×	✓	×
PLE	×	×	×	✓	×
MGFN	×	×	×	✓	×
P5	×	×	×	×	✓
LMRecSys	×	×	×	×	✓
GollaRec	✓	✓	✓	✓	✓

Table 9.2: Summary of the compared approaches.

timise the hyper-parameters of both our GollaRec model and the baseline models using a grid search on the validation set. We use Recall@k and NDCG@k (c.f. Section 2.4) to examine the top-K recommendation performance for both the general and multi-domain recommendation tasks. We set K to 20 (Zhang Jinghao et al., 2021), and report the average performance achieved for all users in the test set. All used baselines and our GollaRec model are implemented with PyTorch and were run on two GPU A6000s with 96GB memory. Our source code and model checkpoints are publicly available at: `https://github.com/zxy-ml84/GollaRec`.

9.2.3 Baselines

To examine the effectiveness of our GollaRec model in the general recommendation task, we compare GollaRec against nine existing state-of-the-art models:

- **LightGCN** (He et al., 2020) is widely-used graph-based model as introduced in Section 3.1;
- **VBPR** (He and McAuley, 2016b), **MMGCL** (Yi et al., 2022) and **BM3** (Zhou et al., 2023b) are multi-modal models as described in Section 3.2;
- **CLIP** (Radford et al., 2021) and **LLaVA** (Liu et al., 2024a) are MLLM models as described in Section 8.3.1.

We further compare GollaRec with three LLM-based recommenders:

- **P5** (Geng et al., 2022), which is pre-trained on the user-item interaction data to adapt the LLMs for recommendations. It converts the recommendation tasks into tailored natural language sentences using personalised prompts;
- **LMRecSys** (Zhang et al., 2021) transforms the recommendation task into a language modelling task by converting a user's interaction sequence into a nature language query;

Dataset	HM		Clothing		Baby	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
LightGCN	0.1254*	0.0743*	0.0553*	0.0246*	0.0714*	0.0319*
VBPR	0.1108*	0.0717*	0.0611*	0.0277*	0.0740*	0.0329*
MMGCL	0.1633*	0.0964*	0.0607*	0.0277*	0.0790*	0.0352*
BM3	<u>0.1711</u>*	<u>0.0981</u>*	<u>0.0797</u>*	0.0358*	<u>0.0863</u>*	<u>0.0380</u>*
CLIP	0.0956*	0.0687*	0.0631*	0.0281*	0.0664*	0.0304*
BEiT-3	0.0874*	0.0661*	0.0617*	0.0265*	0.0688*	0.0311*
LLaVA	0.1346*	0.0910*	0.0702*	0.0315*	0.0674*	0.0316*
P5	0.1417*	0.0872*	0.0766*	<u>0.0360</u>*	0.0825*	0.0356*
LMRecSys	0.1269*	0.0801*	0.0623*	0.0322*	0.0778*	0.0322*
TALLREC	0.1145*	0.0782*	0.0632*	0.0335*	0.0752*	0.0313*
GollaRec-CoT	0.1807*	0.1039	0.0911*	0.0404*	0.0939*	0.0410
GollaRec	**0.1880**	**0.1064**	**0.0932**	**0.0423**	**0.0958**	**0.0425**

Table 9.3: Comparison of GollaRec with the used general recommendation baselines. * denotes a significant difference with a baseline using the Holm-Bonferroni corrected paired t-test with $p < 0.05$.

- **TALLRec** (Bao et al., 2023) models the sequence of user interactions using the Llama-2-7B (Touvron et al., 2023) as its backbone in a sequential recommendation setting and predict users' next interacted item.

For the multi-domain recommendation task, apart from using the same multi-modal methods above, we also compare GollaRec with three multi-domain recommenders: MOME (Ma et al., 2018), PLE (Tang et al., 2020) and MGFN (Zhang et al., 2022a) as described in Section 8.3.1. We summarise the used baselines in Table 9.2.

9.2.4 Performance Comparison (RQ9.1)

We compare GollaRec with all the used baselines. Table 9.3 and Table 9.4 present the comparison results for the general and multi-domain recommendation tasks, where "GollaRec-CoT" refers to a GollaRec variant that uses standard Chain-of-Thought (CoT) instead of our proposed Graph-of-Thought (GoT) that allows reasoning over the user-item interaction graph. To assess the significance of the reported performance differences between GollaRec and the baselines, we use a paired t-test with the Holm-Bonferroni correction (p < 0.05).

9.2.4.1 General Recommendation Task

Table 9.3 shows the performance of our GollaRec model and the general recommendation baselines. The results highlight several key findings: (1) Our GollaRec model consistently achieves the best performance on all the used datasets. Compared with the best baseline model (namely BM3), GollaRec achieves an average improvement of 12.7% across all the datasets. These

Dataset	Pantry		Electronics		Sports	
Methods	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
VBPR	0.0723*	0.0326*	0.0442*	0.0196*	0.0771*	0.0349*
MMGCL	0.0907*	0.0377*	0.0627*	0.0304*	0.0913*	0.0428*
BM3	<u>0.0932</u>*	<u>0.0417</u>*	<u>0.0638</u>*	<u>0.0310</u>*	<u>0.0970</u>*	<u>0.0438</u>*
CLIP	0.0683*	0.0318*	0.0461*	0.0235*	0.0727*	0.0310*
BEiT-3	0.0596*	0.0289*	0.0481*	0.0240*	0.0748*	0.0341*
LLaVA	0.0659*	0.0313*	0.0604*	0.0288*	0.0709*	0.0303*
MOME	0.0797*	0.0352*	0.0573*	0.0261*	0.0749*	0.0318*
PLE	0.0862*	0.0384*	0.0595*	0.0278*	0.0866*	0.0367*
MGFN	0.0891*	0.0413*	0.0623*	0.0305*	0.0894*	0.0383*
GollaRec (CoT)	0.1183	0.0469*	0.0655*	0.0323*	0.1046*	0.0456
GollaRec	**0.1213**	**0.0495**	**0.0681**	**0.0350**	**0.1112**	**0.0502**

Table 9.4: Comparison of GollaRec with the used multi-domain baselines. * denotes a significant difference with a baseline using the Holm-Bonferroni corrected paired t-test with $p < 0.05$.

results demonstrate the effectiveness of GollaRec in the general recommendation task; (2) GollaRec outperforms the MLLM models (CLIP, BEiT-3, LLaVA) by a large margin on all used datasets. The suboptimal performance of the previous MLLM models suggests their failure to effectively prompt the MLLM to a recommendation task and the absence of informative interactions in the models. Overall, our results indicate that GollaRec, which prompts an MLLM with interactions is more effective than relying solely on visual and textual similarities for recommendations; (3) When comparing GollaRec with language-based models (P5, LMRecSys, TALLRec) and multi-modal models (VBPR, MMGCL, BM3) in Table 9.3, we observe that our GollaRec model significantly outperforms both groups of models on the three used datasets. This indicates that our GollaRec model successfully adapts an MLLM to the recommendation task, effectively leveraging textual and visual prompts with GoT to recommend more accurate items; (4) From Table 9.3, we observe that our GollaRec model significantly outperforms the GollaRec-CoT variant on all three datasets. This indicates that GoT, with its integration of the user-item interaction graph as additional context, is more effective in handling the general recommendation task compared to CoT; (5) When comparing our GollaRec model with LightGCN, it is notable that our GollaRec model initialises node embeddings with MLLM-initialised embeddings, whereas LightGCN uses randomly-initialised embeddings. Our GollaRec model significantly outperforms LightGCN by a large margin across all used datasets, indicating the effectiveness of leveraging pre-trained embeddings over training from scratch.

9.2.4.2 Multi-domain Recommendation Task

From the observed multi-domain recommendation results in Table 9.4, we observe the following findings: (1) We observe that GollaRec significantly outperforms all the baselines and GollaRec-CoT in most instances (except in 2 out of 60 instances) across the three used datasets. These results confirm GollaRec's capability in effectively leveraging both domain-specific and

Dataset	General Rec (Clothing)		Multi-domain Rec (Sports)	
Variants	Recall@20	NDCG@20	Recall@20	NDCG@20
w/o GoT	0.0885*	0.0402*	0.0982*	0.0442*
w/o Adapter	0.0821*	0.0358*	0.0848*	0.0363*
w/o Text-image Alignment	0.0868*	0.0389*	0.0941*	0.0436*
w/o Text-graph Alignment	0.0901*	0.0402*	0.1068	0.0468*
GollaRec	**0.0932**	**0.0423**	**0.1112**	**0.0502**

Table 9.5: Results for ablating the key components of GollaRec. * indicates a significant difference using a paired t-test with $p < 0.05$.

shared common knowledge within multi-domain recommendation settings. In addition, the results show that GoT is a more effective prompting technique than CoT, particularly in integrating the interaction graph information to generate effective user/item embeddings. (2) When comparing the multi-modal models (VBPR, MMGCL, BM3, GollaRec) with the multi-domain baselines (MOME, PLE, MGFN), we observe that the models from the former group, which incorporate multi-modal item contents, are consistently more effective. This result suggests that incorporating rich multi-modal data facilitates a more effective transfer of multi-modal semantic knowledge in recommendation scenarios. (3) When compared with the MLLM models (CLIP, BEiT-3, LLaVA), GollaRec demonstrates significant performance improvements on the three used multi-domain datasets. Such a superior performance emphasises the importance to leverage the abundant interactions from the target domain in order to enhance the MLLM's understanding of the multi-domain recommendation task.

9.2.5 Ablation Study (RQ9.2)

We conduct an ablation study to assess the impact of the different components of GollaRec, including the GoT, Adaptor, Text-graph and Text-image alignment methods. We illustrate the results using the Clothing dataset for the general recommendation task and the Sports dataset for multi-domain recommendation task since we observe similar trends and conclusions across all the other used datasets. First, to gauge the effectiveness of GoT in GollaRec, we conduct a comparative analysis by removing GoT and retaining the initial prompt for the task description. From Table 9.5, we observe that GollaRec significantly outperform its "w/o GoT" variant in all instances on both datasets. This result confirms that prompting GollaRec with a graph structure results in an improved recommendation performance. Next, we ablate the adapter used in our GollaRec model, so as to examine its usefulness. We observe a marked decrease in GollaRec's performance when removing the graph adapter (c.f. the "w/o Adaptor" variant in Table 9.5). This confirms the necessity of including personalised information such as the user-item interactions in order to capture domain-specific knowledge in GollaRec. Table 9.5 also shows that the "w/o Text-image Alignment" variant exhibits a reduced performance compared to GollaRec

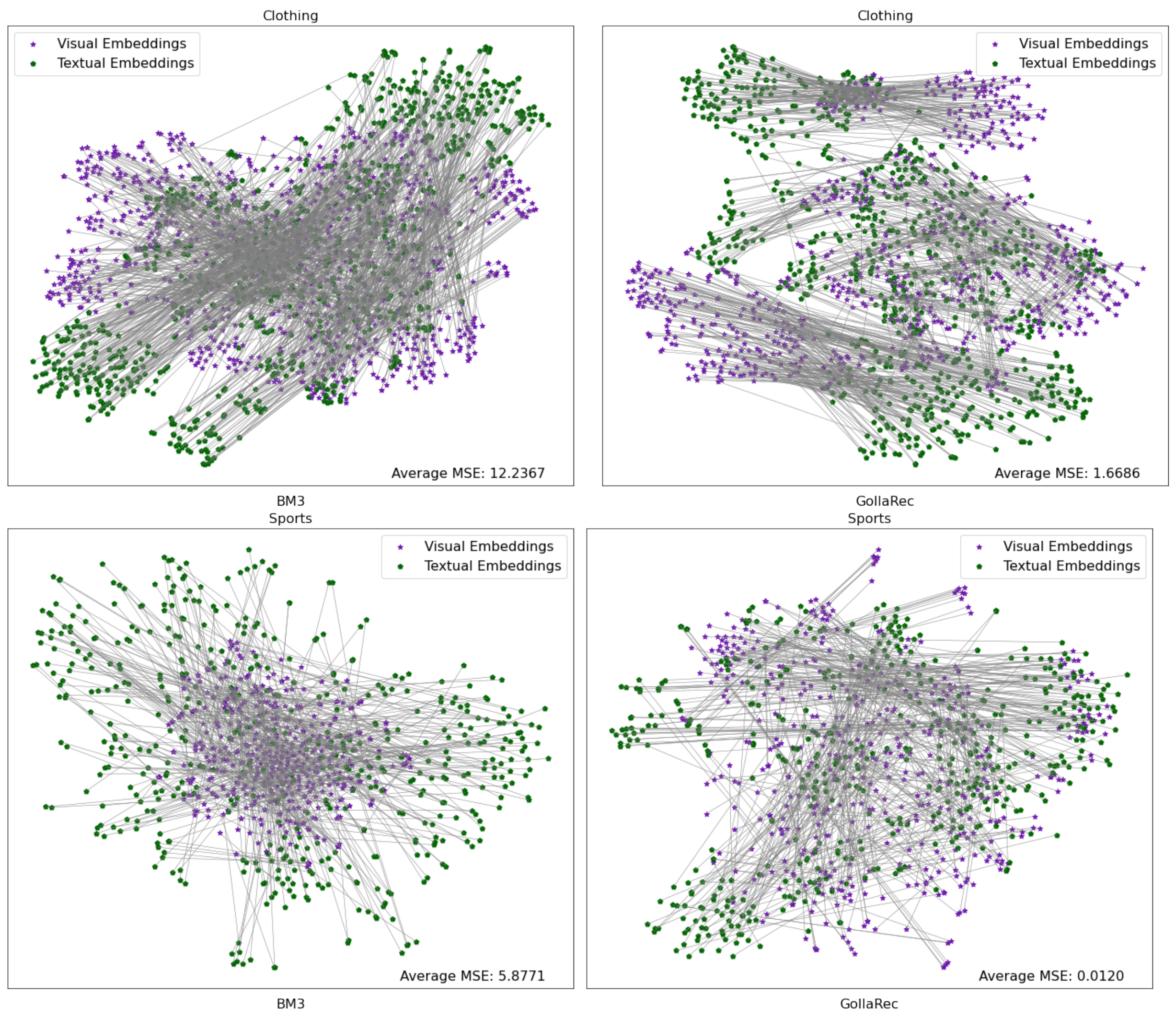


Figure 9.4: The t-SNE visualisation of the item embeddings on the Sports and Clothing datasets. A star refers to a visual embedding while a pentagon represents a text embedding. The average MSE value indicates the average distance between the visual and textual embeddings.

on both datasets. This highlights the importance of using SSL in enabling the understanding the relationship between the item textual descriptions and the images within the GoT prompt, in order to enhance the recommendation performance in both general and cross-domain recommendation tasks. In addition, we observe that GollaRec significantly outperforms its "w/o Text-graph Alignment" variant in 3 out of 4 instances. This result shows the effectiveness of our text-graph alignment method, which leverages SSL to align node embeddings with their corresponding textual descriptions in the semantic space in our GollaRec model, thereby addressing Challenge C9.3 (refer to Section 9.1.2) by enhancing the MLLM's ability in interpreting graph patterns within the user-item graph.

9.2.6 Modality Alignment of GoT-enhanced MLLM (RQ9.3)

To address challenge C9.2 (c.f. Section 9.1.2) and assess the effectiveness of integrating the item descriptions and images in our GoT prompt, we visualise the resulting embeddings to see

Dataset	General Rec (Clothing)		Multi-domain Rec (Sports)	
Variants	Recall@20	NDCG@20	Recall@20	NDCG@20
- RandomDemonstrationPos	0.0920†	0.0424†	0.1027	0.0465†
- RandomImagePos	**0.0941**†	**0.0426**†	0.1061†	**0.0508**†
- RandomLenTrunction (80%)	0.0834	0.0366	0.0983	0.0440
- RandomLenTrunction (60%)	0.0807	0.0334	0.0960	0.0425
GollaRec	0.0932	0.0423	**0.1112**	0.0502

Table 9.6: Overall performance of GollaRec with different GoT lengths and text/image prompts' positions. † indicates an equivalent effectiveness using a two one-sided test (TOST) with $p < 0.05$.

if these modalities are closely aligned under the use of SSL – an indicator of whether GollaRec integrates coherent semantics across different modalities within a unified semantic space. For conciseness and space constraints, we compare the results of GollaRec with the strongest performing baseline according to Table 9.3 and Table 9.4, namely BM3, in both the general (Clothing dataset) and multi-domain (Sports dataset) recommendation scenarios. Note however that we observe similar trends and conclusions with other baselines and datasets. We anticipate that a higher-quality multi-modal embedding exhibits cohesive distributions and lower MSE values, indicating that the corresponding model has effectively interpreted both modalities. In contrast, embeddings that are of poorer quality likely appear more dispersed and should exhibit higher MSE values, indicating a lack of a comprehensive understanding of the modalities by the model. Figure 9.4 shows the visualisations of the obtained items' visual and textual embeddings on the Sports and Clothing datasets. From Figure 9.4, we observe that the items' visual and textual embeddings in BM3 are widely and distantly scattered across both datasets. Conversely, GollaRec's embeddings are cohesively distributed, resulting in a more unified semantic space on the Clothing and Sports datasets. In addition, the markedly lower average MSE values for GollaRec (1.66, 0.12) compared to BM3 (12.23, 5.87) on these datasets indicate that GollaRec appears to have successfully addressed challenge C9.2 by using SSL to integrate coherent semantics across different modalities, thereby improving the model's performance in downstream recommendation tasks.

9.2.7 Impact of GoT's Length and Position (RQ9.4)

Given that the prompt position and input length are crucial in language modelling (Navigli et al., 2023), we assess the GoT structure in our model by randomly shuffling the positions of demonstration prompts and images within GoT, and by constraining the maximum input token length. Table 9.6 reports the obtained results on the Clothing and Sports datasets. We do not report results on other datasets since they show similar trends and conclusions. Specifically, the "-RandomDemonstrationPos" variant randomly shuffles the positions of demonstration prompts within the GoT structure, where "demonstration" serves as an example for each step (Wei et al., 2022). From the table, we observe that GollaRec performs on par with its "- RandomDemonstrationPos" variant in 3 out of 4 instances across both datasets, suggesting that the random repositioning of the step-by-step prompts does not affect the model performance. Similarly, adjusting the position of the image prompt within GoT ("- RandomImagePos" variant) shows no positive effect on the recommendation performance. In addition, we assess the impact of constraining the GoT's length to 80% and 60% of the maximum token length, considering that the image tokens already use 25% of the token capacity, and the other necessary tokens, such as the system's tokens, further reduce the remaining available length. From Table 9.6, we observe decreases in performance in the "- RandomLenTrunction" variants when reducing the maximum input token length on both datasets. This result indicates that GollaRec relies on a richer GoT, which encapsulates essential demonstration prompts for the recommendation tasks. This finding indicates that GollaRec effectively addresses the problem of limited token length.

9.3 Discussion

In this section, we have demonstrated that our GollaRec model, an MLLM-based recommendation model trained with SSL losses, can produce effective user/item embeddings for both top-K general and multi-domain recommendation tasks in a multi-modal setting. This GollaRec model represents a unified approach that synthesises and extends the core principles from all preceding chapters into a single, advanced graph-based recommender system. By leveraging the novel Graph-of-Thought (GoT) prompting technique, GollaRec extends the architectural expressiveness of Chapter 4 into a graph-MLLM architecture. It simultaneously integrates the deep multimodal fusion strategies of Chapter 5 and the end-to-end learning paradigm of Chapter 6 directly into the MLLM's reasoning process. Furthermore, it unifies these capabilities with the universal representation learning and domain adaptation paradigms established in Chapters 7 and 8. The success of this unified approach provides the final empirical validation for our thesis statement. It confirms that graph-based recommender systems can be enhanced by mining self-supervised supervision signals from multiple modalities and domains using a more advanced architecture.

9.4 Conclusions

In this chapter, we showed that a unified recommender system that synthesises and extends the core SSL-enhanced principles (graph transformer architecture, graph prompting technique, multi-modal encoding and multi-domain adaptation) from all preceding chapters can lead to a new state-of-the-art graph-based recommender system. We indeed validate our thesis's argument that graph-based recommender systems can be enhanced by effectively mining more supervision signals from multiple modalities and multiple domains for top-K recommendation. We introduced GollaRec, a novel SSL-enhanced graph-MLLM recommendation model for both general and multi-domain recommendation tasks in a multi-modal setting. We first proposed the GoT prompt that integrates user-item graph information and multi-modal item data for estimating the user preferences within a graph-MLLM architecture. In particular, we used an SSL-based text-graph alignment method and graph instruction tuning to effectively capture the graph patterns within GoT. We then leveraged an adaptive graph truncation method to maximise the number of high-potential items inputted into the GoT, thereby addressing the limited token length in the used MLLM. Our extensive experiments on six public datasets showed that GollaRec significantly outperformed twelve strong existing baselines for both tacked recommendation tasks (see Table 9.3 and Table 9.4). The performance improvement reaches up to 18.2% in comparison to the strongest baseline model on the used datasets. We also conducted an ablation study, which confirmed the significant contributions of the components of GollaRec, namely the GoT technique, the graph adapter, the SSL-based text-graph and text-image alignment methods, to a more effective multi-modal recommendation (Table 9.5). Furthermore, we analysed the impact of different lengths and positions for GoT (see Table 9.6), and confirmed that GollaRec effectively incorporates sufficient context of top-K recommendation – including user-item graph tokens, item images, descriptions, and the initial ranking list – within the limited token length, thereby enhancing performance on top-K recommendation tasks.

In summary, we have validated the hypotheses of our overall thesis statement, namely that graph-based recommender systems can be enhanced by mining self-supervised supervision signals from multiple modalities and domains using a more advanced architecture. In the next chapter, we will close this thesis by summarising the results and conclusions of each chapter and discussing possible future directions uncovered by this work.

Chapter 10

Conclusions and Future Work

10.1 Contributions and Conclusions

This thesis addressed the challenges in top-K graph-based recommender systems (c.f. Section 1.1): the limited expressiveness of graph neural architecture, the insufficient modality encoding issue and the insufficient domain transfer capabilities. In particular, as motivated by the Self-Supervised Learning (SSL) paradigm, we postulated that graph-based recommender systems can be effectively enhanced by mining more supervision signals from multiple modalities and domains along with a more expressive graph neural architecture using an SSL paradigm. In Section 1.1, we argued that existing graph-based models have the following challenges:

- **Challenge 1:** Current graph-based recommender systems exhibit limited expressiveness, as they fail to address the well-known over-smoothing problem (i.e., representations becoming indistinguishable) and adequately denoise the noisy implicit interactions between users and items (c.f. Chapter 4).

- **Challenge 2:** When multiple modalities, such as item descriptions and item images, are available, graph-based recommender systems struggle to effectively fuse these multimodal data in top-K recommendation (c.f. Chapter 5).

- **Challenge 3:** Graph-based recommender systems still face the long-standing isolation problems, namely isolated feature extraction process and isolated modality encoding process, in the recommendation pipeline (c.f. Chapter 6).

- **Challenge 4:** Graph-based recommender systems lack sufficient capability to effectively transfer high-quality and generalisable knowledge across domains (c.f. Chapter 7).

- **Challenge 5:** Graph-based recommender systems typically rely on ID-based features, thereby impeding the effective modelling of highly generalisable user/item representations across multiple domains (c.f. Chapter 8).

To address the five aforementioned challenges, we have proposed various models and techniques throughout this thesis. In the following, we describe our main contributions and key findings from the thesis:

- **Conclusion 1: Effective Graph Neural Architecture Using Graph Positional Encodings and Graph Transformer Architecture:** To address **Challenge 1**, in Chapter 4, we explored how to leverage graph positional encodings (e.g., Laplacian eigenvectors) at the message-passing level and how to use the graph transformer architecture at the overall architectural level to enhance the architectural expressiveness of graph-based recommender systems. Specifically, we leveraged Laplacian Positional Encoding (LapPE) and Random Walk Positional Encoding (RWPE) as graph positional encodings in an SSL paradigm (Section 4.1.2) to generate more distinguishable user/item embeddings (c.f. Table 4.5), thereby alleviating the over-smoothing problem with an improved model expressiveness. Moreover, we proposed a diffusion-based graph transformer model, DiffGT, to denoise the noisy implicit interactions within a diffusion process (Section 4.2.2). In particular, we applied SSL to maximise the similarity between original user/item embeddings and their denoised ones (see Section 4.2.2.3), thereby enhancing the model expressiveness with improved top-K recommendation performance (see Table 4.7).

- **Conclusion 2: Enhanced Modality Fusion By Modal-specific Graph Augmentations and Deep Modality Alignment:** To address **Challenge 2** in Chapter 5, first, we introduced modal-specific edge dropout and modal-specific masking augmentations in the SSL paradigm (see Section 5.1.1) to facilitate the contribution of each modality in user/item representation learning, thereby enhancing modality fusion and improving the recommendation effectiveness in graph-based recommender systems (see Table 5.2). Second, we further enhanced the modality fusion by leveraging SSL-enhanced Large Multi-modal (LMM) encoders (see Section 5.2.2), namely CLIP, VLMo and BEiT-3, to enable deeper modality alignment to better capture the intrinsic semantic correlations between visual and textual data compared to the shallow alignment approaches in existing graph-based recommender systems (see Table 5.7).

- **Conclusion 3: Unified Multi-modal Recommendation Pipeline:** To address **Challenge 3**, in Chapter 6, we proposed the first end-to-end Unified Graph Transformer (UGT) model that combines a multi-way transformer and a unified graph encoder to overcome the isolated feature extraction and modality encoding problems within the multi-modal recommendation pipeline (see Section 6.1.2). In particular, UGT is trained by an SSL loss to unify the multi-modal representations into the same semantic space (see Figure 6.5), thereby enhancing the top-K recommendation performance (see Table 6.1).

- **Conclusion 4: Effective Domain Transfer Using Personalised Graph Prompts:** To address Challenge 4, in Chapter 7, we proposed an ID-based approach, PGPRec, that combines SSL-based pre-training of a graph encoder (see Section 7.1.3) with a novel item-wise graph prompting technique (see Section 7.1.5). This approach enables effective knowledge transfer by using the personalised prompts to adapt the pre-trained, general structural knowledge to the specific characteristics of the target domain (see Table 7.3).

- **Conclusion 5: Effective Domain Transfer Using Multi-modal Features as Universal Representations:** To reduce the reliance of ID-based features, in Chapter 8, we used SSL and cross-entropy losses to train MLLMs to generate highly generalisable representations and effective model configurations across multiple domains, so as to address Challenge 5 (see Section 8.2.2). We then weighted averaged these configurations (i.e., the 'model soup' technique) to obtain a more effective set of MLLM parameters and integrated this MLLM with a graph adaptor (see Section 8.2.3), thereby achieving improved top-K multi-domain recommendation in a multi-modal setting (see Table 8.2).

- **Conclusion 6: A Unified Graph-MLLM Recommender System:** To comprehensively address the full spectrum of challenges (Challenges 1–5), Chapter 9 introduced GollaRec. This model leverages SSL to perform deep text-graph and text-image alignment across multiple domains, thereby enabling the MLLM to understand both the structural knowledge within the user-item graph and the items' images associated with their descriptions (see Section 9.1.2). By integrating this multi-modal, multi-domain knowledge into a novel Graph-of-Thought (GoT) reasoning process, GollaRec performs as a unified Graph-MLLM architecture (see Section 9.1.3), effectively addressing the challenges in architectural expressiveness, modality encoding, and domain transfer within a single model with an improved effectiveness (see Table 9.3 and Table 9.4).

Based on the results from Chapters 4 to 9, we now validate our thesis statement proposed in Section 1.2. Our thesis stated that we can enhance the effectiveness of the graph-based recommender systems by obtaining more supervision signals, sourced from diverse domains and modalities, in a self-supervised learning manner. Specifically, this thesis hypothesised that by developing more expressive graph neural architectures, graph-based recommender systems can more effectively incorporate complex and structured information. In addition, this thesis argued that by effectively mining self-supervised signals from multiple item modalities, recommender systems can gain richer user and item representations, thereby improving the quality of recommendations. Moreover, we postulated that leveraging multi-modal semantics across abundant source domains via self-supervised learning can significantly enhance the effectiveness of graph-based recommendation models, thereby enabling an improved performance across various target domains. In the following, we discuss the corresponding experimental results and observations that validate our proposed thesis statement.

- **Claim 1:** *By developing more expressive graph neural architectures, graph-based recommender systems can more effectively incorporate complex and structured information,* *thus addressing common challenges such as over-smoothing and insufficient denoising capability inherent to existing graph-based models with an improved effectiveness.* Our experiments in Chapter 4 validated this claim by showing that our proposed Positional Graph Contrastive Learning (PGCL) model can significantly outperform five baselines by proposing SSL-enhanced graph positional encodings at the message-passing level (see Table 4.2). Table 4.5 showed that our PGCL model can also alleviate the over-smoothing problem in graph-based recommender systems. Moreover, we further validated this claim in Chapter 4 by proposing a Diffusion Graph Transformer (DiffGT) model at the overall architectural level to explicitly denoise the noisy implicit interactions using an SSL paradigm. Our experiments in Table 4.7 showed that our DiffGT model can outperform eleven existing state-of-the-art baselines. In addition, Figure 4.5 showed that our proposed graph diffusion approach can effectively denoise with lower Signal-to-Noise Ratio values during the diffusion process.

- **Claim 2:** *By effectively mining self-supervised signals from multiple item modalities, graph-based recommender systems can gain richer user and item representations, thereby improving the quality of top-K recommendations.* We have validated this claim in both Chapter 5 and Chapter 6, where we proposed enhanced modality fusion techniques (i.e., modal-specific graph augmentations and deep modality alignment) and a unified multi-modal graph transformer model within the SSL paradigm. In particular, the experimental results in Table 5.2 and Table 5.7 showed that SSL-based graph augmentations and SSL-enhanced deep modality alignment can improve the effectiveness of graph-based recommender systems. Moreover, Figure 6.5 showed that our proposed UGT model effectively uses SSL to unify the multi-modal representations into the same semantic space, which in turn leads to an improved top-K recommendation performance.

- **Claim 3:** *By effectively mining self-supervised signals from multiple domains, graph-based models can enrich user and item representations with transferable knowledge across domains, thereby enabling an improved top-K recommendation performance.* This claim has been validated in Chapter 7 and Chapter 8, where we proposed an SSL-based cross-domain approach and an SSL-based multi-domain approach, respectively. First, Chapter 7 proposed PGPRec, which combines the SSL-based pre-training of a graph encoder with an item-wise graph prompting technique to enable effective, ID-based cross-domain transfer. As shown in Table 7.3, PGPRec demonstrated more effective top-K recommendation performance over eleven strong recommendation baselines. Second, in Chapter 8, we proposed AdapterSoupRec for multi-modal-based domain transfer across domains. This approach uses SSL to train MLLMs and generate effective model configurations, which are then combined via a weighted average (i.e., the "model soup" technique) to obtain a single, more effective set of model parameters. Table 8.2 showed that AdapterSoupRec outperforms nine existing state-of-the-art baselines, while Figure 8.2 further highlighted its strong out-of-domain generalisability.

- **Claim 4:** *By effectively mining self-supervised signals from multiple modalities and domains using an advanced graph neural architecture, graph-based recommender systems can generate more effective user and item representations, thereby improving the top-K recommendation performance.* This claim has been validated in Chapter 9, where we proposed GollaRec that extends the core principles from Chapter 4 to 8. This model used SSL-based text-graph and text-image alignment methods to train a Multi-modal Large Language Model (MLLM) across multiple source domains. This training enabled the MLLM to understand both the structural knowledge within the user-item interaction graph and the semantic meaning of item images associated with their descriptions. This MLLM is then integrated with a graph adapter to ensure effective adaptation to the target domain. The experimental results in Table 9.3 and Table 9.4 demonstrated that GollaRec achieves superior top-K recommendation performance compared to twelve state-of-the-art baselines. Furthermore, Table 9.5 confirmed the effectiveness of using SSL to enable graph pattern and semantic understanding within the MLLM, as well as the effectiveness of the overall graph-MLLM architecture for top-K recommendation.

In summary, we have validated each of the claims of our thesis statement in Section 1.2. We have shown that we can improve the effectiveness of graph-based recommender systems using additional self-supervised signals from multiple modalities and domains along with more expressive graph neural architectures. In the following section, we describe some future research directions for graph-based recommender systems.

10.2 Directions for Future Work

In this section, we discuss possible future directions that could benefit the performance of graph-based recommender systems.

- **Graph Foundation Models for Top-K Recommendation:** This thesis has systematically demonstrated the effectiveness of advanced graph neural architectures (Chapter 4), the importance of multi-modal encoding (Chapters 5 and 6), and the effectiveness of pre-training for domain transfer (Chapters 7 and 8). Therefore, a natural and ambitious extension is to develop an effective Graph Foundation Model (GFM) with multi-modal capabilities for recommendations. Akin to how models like BERT (Kenton and Toutanova, 2019b) or GPT-4 (Achiam et al., 2023) serve as foundations in nature language processing field, a GFM would be pre-trained with SSL on massive, diverse graph-structured datasets integrate both graph structures (e.g., user-item graphs from e-commerce, social networks, etc.) and their associated multi-modal content (e.g., item images, descriptions, and videos).

This model would learn fundamental, generalisable patterns that connect graph topology, user behaviours, and multi-modal signals with the use of SSL, thereby creating a powerful base model that can be effectively and efficiently fine-tuned for various downstream recommendation tasks.

- **Graph-enhanced In-Context-Learning & Reasoning in LLM-based Recommender Systems:** The GollaRec model in Chapter 9 demonstrated a method for injecting static graph knowledge into an LLM via Graph-of-Thought (GoT) prompting. However, this relies on a pre-defined prompt. A more advanced direction is to enable the LLM with dynamic in-context graph reasoning, a capability that allows the model to autonomously extract and retrieve useful structural knowledge in response to a specific query. This might involve a Graph-enhanced Retrieval-Augmented Generation (Graph RAG) component (Han et al., 2024) that retrieves relevant subgraphs (e.g., the user's recent interactions, similar users, or related item communities) within the personalised context for an effective graph-aware reasoning step, thereby enhancing the top-K recommendation task.

- **Graph-enhanced Workflow Optimisation within Multi-agent Recommender Systems:** While a unified model like GollaRec (c.f. Chapter 9) is effective, it relies on a single LLM instance to execute the entire recommendation process via a fixed, sequential workflow. This design inherently suffers from a single-pass limitation, lacking the dynamic ability to critique or refine the intermediate results generated by its internal components before the final prediction. This limitation motivates a new direction, as recent work has begun to involve multiple, specialised LLM agents for top-K recommendation (Zhang et al., 2024a). However, these multi-agent systems often lack a method to adaptively coordinate the agents based on a specific context of personalisation. A promising future direction would be to model these agent interactions as an "agentic graph," where relationships and information flow are explicitly defined. This graph-based workflow could then be optimised using SSL. For instance, an SSL task could be designed to distinguish between effective collaboration patterns (positive samples) and ineffective ones (negative samples) within the agentic workflow graph. This would enable a more dynamic, self-optimising, and effective multi-agent system for top-K recommendation.

10.3 Concluding Remarks

This thesis has addressed the challenging task of top-K recommendation across general, multi-modal, and multi-domain settings. In particular, this thesis contributed to developing more effective graph-based recommender systems by leveraging advanced techniques in mining self-supervised signals from multiple modalities and domains using advanced graph neural architectures. However, there are still exciting topics and complex challenges in this research field, some of which have been highlighted in Section 10.2. This thesis has provided a solid motivation and the groundwork for further exploring these research directions in the future. We believe that using graph representation learning to represent users and items will continue to benefit the future development of the recommendation field.

Bibliography

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Agarwal, S. (2023). Multi-modal deep learning for unified search-recommendation systems in hybrid content platforms. *International Journal of AI, BigData, Computational and Management Studies*.

Ahmadian, S., Berahmand, K., Rostami, M., Forouzandeh, S., Moradi, P., and Jalili, M. (2025). Recommender systems based on non-negative matrix factorization: A survey. *Transactions on Artificial Intelligence*.

Ahmed, S. F., Alam, M. S. B., Hassan, M., Rozbu, M. R., Ishtiak, T., Rafa, N., Mofijur, M., Shawkat Ali, A., and Gandomi, A. H. (2023). Deep learning modelling techniques: current progress, applications, advantages, and challenges. *Artificial Intelligence Review*.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. (2022). Flamingo: a visual language model for few-shot learning. In *Proc. of NeurIPS*.

Anand, V. and Maurya, A. K. (2025). A survey on recommender systems using graph neural network. *Transactions on Information Systems*.

Asif, N. A., Sarker, Y., Chakrabortty, R. K., Ryan, M. J., Ahamed, M. H., Saha, D. K., Badal, F. R., Das, S. K., Ali, M. F., Moyeen, S. I., et al. (2021). Graph neural network: A comprehensive review on non-euclidean space. *Access*.

Ayemowa, M. O., Ibrahim, R., and Bena, Y. A. (2024). A systematic review of the literature on deep learning approaches for cross-domain recommender systems. *Decision Analytics Journal*.

Bao, H., Wang, W., Dong, L., Liu, Q., Mohammed, O. K., Aggarwal, K., Som, S., Piao, S., and Wei, F. (2022). VLMo: Unified vision-language pre-training with mixture-of-modality-experts. In *Proc. of NeurIPS*.

Bao, K., Zhang, J., Zhang, Y., Wang, W., Feng, F., and He, X. (2023). TallRec: An effective and efficient tuning framework to align large language model with recommendation. In *Proc. of RecSys*.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*.

Bolya, D., Fu, C.-Y., Dai, X., Zhang, P., Feichtenhofer, C., and Hoffman, J. (2022). Token merging: Your ViT but faster. In *Proc. of ICML*.

Borko, H. (1962). *Evaluating the effectiveness of information retrieval systems*. System Development Corporation.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. In *Proc. of NeurIPS*.

Buttcher, S., Clarke, C. L., and Cormack, G. V. (2016). *Information retrieval: Implementing and evaluating search engines*. MIT Press.

Cai, X., Xia, L., Ren, X., and Huang, C. (2023). How expressive are graph neural networks in recommendation. In *Proc. of CIKM*.

Chen, J., Wu, S., Gupta, A., and Ying, R. (2023). D4explainer: In-distribution gnn explanations via discrete denoising diffusion. In *Proc. of NeurIPS*.

Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. (2020). Simple and deep graph convolutional networks. In *Proc. of ICML*.

Chen, X., Chen, H., Xu, H., Zhang, Y., Cao, Y., Qin, Z., and Zha, H. (2019). Personalised fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proc. of SIGIR*.

Chen, Z., Gan, W., Wu, J., Hu, K., and Lin, H. (2024). Data scarcity in recommendation systems: A survey. *Transactions on Recommender Systems*.

Chikwendu, I. A., Zhang, X., Agyemang, I. O., Adjei-Mensah, I., Chima, U. C., and Ejiyi, C. J. (2023). A comprehensive survey on deep graph representation learning methods. *Journal of Artificial Intelligence Research*, 78:287–356.

Chronopoulou, A., Peters, M. E., and Dodge, J. (2022). Efficient hierarchical domain adaptation for pretrained language models. In *Proc. of ACL*.

Chronopoulou, A., Peters, M. E., Fraser, A., and Dodge, J. (2023). Adaptersoup: Weight averaging to improve generalisation of pretrained language models. In *Proc. of ACL (Findings)*.

Croce, F., Rebuffi, S.-A., Shelhamer, E., and Gowal, S. (2023). Seasoning model soups for robustness to adversarial and natural distribution shifts. In *Proc. of CVPR*.

Cui, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. (2022). M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084*.

De Gemmis, M., Lops, P., Musto, C., Narducci, F., and Semeraro, G. (2015). Semantics-aware content-based recommender systems. In *Recommender systems handbook*.

Deldjoo, Y., Schedl, M., Cremonesi, P., and Pasi, G. (2020). Recommender systems leveraging multimedia content. *Computing Surveys*.

Deng, Y. (2022). Recommender systems based on graph embedding techniques: A review. *Access*.

Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. In *Proc. of NeurIPS*.

Diao, S., Xu, T., Xu, R., Wang, J., and Zhang, T. (2023). Mixture-of-domain-adapters: Decoupling and injecting domain knowledge to pre-trained language models' memories. In *Proc. of ACL*.

Ding, K., Xu, Z., Tong, H., and Liu, H. (2022). Data augmentation for deep graph learning: A survey. *SIGKDD Explorations Newsletter*.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *Proc. of ICML*.

Du, X., Wang, X., He, X., Li, Z., Tang, J., and Chua, T.-S. (2020). How to learn item representation for cold-start multimedia recommendation? In *Proc. of MM*.

Dwivedi, V. P. and Bresson, X. (2020). A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*.

Dwivedi, V. P., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. (2022). Graph neural networks with learnable structural and positional representations. In *Proc. of ICLR*.

Faghri, F., Fleet, D. J., Kiros, J. R., and Fidler, S. (2018). VSE++: Improving visual-semantic embeddings with hard negatives. In *Proc. of BMVC*.

Fan, G., He, X., Xu, T., Wu, F., Liu, Y., He, M., and Hu, G. (2022). Contrastive clustering for unsupervised learning. In *Proc. of CVPR)*.

Fang, T., Zhang, Y., Yang, Y., and Wang, C. (2022). Prompt tuning for graph neural networks. *arXiv preprint arXiv:2209.15240*.

Feng, A., Li, I., Jiang, Y., and Ying, R. (2023). Diffuser: efficient transformers with multi-hop attention diffusion for long sequences. In *Proc. of AAAI*.

Fuhr, N. (2018). Some common mistakes in IR evaluation, and how they can be avoided. In *Proc. of SIGIR forum*.

Gao, C., Wang, X., He, X., and Li, Y. (2022). Graph neural networks for recommender system. In *Proc. of WSDM*.

Gao, C., Zheng, Y., Li, N., Li, Y., Qin, Y., Piao, J., Quan, Y., Chang, J., Jin, D., He, X., et al. (2023). A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *Transactions on Recommender Systems*.

Garapati, R. and Chakraborty, M. (2025). Recommender systems in the digital age: A comprehensive review of methods, challenges, and applications. *Knowledge and Information Systems*.

Gasteiger, J., Bojchevski, A., and Günnemann, S. (2018). Predict then propagate: Graph neural networks meet personalized pagerank. In *Proc of ICLR*.

Gasteiger, J., Weißenberger, S., and Günnemann, S. (2019). Diffusion improves graph learning. In *Proc. of NeurIPS*.

Ge, X., Xu, S., Chen, F., Wang, J., Wang, G., An, S., and Jose, J. M. (2024). 3shnet: Boosting image–sentence retrieval via visual semantic–spatial self-highlighting. *Information Processing & Management*.

Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. (2017). Audio set: An ontology and human-labeled dataset for audio events. In *Proc. of ICASSP*.

Geng, S., Liu, S., Fu, Z., Ge, Y., and Zhang, Y. (2022). Recommendation as language processing (RLP): A unified pretrain, personalised prompt & predict paradigm (P5). In *Proc. of RecSys*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *Proc. of ICLR*.

Gouk, H., Hospedales, T. M., and Pontil, M. (2021). Distance-based regularisation of deep networks for fine-tuning. In *Proc. of ICLR*.

Gui, J., Chen, T., Zhang, J., Cao, Q., Sun, Z., Luo, H., and Tao, D. (2024). A survey on self-supervised learning: Algorithms, applications, and future trends. *Transactions on Pattern Analysis and Machine Intelligence*.

Gunel, B., Du, J., Conneau, A., and Stoyanov, V. (2020). Supervised contrastive learning for pre-trained language model fine-tuning. *arXiv preprint arXiv:2011.01403*.

Guo, J., Du, L., Liu, H., Zhou, M., He, X., and Han, S. (2023). Gpt4graph: Can large language models understand graph structured data? an empirical evaluation and benchmarking. *arXiv preprint arXiv:2305.15066*.

Gupta, S., Kutlu, M., Khetan, V., and Lease, M. (2019). Correlation, prediction and ranking of evaluation metrics in information retrieval. In *Proc. of ECIR*.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., and Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. In *Proc. of ACL*.

Haefeli, K. K., Martinkus, K., Perraudin, N., and Wattenhofer, R. (2022). Diffusion models for graphs benefit from discrete state spaces. In *Proc. of LoG*.

Hamilton, W., Ying, Z., and Leskovec, J. (2017). Inductive representation learning on large graphs. *Proc. of NeurIPS*.

Han, C., Castells, P., Gupta, P., Xu, X., and Salaka, V. (2022). Addressing cold start in product search via empirical bayes. In *Proc. of CIKM*.

Han, H., Wang, Y., Shomer, H., Guo, K., Ding, J., Lei, Y., Halappanavar, M., Rossi, R. A., Mukherjee, S., Tang, X., et al. (2024). Retrieval-augmented generation with graphs (GraphRAG). *arXiv preprint arXiv:2501.00309*.

Hasan, E., Rahman, M., Ding, C., Huang, J., and Raza, S. (2024). Based recommender systems: a survey of approaches, challenges and future perspectives. *Computing Surveys*.

Hassani, K. and Khasahmadi, A. H. (2020). Contrastive multi-view representation learning on graphs. In *Proc. of ICML*.

Haveliwala, T. H. (2002). Topic-sensitive PageRank. In *Proc. of WWW*.

He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., and Neubig, G. (2022). Towards a unified view of parameter-efficient transfer learning. In *Proc. of ICLR*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proc. of CVPR*.

He, R. and McAuley, J. (2016a). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proc. of WWW*.

He, R. and McAuley, J. (2016b). Vbpr: Visual bayesian personalised ranking from implicit feedback. In *Proc. of AAAI*.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., and Wang, M. (2020). LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proc. of SIGIR*.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S. (2017). Neural collaborative filtering. In *Proc. of WWW*.

He, X. and Wang, X. E. (2023). Multi-modal graph transformer for multimodal question answering. In *Proc. of ACL*.

Hidasi, B., Karatzoglou, A., Baltrunas, L., and Tikk, D. (2015). Session-based recommendations with recurrent neural networks. In *Proc. of ICLR*.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. In *Proc. of NeurIPS*.

Hodson, T. O. (2022). Root mean square error (RMSE) or mean absolute error (MAE): When to use them or not. *Geoscientific Model Development Discussions*.

Hou, Z., Liu, X., Cen, Y., Dong, Y., Yang, H., Wang, C., and Tang, J. (2022). Graphmae: Self-supervised masked graph autoencoders. In *Proc. of KDD*.

Hu, G., Zhang, Y., and Yang, Q. (2018). Conet: Collaborative cross networks for cross-domain recommendation. In *Proc of CIKM*.

Hu, L., Xu, S., Li, C., Yang, C., Shi, C., Duan, N., Xie, X., and Zhou, M. (2020). Graph neural news recommendation with unsupervised preference disentanglement. In *Proc. of ACL*.

Hua, C., Luan, S., Xu, M., Ying, R., Fu, J., Ermon, S., and Precup, D. (2023). MUDiff: Unified diffusion for complete molecule generation. In *Proc. of PMLR*.

Huang, C., Wang, X., He, X., and Yin, D. (2022). Self-supervised learning for recommender system. In *Proc. of SIGIR*.

Huang, J. and Chang, K. C.-C. (2023). Towards reasoning in large language models: A survey. In *Proc. of ACL (Findings)*.

Jaiswal, A. K., Liu, S., Chen, T., Ding, Y., and Wang, Z. (2023). Instant soup: Cheap pruning ensembles in a single pass can draw lottery tickets from large models. In *Proc. of ICML*.

Javed, U., Shaukat, K., Hameed, I. A., Iqbal, F., Alam, T. M., and Luo, S. (2021). A review of content-based and context-based recommendation systems. *International Journal of Emerging Technologies in Learning*.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., et al. (2024). Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Jiang, J., Zhou, K., Dong, Z., Ye, K., Zhao, W. X., and Wen, J.-R. (2023). Structgpt: A general framework for large language model to reason over structured data. In *Proc of EMNLP*.

Jin, W., Cheng, Y., Shen, Y., Chen, W., and Ren, X. (2022). A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In *Proc. of ACL*.

Jin, Y. and Zhu, X. (2025). Oversmoothing alleviation in graph neural networks: a survey and unified view. *Knowledge and Information Systems*.

Jing, L. and Tian, Y. (2020). Self-supervised visual feature learning with deep neural networks: A survey. *Transactions on Pattern Analysis and Machine Intelligence*.

Jing, M., Zhu, Y., Zang, T., and Wang, K. (2023). Contrastive self-supervised learning in recommender systems: A survey. *Transactions on Information Systems*.

Joshi, C. (2020). Transformers are graph neural networks. *The Gradient*.

Ju, W., Fang, Z., Gu, Y., Liu, Z., Long, Q., Qiao, Z., Qin, Y., Shen, J., Sun, F., Xiao, Z., et al. (2024). A comprehensive survey on deep graph representation learning. *Neural Networks*.

Jung, H. and Park, H. (2025). Balancing graph embedding smoothness in self-supervised learning via information-theoretic decomposition. In *Proc. of WWW*.

Kaminskas, M. and Ricci, F. (2012). Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review*.

Kang, W.-C. and McAuley, J. (2018). Self-attentive sequential recommendation. In *Proc. of ICDM*.

Kanwal, S., Nawaz, S., Malik, M. K., and Nawaz, Z. (2021). A review of text-based recommendation systems. *Access*.

Kenton, J. D. M.-W. C. and Toutanova, L. K. (2019a). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*.

Kenton, J. D. M.-W. C. and Toutanova, L. K. (2019b). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

Kim, T., Lee, Y.-C., Shin, K., and Kim, S.-W. (2022). Mario: Modality-aware attention and modality-preserving decoders for multimedia recommendation. In *Proc. of CIKM*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Kolesnikov, A., Dosovitskiy, A., Weissenborn, D., Heigold, G., Uszkoreit, J., Beyer, L., Minderer, M., Dehghani, M., Houlsby, N., Gelly, S., Unterthiner, T., and Zhai, X. (2022). An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. of ICLR*.

Kong, L., Cui, J., Sun, H., Zhuang, Y., Prakash, B. A., and Zhang, C. (2023). Autoregressive diffusion model for graph generation. In *Proc. of ICML*.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*.

Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., and Tossou, P. (2021). Rethinking graph transformers with spectral attention. In *Proc. of NeurIPS*.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proc. of ICML*.

Lee, D., Kang, S., Ju, H., Park, C., and Yu, H. (2021a). Bootstrapping user and item representations for one-class collaborative filtering. In *Proc. of SIGIR*.

Lee, J. D., Lei, Q., Saunshi, N., and Zhuo, J. (2021b). Predicting what you already know helps: Provable self-supervised learning. In *Proc. of NeurIPS*.

Lei, B., Liao, C., Ding, C., et al. (2023). Boosting logical reasoning in large language models through a new framework: The chain of thought. *arXiv preprint arXiv:2308.08614*.

Lester, B., Al-Rfou, R., and Constant, N. (2021). The power of scale for parameter-efficient prompt tuning. In *Proc. of EMNLP*.

Li, C., Xia, L., Ren, X., Ye, Y., Xu, Y., and Huang, C. (2023a). Graph transformer for recommendation. In *Proc. of SIGIR*.

Li, J., Li, D., Xiong, C., and Hoi, S. (2022). BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proc. of ICML*.

Li, J., Zhang, Q., Liu, W., Chan, A. B., and Fu, Y.-G. (2024). Another perspective of over-smoothing: Alleviating semantic over-smoothing in deep gnns. *Transactions on Neural Networks and Learning Systems*.

Li, J., Zhang, X., Wang, M., Han, B., and Liu, Y. (2020a). Global-local gnn: A graph neural network with global and local information propagation. In *Proc. of CIKM*.

Li, P., Wang, Y., Wang, H., and Leskovec, J. (2020b). Distance encoding–design provably more powerful GNNs for structural representation learning. In *Proc. of NeurIPS*.

Li, Q., Han, Z., and Wu, X.-M. (2018). Deeper insights into graph convolutional networks for semi-supervised learning. In *Proc. of AAAI*.

Li, X. L. and Liang, P. (2021). Prefix-tuning: Optimising continuous prompts for generation. In *Proc. of ACL*.

Li, Z. (2024). Construction of multi modal information knowledge graph for e-commmerce products based on pre-trained model. In *Proc. of ICCDA*.

Li, Z., Sun, A., and Li, C. (2023b). Diffurec: A diffusion model for sequential recommendation. *Transactions on Information Systems*.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. (2018). Variational autoencoders for collaborative filtering. In *Proc. of WWW*.

Lim, D., Robinson, J. D., Zhao, L., Smidt, T., Sra, S., Maron, H., and Jegelka, S. (2022). Sign and basis invariant networks for spectral graph representation learning. In *Proc. of ICLR*.

Lin, X., Wang, W., Li, Y., Feng, F., Ng, S.-K., and Chua, T.-S. (2024). Bridging items and language: A transition paradigm for large language model-based recommendation. In *Proc. of SIGKDD*.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. (2024a). Visual instruction tuning. In *Proc. of NeurIPS*.

Liu, J., Wang, H., Ma, H., Li, P., Lin, Z., Zhang, J., Zhu, Y., Wang, W., and Zhang, X. (2021a). Exploring the learning behaviors of self-supervised recommender systems. In *Proc. of SIGIR*.

Liu, M., Li, J., Li, G., and Pan, P. (2020a). Cross domain recommendation via bi-directional transfer graph collaborative filtering networks. In *Proc. of CIKM*.

Liu, Q., Hu, J., Xiao, Y., Zhao, X., Gao, J., Wang, W., Li, Q., and Tang, J. (2024b). Multimodal recommender systems: A survey. *Computing Surveys*.

Liu, Q., Yan, F., Zhao, X., Du, Z., Guo, H., Tang, R., and Tian, F. (2023). Diffusion augmentation for sequential recommendation. In *Proc. of CIKM*.

Liu, W., Zhang, Z., Li, X., Hu, J., Luo, Y., and Du, J. (2024c). Enhancing recommendation systems with gnns and addressing over-smoothing. In *Proc. of EIECC*.

Liu, X., Tao, Z., Shao, J., Yang, L., and Huang, X. (2022a). Elimrec: Eliminating single-modal bias in multimedia recommendation. In *Proc. of MM*.

Liu, X., Zhang, F., Hou, Z., Mian, L., Wang, Z., Zhang, J., and Tang, J. (2021b). Self-supervised learning: Generative or contrastive. *Transactions on Knowledge and Data Engineering*.

Liu, Y., Jin, M., Pan, S., Zhou, C., Zheng, Y., Xia, F., and Yu, P. S. (2022b). Graph self-supervised learning: A survey. *Transactions on knowledge and data engineering*.

Liu, Y., Yang, S., Lei, C., Wang, G., Tang, H., Zhang, J., Sun, A., and Miao, C. (2021c). Pre-training graph transformer with multimodal side information for recommendation. In *Proc. of MM*.

Liu, Y., Yi, L., Zhang, S., Fan, Q., Funkhouser, T., and Dong, H. (2020b). P4contrast: Contrastive learning with pairs of point-pixel pairs for rgb-d scene understanding. *arXiv preprint arXiv:2012.13089*.

Liu, Z., Chen, Y., Li, J., Yu, P. S., McAuley, J., and Xiong, C. (2021d). Contrastive self-supervised sequential recommendation with robust augmentation. *arXiv preprint arXiv:2108.06479*.

Liu, Z., Li, J., Xie, H., Li, P., Ge, J., Liu, S.-A., and Jin, G. (2024d). Towards balanced alignment: Modal-enhanced semantic modeling for video moment retrieval. In *Proc. of AAAI*.

Liu, Z., Ma, Y., Schubert, M., Ouyang, Y., and Xiong, Z. (2022c). Multi-modal contrastive pre-training for recommendation. In *Proc. of ICMR*.

Loni, B., Shi, Y., Larson, M., and Hanjalic, A. (2014). Cross-domain collaborative filtering with factorisation machines. In *Proc. of ECIR*.

Loukas, A. (2020). What graph neural networks cannot learn: Depth vs width. In *Prof. of ICLR*.

Lu, J., Batra, D., Parikh, D., and Lee, S. (2019). VilBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Proc. of NeurIPS*.

Luo, L., Li, Y., Gao, B., Tang, S., Wang, S., Li, J., Zhu, T., Liu, J., Li, Z., and Pan, S. (2023). MAMDR: A model agnostic learning method for multi-domain recommendation. In *Proc. of ICDE*.

Luo, Z. and Luo, Y. (2020). Feature-based collaborative filtering for recommender systems. In *Proc. of CIKM*.

Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. H. (2018). Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proc. of SIGKDD*.

Ma, S. and Liu, J.-w. (2023). Self-supervised contrastive learning for heterogeneous graph based on multi-pretext tasks. *Neural Computing and Applications*.

Man, T., Shen, H., Jin, X., and Cheng, X. (2017). Cross-domain recommendation: an embedding and mapping approach. In *Proc. of IJCAI*.

Mao, K., Zhu, J., Xiao, X., Lu, B., Wang, Z., and He, X. (2021). Ultragcn: Ultra simplification of graph convolutional networks for recommendation. In *Proc. of CIKM*.

Marcuzzo, M., Zangari, A., Albarelli, A., and Gasparetto, A. (2022). Recommendation systems: An insight into current development and future research challenges. *Access*.

Matena, M. S. and Raffel, C. A. (2022). Merging models with fisher-weighted averaging. In *Proc. of NeurIPS*.

McAuley, J., Targett, C., Shi, Q., and Van Den Hengel, A. (2015). Image-based recommendations on styles and substitutes. In *Proc. of SIGIR*.

Meng, Z., Liu, S., Macdonald, C., and Ounis, I. (2021). Graph neural pre-training for enhancing recommendations using side information. *Transitions on Information Systems*.

Meng, Z., Ounis, I., Macdonald, C., and Yi, Z. (2024a). Knowledge graph cross-view contrastive learning for recommendation. In *Proc. of ECIR*.

Meng, Z., Yi, Z., and Ounis, I. (2024b). Knowledge-enhanced multi-behaviour contrastive learning for effective recommendation. In *Proc. of RecSys*.

Mialon, G., Chen, D., Selosse, M., and Mairal, J. (2021). Graphit: Encoding graph structure in transformers. *arXiv preprint arXiv:2106.05667*.

Milogradskii, A., Lashinin, O., P, A., Ananyeva, M., and Kolesnikov, S. (2024). Revisiting bpr: A replicability study of a common recommender system baseline. In *Proc. of RecSys*.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. (2019). Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proc. of AAAI*.

Murphy, R., Srinivasan, B., Rao, V., and Ribeiro, B. (2019). Relational pooling for graph representations. In *Proc. of ICML*.

Navigli, R., Conia, S., and Ross, B. (2023). Biases in large language models: origins, inventory, and discussion. *Journal of Data and Information Quality*.

Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proc. of EMNLP-IJCNLP*.

Ning, W., Yan, X., Liu, W., Cheng, R., Zhang, R., and Tang, B. (2023). Multi-domain recommendation with embedding disentangling and domain alignment. In *Proc. of CIKM*.

Ortiz-Jimenez, G., Modas, A., Moosavi, S.-M., and Frossard, P. (2020). Neural anisotropy directions. In *Proc. of NeurIPS*.

Pan, X., Chen, Y., Tian, C., Lin, Z., Wang, J., Hu, H., and Zhao, W. X. (2022). Multimodal meta-learning for cold-start sequential recommendation. In *Proc. of CIKM*.

Panda, D. K. and Ray, S. (2022). Approaches and algorithms to mitigate cold start problems in recommender systems: a systematic literature review. *Journal of Intelligent Information Systems*.

Qian, S., Pham, V. H., Lutellier, T., Hu, Z., Kim, J., Tan, L., Yu, Y., Chen, J., and Shah, S. (2021). Are my deep learning systems fair? an empirical study of fixed-seed training. In *Proc. of NeuIPS*.

Qin, G. and Eisner, J. (2021). Learning how to ask: Querying lms with mixtures of soft prompts. In *Proc. of NAACL*.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *Proc. of ICML*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multi-task learners. *OpenAI Blog*.

Rao, J., Wang, F., Ding, L., Qi, S., Zhan, Y., Liu, W., and Tao, D. (2022). Where does the performance improvement come from? a reproducibility concern about image-text retrieval. In *Proc. of SIGIR*.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using siamese bert-networks. In *Proc. of EMNLP-IJCNLP*.

Ren, X., Wei, W., Xia, L., and Huang, C. (2025). A comprehensive survey on self-supervised learning for recommendation. *Computing Surveys*.

Ren, X., Xia, L., Yang, Y., Wei, W., Wang, T., Cai, X., and Huang, C. (2024). Sslrec: A self-supervised learning framework for recommendation. In *Proc. of ICDM*.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *Proc. of UAI*.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proc. of the CVPR*.

Roy, S. S., Kumar, A., and Kumar, R. S. (2024). Metadata and review based hybrid apparel recommendation system using cascaded large language models. *Access*.

Sarker, I. H. (2021). Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*.

Schuirmann, D. J. (1987). A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of Pharmacokinetics and Biopharmaceutics*.

Sguerra, B., Tran, V.-A., Hennequin, R., and Moussallam, M. (2025). Uncertainty in repeated implicit feedback as a measure of reliability. In *Proc. of UMAP*.

Simmons-Mackie, N., Worrall, L., Murray, L. L., Enderby, P., Rose, M. L., Paek, E. J., and Klippi, A. (2017). The top ten: Best practice recommendations for aphasia. *Aphasiology*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proc. of ICLR*.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. of ICML*.

Stickland, A. C., Bérard, A., and Nikoulina, V. (2021). Multilingual domain adaptation for nmt: Decoupling language and domain information with adapters. In *Proc. of ACL*.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*.

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., and Jiang, P. (2019). BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proc. of CIKM*.

Sun, M., Zhou, K., He, X., Wang, Y., and Wang, X. (2022). Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *Proc. of KDD*.

Sun, R., Cao, X., Zhao, Y., Wan, J., Zhou, K., Zhang, F., Wang, Z., and Zheng, K. (2020). Multi-modal knowledge graphs for recommender systems. In *Proc. of CIKM*.

Suresh, R., Alim, K., Das, I., Sun, Y., Dong, H., and Li, R. (2021). Adaptive graph-based learning for malicious account detection. In *Proc. of ICCV Workshops*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proc. of CVPR*.

Tailor, S. A., Opolka, F. L., Lio, P., and Lane, N. D. (2022). Do we need anisotropic graph neural networks? In *Proc. of ICLR*.

Tang, H., Liu, J., Zhao, M., and Gong, X. (2020). Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Proc. of RecSys*.

Tang, J., Jin, H., Tan, S., and Liang, D. (2016). Cross-domain action recognition via collective matrix factorization with graph laplacian regularization. *Image and Vision Computing*.

Tang, J., Yang, Y., Wei, W., Shi, L., Su, L., Cheng, S., Yin, D., and Huang, C. (2024). Graphgpt: Graph instruction tuning for large language models. In *Proc. of SIGIR*.

Tao, Z., Liu, X., Xia, Y., Wang, X., Yang, L., Huang, X., and Chua, T.-S. (2022). Self-supervised learning for multimedia recommendation. *Transactions on Multimedia*.

Thakoor, S., Jin, H., Chen, K., Zhang, X., Hsieh, C.-Y., Liang, J., Han, J., Hsieh, C.-H., Wu, Y., and Zheng, T. (2021). Bootstrap your own latent graph: Towards general graph contrastive learning. In *Proc. of ICLR*.

Tian, Y., Krishnan, D., and Isola, P. (2020). Contrastive multiview coding. In *Proc. of ECCV*.

Torgo, L. and Ribeiro, R. (2009). Precision and recall for regression. In *Proc. of ICDS*.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Proc. of NeurIPS*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks. In *Proc. of ICLR*.

Veličković, P., Fedus, W., Liò, P., Lučić, D., Kavukcuoglu, K., and Bengio, Y. (2019). Deep graph infomax. In *Proc. of ICLR*.

Venice, J. A., Arivazhagan, D., Suman, N., Shanthi, H., and Swadhi, R. (2025). Recommendation systems and content personalisation: Algorithms, applications, and adaptive learning. In *AI for Large Scale Communication Networks*.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. (2023). Digress: Discrete denoising diffusion for graph generation. In *Proc. of ICLR*.

Volkovs, M., Yu, G. W., and Poutanen, T. (2017). Content-based neighbor models for cold start in recommender systems. In *Proc. of RecSys*.

Walker, J., Zhong, T., Zhang, F., Gao, Q., and Zhou, F. (2022). Recommendation via collaborative diffusion generative model. In *Proc. of KSEM*.

Wang, C., Liang, Y., Liu, Z., Zhang, T., and Philip, S. Y. (2021a). Pre-training graph neural network for cross-domain recommendation. In *Proc. of CogMI*.

Wang, G., Ying, R., Huang, J., and Leskovec, J. (2020a). Multi-hop attention graph neural network. *arXiv preprint arXiv:2009.14332*.

Wang, H., Le, Z., and Gong, X. (2020b). Recommendation system based on heterogeneous feature: A survey. *Access*.

Wang, J., Sun, G., Wang, P., Liu, D., Dianat, S., Rabbani, M., Rao, R., and Tao, Z. (2024). Text is mass: Modeling as stochastic embedding for text-video retrieval. In *Proc. of CVPR*.

Wang, J., Zeng, Z., Wang, Y., Wang, Y., Lu, X., Li, T., Yuan, J., Zhang, R., Zheng, H.-T., and Xia, S.-T. (2023a). MISSRec: Pre-training and transferring multi-modal interest-aware sequence representation for recommendation. In *Proc. of MM*.

Wang, J., Zhu, J., and He, X. (2021b). Cross-batch negative sampling for training two-tower recommenders. In *Proc. of SIGIR*.

Wang, K., Zhu, Y., Zang, T., Wang, C., Liu, K., and Ma, P. (2023b). Multi-aspect graph contrastive learning for review-enhanced recommendation. *Transactions on Information Systems*.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. (2020c). Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Wang, W., Bao, H., Dong, L., Bjorck, J., Peng, Z., Liu, Q., Aggarwal, K., Mohammed, O. K., Singhal, S., Som, S., et al. (2023c). Image as a foreign language: Beit pretraining for all vision and vision-language tasks. In *Proc. of CVPR*.

Wang, W., Bao, H., Dong, L., Bjorck, J., Peng, Z., Liu, Q., Aggarwal, K., Mohammed, O. K., Singhal, S., Som, S., et al. (2023d). Image as a foreign language: Beit pretraining for vision and vision-language tasks. In *Proc. of CVPR*.

Wang, W., Feng, F., He, X., Nie, L., and Chua, T.-S. (2021c). Denoising implicit feedback for recommendation. In *Proc. of WSDM*.

Wang, W., Xu, Y., Feng, F., Lin, X., He, X., and Chua, T.-S. (2023e). Diffusion recommender model. In *Proc. of SIGIR*.

Wang, X., Chen, G., Qian, G., Gao, P., Wei, X.-Y., Wang, Y., Tian, Y., and Gao, W. (2023f). Large-scale multi-modal pre-trained models: A comprehensive survey. *Machine Intelligence Research*.

Wang, X., He, X., Cao, Y., Liu, M., and Chua, T.-S. (2019a). KGAT: Knowledge graph attention network for recommendation. In *Proc. of SIGKDD*.

Wang, X., He, X., Wang, M., Feng, F., and Chua, T.-S. (2019b). Neural graph collaborative filtering. In *Proc. of SIGIR*.

Wang, X., Huang, T., Wang, D., Yuan, Y., Liu, Z., He, X., and Chua, T.-S. (2021d). Learning intents behind interactions with knowledge graph for recommendation. In *Proc. of WWW*.

Wang, X., Ounis, I., and Macdonald, C. (2021e). Leveraging review properties for effective recommendation. In *Proc. of WWW*.

Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., and Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. In *Proc. of ICLR*.

Wang, Y., Yu, H., Wang, G., and Xie, Y. (2020d). Cross-domain recommendation based on sentiment analysis and latent feature mapping. *Entropy*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. In *Proc. of NeurIPS*.

Wei, Y., Liu, W., Liu, F., Wang, X., Nie, L., and Chua, T.-S. (2023). Lightgt: A light graph transformer for multimedia recommendation. In *Proc. of SIGIR*.

Wei, Y., Wang, X., Nie, L., He, X., Hong, R., and Chua, T.-S. (2019). Mmgcn: Multi-modal graph convolution network for personalised recommendation of micro-video. In *Proc. of MM*.

Weisfeiler, B. and Leman, A. (1968). The reduction of a graph to canonical form and the algebra which appears therein. *NTI Series*.

Welling, M. and Kipf, T. N. (2016). Semi-supervised classification with graph convolutional networks. In *Proc. of ICLR*.

Wen, Z. and Fang, Y. (2023). Augmenting low-resource text classification with graph-grounded pre-training and prompting. In *Proc. of SIGIR*.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. (2022). Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proc. of ICML*.

Wu, F., Qiao, Y., Chen, J.-H., Wu, C., Qi, T., Lian, J., Liu, D., Xie, X., Gao, J., Wu, W., et al. (2020a). Mind: A large-scale dataset for news recommendation. In *Proc. of ACL*.

Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J., and Xie, X. (2021a). Self-supervised graph learning for recommendation. In *Proc. of SIGIR*.

Wu, J., Zhong, S.-H., and Liu, Y. (2019a). Mvsgcn: A novel graph convolutional network for multi-video summarization. In *Proc. of MM*.

Wu, L. (2002). A model for implementing bpr based on strategic perspectives: An empirical study. *Information & Management*.

Wu, L., Quan, C., Li, C., Wang, Q., Zheng, B., and Luo, X. (2019b). A context-aware user-item representation learning for item recommendation. *Transactions on Information Systems*.

Wu, Q., Yang, C., Zhao, W., He, Y., Wipf, D., and Yan, J. (2023). Difformer: Scalable (graph) transformers induced by energy constrained diffusion. In *Proc. of ICLR*.

Wu, S., Sun, F., Zhang, W., Xie, X., and Cui, B. (2022). Graph neural networks in recommender systems: a survey. *Computing Surveys*.

Wu, Y., DuBois, C., Zheng, A. X., and Ester, M. (2016). Collaborative denoising auto-encoders for top-n recommender systems. In *Proc. of WSDM*.

Wu, Y., Xie, R., Zhu, Y., Zhuang, F., Zhang, X., Lin, L., and He, Q. (2024). Personalized prompt for sequential recommendation. *Transactions on Knowledge and Data Engineering*.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. (2020b). A comprehensive survey on graph neural networks. *Transactions on Neural Networks and Learning Systems*.

Wu, Z., Shen, F., Wu, L., Xiao, H., Zhang, L., and Lin, G. (2021b). Handling information loss in graph neural networks for session-based recommendation. In *Proc. of MM*.

Xanthopoulos, P., Pardalos, P. M., Trafalis, T. B., Xanthopoulos, P., Pardalos, P. M., and Trafalis, T. B. (2013). Linear discriminant analysis. *Robust Data Mining*.

Xian, D., Cui, S., Wang, B., and Cui, L. (2023). H&M personalized fashion product recommendation using lightgbmranker. In *Proc, of ICBIS*.

Xiao, C., Xie, R., Yao, Y., Liu, Z., Sun, M., Zhang, X., and Lin, L. (2021). Uprec: User-aware pre-training for recommender systems. *arXiv preprint arXiv:2102.10989*.

Xie, Q., Hsieh, C.-Y., Hsieh, C.-H., Wu, Y., and Zheng, T. (2020a). Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*.

Xie, S., Gu, J., Guo, D., Qi, C. R., Guibas, L., and Litany, O. (2020b). Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Proc. of ECCV*.

Xie, X., Sun, F., Liu, Z., Wu, S., Gao, J., Zhang, J., Ding, B., and Cui, B. (2022a). Contrastive learning for sequential recommendation. In *Proc. of ICDE*.

Xie, Y., Xu, Z., Zhang, J., Wang, Z., and Ji, S. (2022b). Self-supervised learning of graph neural networks: A unified review. *Transactions on Pattern Analysis and Machine Intelligence*.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks? In *Proc. of ICLR*.

Xu, Y.-H., Wang, Z.-H., Wang, Z.-R., Fan, R., and Wang, X. (2023). A recommendation algorithm based on a self-supervised learning pretrain transformer. *Neural processing letters*.

Yago, H., Clemente, J., and Rodriguez, D. (2018). Competence-based recommender systems: a systematic literature review. *Behaviour & Information Technology*.

Yang, K. and Toni, L. (2018). Graph-based recommendation system. In *Proc. of GlobalSIP*.

Yang, R., Yang, Y., Zhou, F., and Sun, Q. (2023). Directional diffusion models for graph representation learning. In *Proc. of NeurIPS*.

Yang, Y., Huang, C., Xia, L., and Li, C. (2022a). Knowledge graph contrastive learning for recommendation. In *Proc. of SIGIR*.

Yang, Z., Ding, M., Xu, B., Yang, H., and Tang, J. (2022b). Stam: A spatiotemporal aggregation method for graph neural network-based recommendation. In *Proc. of WWW*.

Yao, T., Yi, X., Cheng, D. Z., Yu, F., Chen, T., Menon, A., Hong, L., Chi, E. H., Tjoa, S., Kang, J., et al. (2021). Self-supervised learning for large-scale item recommendations. In *Proc. of CIKM*.

Ye, H., Li, X., Yao, Y., and Tong, H. (2023). Towards robust neural graph collaborative filtering via structure denoising and embedding perturbation. *Transactions on Information Systems*.

Yi, Z., Long, Z., Ounis, I., Macdonald, C., and Mccreadie, R. (2025). Enhancing recommender systems: Deep modality alignment with large multi-modal encoders. *Transactions on Recommender Systems*.

Yi, Z. and Ounis, I. (2024). A unified graph transformer for overcoming isolations in multi-modal recommendation. In *Proc. of RecSys*.

Yi, Z., Ounis, I., and Macdonald, C. (2023a). Contrastive graph prompt-tuning for cross-domain recommendation. *Transactions on Information Systems*.

Yi, Z., Ounis, I., and Macdonald, C. (2023b). Graph contrastive learning with positional representation for recommendation. In *Proc. of ECIR*.

Yi, Z., Wang, X., and Ounis, I. (2024). A directional diffusion graph transformer for recommendation. *arXiv preprint arXiv:2404.03326*.

Yi, Z., Wang, X., Ounis, I., and Macdonald, C. (2022). Multi-modal graph contrastive learning for micro-video recommendation. In *Proc. of SIGIR*.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. (2021). Do transformers really perform badly for graph representation. In *Proc. of NeuIPS*.

You, J., Ying, R., and Leskovec, J. (2019). Position-aware graph neural networks. In *Proc. of ICML*.

You, Y., Chen, T., Shen, Y., Wang, Z., Schuurmans, D., and Gong, B. (2020). Graph contrastive learning with augmentations. In *Proc. of NeurIPS*.

Yu, J., He, R., and Ying, Z. (2023a). Thought propagation: An analogical approach to complex reasoning with large language models. In *Proc. of ICLR*.

Yu, J., Yin, H., Li, J., Wang, Q., Hung, N. Q. V., and Zhang, X. (2021). Self-supervised multi-channel hypergraph convolutional network for social recommendation. In *Proc. of WWW*.

Yu, J., Yin, H., Xia, X., Chen, T., Cui, L., and Nguyen, Q. V. H. (2022). Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proc. of SIGIR*.

Yu, J., Yin, H., Xia, X., Chen, T., Li, J., and Huang, Z. (2023b). Self-supervised learning for recommender systems: A survey. *Transactions on Knowledge and Data Engineering*.

Yu, X., Zhou, C., Fang, Y., and Zhang, X. (2024). Multigprompt for multi-task pre-training and prompting on graphs. In *Proc. of WWW*.

Yuan, C., Zhao, K., Kuruoglu, E. E., Wang, L., Xu, T., Huang, W., Zhao, D., Cheng, H., and Rong, Y. (2025). A survey of graph transformers: Architectures, theories and applications. *arXiv preprint arXiv:2502.16533*.

Yuan, F., Yao, L., and Benatallah, B. (2019). Darec: Deep domain adaptation for cross-domain recommendation via transferring rating patterns. In *Proc. of IJCAI*.

Yuan, X. (2024). Design of e-commerce personalized recommendation system based on multi-modal data fusion. In *Proc. of TELEPE*.

Yuan, Y., Chen, W., Yang, Y., and Wang, Z. (2020). In defense of the triplet loss again: Learning robust person re-identification with fast approximated triplet loss and label distillation. In *Proc. of CVPR*.

Yun, S., Jeong, M., Kim, R., Kang, J., and Kim, H. J. (2019). Graph transformer networks. In *Proc. of NeurIPS*.

Zang, T., Zhu, Y., Liu, H., Zhang, R., and Yu, J. (2022). A survey on cross-domain recommendation: taxonomies, methods, and future directions. *Transactions on Information Systems*.

Zeng, Z., Xiao, C., Yao, Y., Xie, R., Liu, Z., Lin, F., Lin, L., and Sun, M. (2021). Knowledge transfer via pre-training for recommendation: A review and prospect. *Frontiers in big Data*.

Zhai, X., Mustafa, B., Kolesnikov, A., and Beyer, L. (2023). Sigmoid loss for language image pre-training. In *Proc. of ICCV*.

Zhang, A., Chen, Y., Sheng, L., Wang, X., and Chua, T.-S. (2024a). On generative agents in recommendation. In *Proc. of SIGIR*.

Zhang, B., Fan, C., Liu, S., Huang, K., Zhao, X., Huang, J., and Liu, Z. (2024b). The expressive power of graph neural networks: A survey. *Transactions on Knowledge and Data Engineering*.

Zhang, F., Peng, Q., Wu, Y., Pan, Z., Zeng, R., Lin, D., and Qi, Y. (2022a). Multi-graph based multi-scenario recommendation in large-scale online video services. In *Proc. of WWW*.

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. (2016). Collaborative knowledge base embedding for recommender systems. In *Proc. of SIGKDD*.

Zhang, G., Li, D., Gu, H., Lu, T., and Gu, N. (2024c). Heterogeneous graph neural network with personalized and adaptive diversity for news recommendation. *Transactions on the Web*.

Zhang, S., Jiang, T., Kuang, K., Feng, F., Yu, J., Ma, J., Zhao, Z., Zhu, J., Yang, H., Chua, T.-S., et al. (2023a). Sled: Structure learning based denoising for recommendation. *Transactions on Information Systems*.

Zhang, S., Ma, X., Wang, Y., Zhou, Y., and Yu, D. (2022b). An embedding and interactions learning approach for id feature in deep recommender system. *Expert Systems with Applications*.

Zhang, S., Yao, L., Sun, A., and Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *Computing surveys*.

Zhang, T., Chen, C., Wang, D., Guo, J., and Song, B. (2023b). A VAE-based user preference learning and transfer framework for cross-domain recommendation. *Transactions on Knowledge and Data Engineering*.

Zhang, Y., Ding, H., Shui, Z., Ma, Y., Zou, J., Deoras, A., and Wang, H. (2021). Language models as recommender systems: Evaluations and limitations. In *Proc. of NeurIPS' I (Still) Can't Believe It's Not Better Workshop*.

Zhang Jinghao, Zhu Yanqiao, L. Q., Shu, W., Shuhui, W., and Liang, W. (2021). Mining latent structures for multimedia recommendation. In *Proc. of MM*.

Zhao, C., Li, C., and Fu, C. (2019). Cross-domain recommendation via preference propagation graphnet. In *Proc. of CIKM*.

Zhao, G., Zhang, X., Tang, H., Shen, J., and Qian, X. (2024). Domain-oriented knowledge transfer for cross-domain recommendation. *Transactions on Multimedia*.

Zhou, H., Zhou, X., Zeng, Z., Zhang, L., and Shen, Z. (2023a). A comprehensive survey on multi-modal recommender systems: Taxonomy, evaluation, and future directions. *arXiv preprint arXiv:2302.04473*.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020a). Graph neural networks: A review of methods and applications. *AI Open*.

Zhou, J., Xu, E., Wu, Y., and Li, T. (2025). Rescriber: Smaller-LLM-powered user-led data minimization for LLM-based chatbots. In *Proc. of CHI*.

Zhou, K., Wang, H., Zhao, W. X., Zhu, Y., Wang, S., Zhang, F., Wang, Z., and Wen, J.-R. (2020b). S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proc. of CIKM*.

Zhou, X. and Shen, Z. (2023). A tale of two graphs: Freezing and denoising graph structures for multimodal recommendation. In *Proc. of MM*.

Zhou, X., Zhou, H., Liu, Y., Zeng, Z., Miao, C., Wang, P., You, Y., and Jiang, F. (2023b). Bootstrap latent representations for multi-modal recommendation. In *Proc. of WWW*.

Zhou, X., Zhou, H., Liu, Y., Zeng, Z., Miao, C., Wang, P., You, Y., and Jiang, F. (2023c). Bootstrap latent representations for multi-modal recommendation. In *Proc. of WWW*.

Zhu, B., Lin, B., Ning, M., Yan, Y., Cui, J., HongFa, W., Pang, Y., Jiang, W., Zhang, J., Li, Z., et al. (2023). Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment. In *Proc. of ICLR*.

Zhu, F., Chen, C., Wang, Y., Liu, G., and Zheng, X. (2019). DTCTR: A framework for dual-target cross-domain recommendation. In *Proc. of CIKM*.

Zhu, F., Wang, Y., Chen, C., Zhou, J., Li, L., and Liu, G. (2021). Cross-domain recommendation: challenges, progress, and prospects. In *Proc. of IJCAI*.

Zhu, Y., Tang, Z., Liu, Y., Zhuang, F., Xie, R., Zhang, X., Lin, L., and He, Q. (2022). Personalized transfer of user preferences for cross-domain recommendation. In *Proc. of WSDM*.

Zhu, Y., Wang, C., Zhang, Q., and Xiong, H. (2024). Graph signal diffusion model for collaborative filtering. In *Proc. of SIGIR*.

Zhu, Y., Xu, Y., Shi, F., Lu, C., and Liu, Q. (2020). Graph contrastive learning with adaptive augmentation. In *Proc. of NeurIPS*.

Zivic, P., Vazquez, H., and Sánchez, J. (2024). Scaling sequential recommendation models with transformers. In *Proc. of SIGIR*.

Zong, Y., Mac Aodha, O., and Hospedales, T. (2024). Self-supervised multimodal learning: A survey. *Transactions on Pattern Analysis and Machine Intelligence*.

Zou, D., Wei, W., Mao, X.-L., Wang, Z., Qiu, M., Zhu, F., and Cao, X. (2022). Multi-level cross-view contrastive learning for knowledge-aware recommender system. In *Proc. of SIGIR*.

Zou, H., Kim, Z. M., and Kang, D. (2023). Diffusion models in nlp: A survey. *arXiv preprint arXiv:2305.14671*.