Multi-Task Learning for Effective Open-Retrieval Conversational Question Answering

Sarawoot Kongyoung

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

December 2023

Abstract

Conversational Question Answering (ConvQA) is a rapidly growing area of research that aims to improve the search experience for users by allowing for more natural interactions between users and search systems. ConvQA systems are designed to gauge and answer questions in the context of a conversation, taking into account the previous questions and answers in the dialogue. One of the challenges of ConvQA is resolving ambiguities in the user's questions based on the conversation history. This requires the system to not only consider the question being asked but to also take into account the conversation context to provide relevant and accurate answers. Open-Retrieval Conversational Question Answering (ORConvQA) is a more challenging variant of ConvQA, as it requires the system to retrieve relevant passages from a large collection of documents before extracting the required answers. This task requires the system to effectively search and retrieve the most relevant information, adding further complexity. In order to build an ORConvQA system, to address the ambiguities in conversational questions, a number of approaches have been proposed, such as follow-up question identification, conversational question rewriting, and asking clarifying questions. These approaches can help the system better gauge the user's intent and context, thereby allowing it to generate more precise and relevant responses. Another challenge in ORConvQA is retrieving relevant passages from a large collection of documents and identifying the most relevant ones based on the conversation context. This is important because the extracted answers need to be based on the relevant passages, in order to ensure accuracy. On the other hand, Multi-Task Learning (MTL) has emerged as a promising approach to facilitate the learning of multiple related tasks by sharing the learner structure in a single model. MTL has gained considerable attention in recent years due to its effectiveness in addressing a diverse range of complex problems within a unified model. Therefore, we argue that learning ORConvQA approaches simultaneously can help to improve the system's performance.

In this thesis, we propose a novel ORConvQA framework leveraging Multi-Task Learning (MTL) to improve the performance of multiple related tasks by sharing their learned structure. By applying MTL to ORConvQA, we aim to leverage the benefits of addressing several related tasks to build a more effective and efficient model that addresses two main challenges: (i) ambiguities in conversational questions; and (ii) retrieving relevant passages from a large collection of documents before extracting the answers.

To address ORConvQA effectively, we first propose an ORConvQA framework, which leverages a novel hybrid dynamic MTL method combining Abridged Linear for the main answer extraction task with a Loss-Balanced Task Weighting (LBTW) for the auxiliary related tasks, such as follow-up question identification, yes/no prediction, and unanswerable prediction, so as to automatically fine-tune task weighting during learning, ensuring that each of the tasks' weights is adjusted by the relative importance of the different tasks. We conduct experiments using QuAC, a large-scale ConvQA dataset. Our results demonstrate the effectiveness of our proposed method, which significantly outperforms both the single-task learning and existing static task weighting methods with improvements ranging from +2.72% to +3.20% in F1 scores. Our findings also show that the performance of using MTL in developing the ORConvQA model is sensitive to the correct selection of the auxiliary tasks as well as to an adequate balancing of the loss rates of these tasks during training by using LBTW.

To address the ambiguities in conversational questions, we propose the use of a text generation model with Multi-Task Learning for follow-up question identification and conversational question rewriting. Our derived models are based on text generation models –BART and T5–, and are trained to rewrite the conversational question and identify follow-up questions simultaneously. We evaluate our method using three test sets from the recent LIF (Learning to Identify Follow-up questions) dataset and a test set from the OR-QuAC dataset. Our results show that our proposed method significantly outperforms the single-task learning baselines on the LIF dataset, with statistically significant improvements ranging from +3.5% to +10.5% across all test sets, and also significantly outperforms the single-task learning of question rewriting models for passage retrieval on the OR-QuAC test set.

Next, we employ an approach for asking clarifying questions to further address the ambiguities in conversational questions by proposing a novel hybrid method combining the generation and selection processes. Our method leverages Multi-Task Learning, combining the tasks of clarification need classification and the generation of the clarifying question to simultaneously determine when the initial user's query necessitates a clarifying question and to generate a set of clarifying questions based on the user's initial query and conversation history. A selection model is used to select the relevant questions from a question pool. To rank the candidate clarifying questions obtained from both the selection and generation approaches, the questions are scored using a text generation model for question classification. By using both the generation and selection approaches, our proposed method is able to generate a comprehensive set of questions while still ensuring that the selected question is relevant to the user's queries. Our results on the TREC CAsT 2022 datasets demonstrate the effectiveness of our proposed method, which significantly outperforms existing strong baselines with improvements at P@1 by up to 20% on the relevance criteria and 30% on the novelty criteria.

Finally, to effectively address our second challenge of retrieving relevant passages from a large collection of documents and extracting the answers, we propose monoQA, which uses a

text generation model with Multi-Task Learning for both the reranker and reader. Our model, which is based on the T5 text generation model, is fine-tuned simultaneously for both reranking (in order to improve the precision of the top retrieved passages) and extracting the answer. Our results on the OR-QuAC and OR-CoQA datasets demonstrate the effectiveness of our proposed model, which significantly outperforms existing strong baselines with improvements ranging from +12.31% to +19.51% in MAP and from +5.70% to +23.34% in F1 on all used test sets.

Overall, this thesis contributes an effective ORConvQA framework leveraging Multi-Task Learning to address the challenges of resolving ambiguities in conversational questions and retrieving relevant passages from a large collection of documents. Our proposed framework significantly outperforms existing strong baselines on a variety of benchmark datasets, demonstrating the effectiveness of MTL in improving the performance of ORConvQA models.

Contents

List of Tables

List of Figures

Acknowledgements

I arrived in Glasgow over four years ago and still remember joining the Terrier team for their Christmas away day just the day after. The excitement of meeting IR professionals, my supervisors, and what would soon become a new group of friends is still fresh in my memory. Reflecting on these past four years, I want to express my deepest gratitude to several individuals who have provided immense support throughout my PhD.

First, I would like to express my sincere thanks to my supervisors, Iadh Ounis and Craig Macdonald, for their patience, enthusiasm, guidance, support, and encouragement. Their mentorship has been essential not only in my PhD research but also in my personal life, especially during the challenging times of the COVID-19 pandemic. Their assistance and understanding have been important in helping me navigate both my academic journey and the difficulties posed by the pandemic. Without their help, this work would not have been possible. I am deeply appreciative of their commitment and the important role they have played in my growth as a researcher and individual.

I am also grateful to my friends and colleagues at the Terrier team and the School of Computing Science, including Xiao Wang, Thomas Benedikt Janich, Sean Macavaney, Zixuan Yi, Zeyuan Meng, Graham McDonald, Richard McCreadie, Jeff Dalton, Siwei Liu, Yashon Wu, Javier Sanz-Cruzado Puig, Aleksandr Petrov, and many others. They have been very helpful and shared a lot of knowledge. Working with them has been a pleasure.

I owe a profound debt of gratitude to my mother, Thanom Kongyoung, whose unwavering love and support have been the cornerstone of my ability to complete this work. Despite being alone in Thailand throughout my PhD, and with my visits home being an impossibility for nearly four years, her strength and resilience have been a constant source of inspiration. Her enduring support has been my anchor during challenging times, and I am deeply appreciative of the sacrifices she has made, which have been instrumental in helping me persevere through this journey.

Last but not least, I hold a special place in my heart for my beloved cats - Mocca, Latte, Cappuccino, and Kope. To Latte, Cappuccino, and Kope, who are no longer with me, I deeply regret not being there in your final moments. Your memory remains a cherished part of my life. And to Mocca, I miss you immensely and look forward to the day when we can be together again.

Chapter 1

Introduction

1.1 Introduction

Information seeking is an important part of our daily lives, as we continually seek knowledge and answers to our information needs. Traditional search engines have been the go-to tool for information retrieval, where users input specific query text and retrieve a list of relevant documents or web pages. However, this interaction paradigm does not fully capture the natural and interactive nature of human information seeking (Zamani et al. 2022).

Conversational Question Answering (QA) has emerged as an exciting subfield within the realm of conversational search, which aims to bridge the gap between humans and machines by simulating natural dialogue interactions. Unlike conventional search engines, Conversational QA systems enable users to engage in conversation-like exchanges to obtain the desired information. By emulating human conversational patterns, these systems offer a more intuitive and user-friendly approach to information retrieval. For example, instead of submitting a search query like "Famous scientists" to a search engine, users can engage with a conversational system by asking questions such as "Who are some famous scientists?" and continue the conversation with follow-up enquiries like "What are their notable discoveries?" or "How did they contribute to their fields?" based on the system's previous responses. This conversational paradigm allows users to interact naturally, as if they were engaging in a dialogue with a knowledgeable person. The shift towards conversational information seeking is driven by the desire to enhance user experiences and to facilitate more effective and efficient access to knowledge (Zamani et al. 2022). By harnessing natural language understanding and dialogue techniques, Conversational QA systems aim to provide informative and contextually relevant responses that cater to the users' information needs.

The task of a Conversational QA system (Rajpurkar et al. 2018, 2016) is to effectively answer a user's questions by resolving any ambiguities that may arise from the posed questions based on the conversation history with the user (Choi et al. 2018, Reddy et al. 2019). This means that the system must not only consider the question being asked but it should also take into account

the conversation context to provide relevant and accurate responses (Kundu et al. 2020, Qu et al. 2020, Qu, Yang, Qiu, Croft, Zhang & Iyyer 2019, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019). Unlike traditional QA systems that work on a single-turn basis and require a clear and unambiguous question to return an answer, Conversational QA systems must be able to interpret and respond to questions in the context of the ongoing conversation. This makes the task more challenging, as the system must not only identify the information being sought but needs to also identify the user's intent and the background knowledge that the user has (Krasakis et al. 2020).

Open-Retrieval Conversational Question Answering (ORConvQA) is a challenging variant of the Conversational QA task due to the additional step of retrieving relevant passages from a large collection of documents before extracting the required answers (Qu et al. 2021, 2020). The complexity arises from the necessity of effectively searching and retrieving the most relevant information, making ORConvQA a demanding and intricate task. ORConvQA has various applications in domains such as education, healthcare, and customer support. For example, in healthcare, an ORConvQA system could be used to retrieve relevant medical research articles to assist doctors in making informed decisions. In education, an ORConvQA system could be used as a virtual assistant to answer the students' questions related to a particular topic. In customer support, an ORConvQA system could be used to retrieve relevant documents related to a customer's query, such as product manuals or user guides, so as to provide a more personalised and effective support experience. The ORConvQA system must identify the most relevant documents based on the conversation history and retrieve the passages that contain the most relevant information for answering the question.

To accomplish the ORConvQA task, the system must first understand the question based on the given conversational history (Qu et al. 2020). This requires the ORConvQA system to analyse the entire conversation and to interpret the user's intent to determine what information they are seeking. Once the ORConvQA system has a clear understanding of the question, it must retrieve the most relevant documents from a large collection of texts. This is typically done by applying a retrieval model that can rank the documents based on their relevance to the question.

After the relevant documents have been identified, the system must then extract the relevant passages that contain the information required to answer the question. This requires the system to analyse each document and to identify the most informative and relevant passages. The system may use various techniques to extract relevant passages, such as natural language processing, machine learning models, or information retrieval (Izacard & Grave 2021, Jiang et al. 2022*a*, Lee et al. 2022*a*, Qu et al. 2021, 2020, Wen et al. 2022).

In recent years, Multi-Task Learning (MTL) has emerged as a promising approach to improve the performance of machine learning systems by allowing them to learn multiple related tasks simultaneously. In the context of ORConvQA, MTL can be used to jointly optimise multiple objectives, such as conversational question answering, conversational question rewriting, passage retrieval, and passage reranking, leading to more effective and efficient ORConvQA systems.

This thesis explores the application of MTL techniques to improve the performance of ORConvQA systems. In particular, it investigates the benefits of jointly optimising multiple related tasks in ORConvQA, such as conversational question answering, conversational question rewriting, passage retrieval, and passage reranking, and explores various architectures and training strategies for MTL. Furthermore, the thesis also explores the effectiveness of transfer learning in ORConvQA, where a pre-trained model is used as a starting point for training ORConvQA systems. The use of pre-trained models can markedly reduce the amount of data required for training and improve the performance of ORConvQA systems. Overall, this thesis aims to investigate the potential of MTL and transfer learning in improving the performance of ORConvQA systems and proposes novel architectures and training strategies for effective ORConvQA. The thesis provides valuable insights into the development of more efficient and accurate open-retrieval conversational search systems.

1.2 Motivations

Open-Retrieval Conversational Question Answering (ORConvQA) is a Conversational Search task, where the passages need to be retrieved from a large collection of documents instead of being given as in a traditional Conversational Question Answering (ConvQA) (Choi et al. 2018, Reddy et al. 2019) task, before extracting the required answers. In order to effectively answer the user's questions, an ORConvQA system needs to understand the user's question based on a given conversational history, retrieve the relevant documents from a text collection, calculate the relevance score, and extract an answer from the retrieved passages.

Figure 1.1: An example dialogue.

One of the challenges of the ORConvQA task is that the system needs to correctly interpret a

question in the context of the previous conversation. Figure 1.1 illustrates how typical dialogues in conversational systems can lead to various ambiguities. For example, the user's utterance $U2$ highlights the importance of the conversation context in resolving potential ambiguities. Indeed, without knowing the specific city the user is referring to from the previous user utterance $U1$, the system cannot provide the correct answer. However, by analysing the entire conversation history, including the user's initial question and the subsequent responses (which also serve as the returned answers/passages), the system can better understand the user's needs and preferences and provide more accurate and helpful answers. These ambiguities can arise from different factors such as the user's unclear intent, use of ambiguous language, or incomplete information such as in the case of the user's utterance $U1$. Identifying and resolving these ambiguities is a critical challenge in developing effective conversational systems for tasks such as Open-Retrieval Conversational Question Answering (ORConvQA). To address the ambiguity of conversational questions, previous works in ORConvQA have explored various techniques as follows:

1. **Conversational Question Rewriting** (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Mele et al. 2021, Ren et al. 2018, Vakulenko, Longpre, Tu & Anantha 2021, Vakulenko, Voskarides, Tu & Longpre 2021, Voskarides et al. 2020, Yu et al. 2020) approaches are employed in ORConvQA to enhance the accuracy of the retrieved information by refining the user's original question. These techniques usually involve modifying the original question or generating new queries that better represent the user's intent by transforming a concise conversational question into a fully-grown, de-contextualised ad-hoc query. (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Mele et al. 2021, Ren et al. 2018, Vakulenko, Longpre, Tu & Anantha 2021, Vakulenko, Voskarides, Tu & Longpre 2021, Voskarides et al. 2020, Yu et al. 2020). For instance, existing work (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*) has fine-tuned a Text-to-Text Transfer Transformer (T5) to automatically reformulate the question by injecting information that exists in the conversation's context into a fully defined query. Figure 1.1 illustrates how the user's original utterance $U2$, "Hmm, I'm not sure. I'm interested in exploring the city and getting a feel for the local culture," can be reformulated into $U'2$ by resolving the term "city" to refer specifically to "Glasgow" (from $U1$).

2. **Follow-up Question Identification** (Bertomeu et al. 2006, Kirschner & Bernardi 2007, 2009, Kundu et al. 2020) is another approach used in ORConvQA to better understand the user's intent and context. The idea is to identify whether a follow-up question is related to the previous conversation history with the user or not. If the follow-up question is related to the previous conversation, the system can use the conversation's context to generate more accurate responses. As shown in Figure 1.1, the user's utterance $U2$ response to the system response $S1$ is a follow-up question related to the previous conversation using the user's utterance U_1, as they are seeking recommendations for things to do in Glasgow. Therefore, the system can use the context of the previous conversation to generate relevant responses

for $U2$. However, when the user's utterance $U3$ asks about historical landmarks, it is not directly related to $U2$'s previous response $S2$ about exploring the local culture. In this case, follow-up question identification can help the system to identify that $U3$'s question is not related to the previous conversation and that a new context has been established. The system can then use only $U1$ as a context to better understand $U3$'s intent and provide more accurate recommendations based on their specific request for historical landmarks. By doing this, previous works (Bertomeu et al. 2006, Kirschner & Bernardi 2007, 2009, Kundu et al. 2020) proposed to identify whether the follow-up question is related to the conversation history with the user.

3. **Asking Clarifying Questions** (Aliannejadi et al. 2021, 2019) is an approach used in ORConvQA to gather more information about the user's intent and context, particularly when the user's question is ambiguous or unclear. When the system is unsure about the user's question or needs more information to provide a relevant response, it can ask clarifying questions to better understand the user's intent and preferences. As exemplified in Figure 1.1, the user's utterance $U1$ is ambiguous and does not provide clear information on their interests. The system uses a clarifying question by asking $S1$ to gather more information. The user then clarifies their interests ($U2$) in exploring the city and getting a feel for the local culture. By doing so, the system can provide a more accurate answer that aligns with $U1$'s interests and preferences. Existing approaches for asking clarification questions consist of selecting clarification questions from a pool of pre-determined questions (Aliannejadi et al. 2021, 2019, Mass et al. 2021, Ou & Lin 2020) or generating clarification questions using rules or using text-generation models (Zamani et al. 2020). In particular, as introduced by Aliannejadi et al. (2021), the task of asking clarifying questions in conversational search can be broken down into two subtasks: determining when to ask clarifying questions (*Clarification Need Classification*) and how to generate them (*Asking Clarifying Questions*).

These techniques can be used together to improve the accuracy and relevance of the system's responses to conversational questions. For example, in Figure 1.1, the system could use follow-up question identification techniques to identify that $U3$'s question is related to historical landmarks and not to explore the local culture as in $U2$. Therefore, using only $U1$ as context, the system could then employ query reformulation techniques to better understand $U3$'s interests in historical landmarks and to provide more relevant responses. By reformulating $U3$'s question, the system can refine its understanding of the user's intent and tailor its responses to offer more accurate answers. We argue that by integrating the Follow-up Question Identification and Conversational Question Rewriting tasks, we aim to develop a more effective ORConvQA system. This integrated approach ensures the system not only understands but also adapts to the user's conversation history, effectively resolving ambiguities and enhancing response quality. In addition, consider a user who asks in $U1$, "I'm thinking of visiting Glasgow, what should I do while I'm there?"

without specifying types of interests. The Clarification Need Classification could detect the vagueness of the request. The system would then activate the Asking Clarifying Questions mechanism to ask in $S1$, "Can you give me an idea of your interests, such as art, history, or outdoor activities?" This interactive method refines the user's query and enhances the relevancy and precision of the system's responses. By integrating the Clarification Need Classification and Asking Clarifying Questions tasks, we aim to develop a conversational system that not only addresses user ambiguities but also enhances the interaction quality, making it more dynamic and user-focused. This approach is important for developing ORConvQA systems that can handle real-world conversational complexities effectively.

Another challenge in ORConvQA is to retrieve relevant passages from a large collection of documents, identify the most relevant ones based on the conversation context, and extract answers from the relevant passages. However, this process is further complicated by the need to retrieve these relevant passages from a large collection of documents – for instance, the OR-QuAC (Qu et al. 2020) dataset used in recent studies contains over 11 million passages. Given the complexity and variability of natural language, it is difficult to identify relevant information using traditional keyword-based search algorithms, making it a difficult computational challenge to accurately identify the most relevant passages for a given question. As exemplified in Figure 1.1, to respond to the user's utterance $U3$ "Hmm, I'm not sure. I'm interested in exploring the city and getting a feel for the local culture.", the system needs to identify and extract relevant information from a different set of passages that are focused on historical landmarks in Glasgow, and provide the user with accurate and informative answers. By accurately extracting information from the retrieved passages, the system can provide a personalised and satisfying experience for the user. This highlights the importance of developing effective methods for extracting relevant information in ORConvQA, which is an open challenge in the field. In particular, a more effective retriever improves the initial set of passages that the reranker processes and a precise reranker enhances the quality of passages from which the reader extracts the answer. We argue that the combination of these tasks in a unified workflow allows for a systematic approach to addressing ORConvQA challenges. By improving each component —retrieval, reranking, and reading— we aim to develop more effective systems that not only understand but also precisely respond to user's questions in conversational settings.

To address this challenge, previous works (Liang et al. 2022, Qu et al. 2021, 2020) have adopted a three-stage architecture, including a *retriever*, a *reranker*, and a *reader* to extract the answers. First, the *retriever* retrieves the top K relevant passages from the collection based on a question and the conversation history. The *reranker* and the *reader* then respectively rerank and identify an answer in the top K passages. For the *retriever*, existing works (Liang et al. 2022, Qu et al. 2021, 2020, Xiong et al. 2020, Yu et al. 2021) have focused on using a bi-encoder dense retrieval (a question encoder and a passage encoder e.g., convDR (Yu et al. 2021)), which applies neural contextual language models, such as ALBERT (Lan et al. 2020) or BERT (Devlin

et al. 2019), for encoding the question and passage into low-dimensional vectors and computing their relevance scores. For example, Yu et al. (2021) proposed ConvDR, which encodes the question and its history in a dense vector learned with a teacher-student model to mimic a dense representation of the manually rewritten question. ConvDR has also been shown to outperform other retriever models for conversational search such as sparse BM25, and bi-encoders using ALBERT (Qu et al. 2020) or BERT (Karpukhin et al. 2020, Xiong et al. 2020). However, despite the good effectiveness of bi-encoder dense retrievers for passage retrieval, there is still room for improvement in the ORConvQA task. Moreover, to extract the answers, Choi et al. (2018) introduce a main task, namely Answer Span prediction, which consists in answering a question by extracting text spans from a given passage. In addition to this main task, they also introduce several auxiliary tasks:

- **Yes/No prediction:** Determines whether the answer to a question is simply "yes" or "no."
- **Follow-up prediction:** Classifies whether the current question is a follow-up question.
- **Unanswerable prediction:** Recognises when a question cannot be answered based on the information available in the given passage.

Indeed, we argue that by integrating these tasks, we aim to enhance each task's effectiveness and contribute to overall system performance. By leveraging the interconnections of these tasks, our ConvQA system is designed to provide more accurate, relevant, and contextually appropriate responses.

Therefore, in this thesis, we aim to adapt and extend the bi-encoder dense retriever for ORConvQA by incorporating multi-task learning techniques. In particular, we propose a novel multi-task learning framework that not only trains the answer extraction but also incorporates auxiliary tasks such as passage retrieval and conversational question rewriting. By jointly optimising these tasks, our framework aims to improve the retriever's ability to identify relevant passages for a given question in a conversational context, while simultaneously predicting the answer for the question using the same model.

Indeed, Multi-Task Learning (MTL) has emerged as a promising solution to facilitate the learning of multiple related tasks by sharing the learner structure in a single model. MTL has gained considerable attention in recent years due to its effectiveness in addressing a diverse range of complex problems within a unified model (Ide & Kawahara 2021, Qu et al. 2021, 2020, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Ruder 2017). This MTL approach enables the model to leverage the shared knowledge and relationships among the tasks, leading to improved performance and enhanced generalisation capabilities. By adopting MTL, we can effectively address the challenges associated with learning multiple interconnected tasks, paving the way for more robust and comprehensive models (Ruder 2017). The MTL methods can be categorised into static or dynamic approaches. In the static MTL methods, the weights assigned to each task remain unchanged throughout the training phase, which may divert training resources to

unnecessary tasks. In contrast, in the dynamic MTL methods, the weights of each task are adjusted automatically to balance the loss rate or to balance the weights across tasks. This weight adjustment ensures that each of the tasks' weights is adjusted according to their relative importance, which can lead to better performance on all tasks.

1.3 Thesis Statement

The statement of this thesis is that the effectiveness of an ORConvQA system can be improved by leveraging a Multi-Task Learning method that jointly learns numerous different but related tasks at the same time in a uniform model. In particular, in the dynamic MTL approaches, the tasks' weights are automatically adjusted during learning, ensuring that each of the tasks' weights is adjusted by the relative importance of the different tasks. To accomplish our research objectives, we identify five areas of investigation in our thesis:

1. We propose a dynamic Multi-Task Learning approach that simultaneously trains the main task of answer extraction, along with auxiliary tasks such as follow-up question identification, yes/no question prediction, and unanswerable prediction. By incorporating these tasks into a unified model, we aim to enhance the system's effectiveness for Conversational Question Answering.

2. We explore the use of Multi-Task Learning to improve the performance of follow-up question identification and conversational question rewriting. By leveraging the shared learned structure of these tasks in a text generation model, we aim to enhance the system's effectiveness in follow-up question identification, conversational question rewriting and passage retrieval.

3. We investigate how Multi-Task Learning can be used to generate more effective clarifying questions by jointly learning clarification need classification and clarifying question generation. The proposed method aims to enhance the quality and relevance of the clarifying questions, thereby improving their effectiveness in assisting users and providing more precise and relevant responses.

4. We also propose to leverage Multi-Task Learning to enhance the performance of the Open-Retrieval Conversational Question Answering task by sharing the learned structure of the reranker and reader in a single text generation model. This approach aims to improve the system's effectiveness of retrieving relevant passages and extracting answers in ORConvQA.

5. We introduce a dynamic Multi-Task Learning approach that can jointly learn conversational question rewriting, passage retrieval, and answer extraction in a unified model. Our

hypothesis is that it's possible to enhance the overall effectiveness of Open-Retrieval Conversational Question Answering.

Through our research in these five areas, we aim to advance the field of Conversational Question Answering and Open-Retrieval Conversational Question Answering by developing novel Multi-Task Learning methodologies that improve the system's effectiveness and provide more relevant responses to the user's questions.

1.4 Contributions

The contribution of this thesis is six-fold:

1. In Chapter 4, we introduced our proposed Open-Retrieval Conversational Question Answering (ORConvQA) framework, consisting of four different methods ($ORConvQA_{1-5}$), which aim to bridge the gaps in existing ORConvQA research and improve the performance of ORConvQA system. All four ORConvQA methods leverage MTL to jointly learn multiple related tasks. MTL helps to improve the overall performance of the systems, as each task can help to inform the other tasks.

2. In Chapter 5, we study the effectiveness and efficiency of dynamic MTL methods including Evolving Weighting, Uncertainty Weighting, and Loss-Balanced Task Weighting, compared to *static* MTL methods such as the uniform weighting of tasks. We also propose a novel hybrid dynamic method that combines Abridged Linear with Loss-Balanced Task Weighting (LBTW) for auxiliary tasks, allowing the automatic fine-tuning of task weighting during learning. This approach ensures that the task weights are adjusted based on their relative importance, improving overall performance. Specifically:

    - We leverage dynamic Multi-Task Learning with BERT to effectively address the task of learning Answer Span extraction with its auxiliary tasks including Yes/No prediction, Follow-up Question Identification, and Unanswerable prediction.
    - To further enhance the performance of Multi-Task Learning, we introduce a hybrid strategy, which automatically fine-tunes the multiple tasks' weights along the learning steps. Our method uses Abridged Linear for the primary task and Loss-Balanced Task Weighting for the auxiliary tasks.
    - The proposed hybrid method yields the best performance improvements over the baselines on the QuAC dataset.

3. In Chapter 6, we explore the potential of Multi-Task Learning (MTL) to enhance the performance of the conversational question rewriting and classification tasks by leveraging their shared learned structure. In particular, we propose a novel approach to employing text

generation models (BART and T5). Our models are trained using MTL to simultaneously rewrite conversational questions and identify follow-up questions. This approach has the potential to improve the overall performance of both tasks and achieves better results than traditional approaches that treat them as separate tasks. Specifically:

- We leverage Multi-Task Learning with a text generation model to effectively address the tasks of follow-up question identification and conversational question rewriting.
- Using the recent LIF dataset (Kundu et al. 2020), we compare our models to two recent baselines from the literature, and show that our Multi-Task Learning BART model yields the best F1 and Macro-F1 performance improvements over the strongest baseline, namely the three-way attentive pooling (Kundu et al. 2020).
- The proposed Multi-Task Learning T5 model significantly outperforms the single-task learning of question rewriting models for passage retrieval on the OR-QuAC test set.

4. In Chapter 7, we leverage Multi-Task Learning by combining the tasks of clarification need classification and generation for asking clarifying questions. In particular, we propose a novel hybrid method that combines both generation and selection processes to generate clarifying questions in a conversational setting. By integrating both the generation and selection approaches, we can produce a better set of questions and ensure that the selected question is relevant to the user's query. Specifically:

   - We leverage Multi-Task Learning (MTL) with a single text generation T5 model that jointly learns both a classifier for clarification need and the generation of clarifying questions to effectively generate the clarifying questions.
   - We introduce a hybrid method for generating and selecting clarifying questions. Our method generates clarifying questions using generative models and selects clarifying questions from a pool of pre-determined questions to effectively address the task of asking clarifying questions.
   - We evaluate the performance of our method on a recent dataset of mixed-initiative conversational search and show the effectiveness of our proposed method, which significantly outperforms existing strong baselines.

5. In Chapter 8, we explore the use of Multi-Task Learning (MTL) to enhance performance on the ORConvQA task by leveraging the learned structure of the reranker and reader in a single text generation model. In particular, we introduce monoQA, a novel approach that employs a single text generation model with MTL for both passage reranking and answer extraction. Our model is based on the T5 text generation model and is fine-tuned simultaneously for both reranking (to improve the precision of the top retrieved passages)

and answer extraction. By integrating the reranker and reader tasks in a single model, we aim to improve the overall performance of the ORConvQA task. Specifically:

- We leverage Multi-Task Learning with a text generation model by sharing the reranker and reader's learned structure to effectively address the ORConvQA task.
- Using two different ORConvQA datasets, we compare our model to two strong baselines from the literature, and show that our MTL reranker and generative reader approach yields the best F1, Recall, MRR, and MAP performance improvements over a strong existing baseline from the literature, e.g. the ORConvQA system proposed by Qu et al. (2020).
- The proposed MTL model combining the reranker and generative reader is significantly more effective and is twice as fast for inference than the individual application of the monoT5 and UnifiedQA models for reranking and extracting the answers.

In addition, we further investigate how to leverage the Multi-Task Learning (MTL) of the three tasks: Conversational Question Rewriting, Passage Retrieval, and Answer Extraction. In particular, we introduce a novel method that leverages MTL to combine these tasks into a uniform model. Our MTL method is based on a bi-encoder BERT model, which is fine-tuned simultaneously for Conversational Question Rewriting (to better understand the users' questions within ongoing conversations), retrieval (to effectively retrieve the relevant passages from a large passages/documents collection), and answer extraction (to extract answers from the retrieved passages/documents accurately). Specifically:

- We leverage dynamic Multi-Task Learning with bi-encoder BERT to effectively address the task of learning.
- We evaluate the performance of our method on an ORConvQA OR-QuAC dataset.
- The proposed MTL model, which combines conversational question rewriting, retriever and reader, achieves better results than the baseline on existing two-stage pipeline baselines, such as a baseline that uses ConvDR as a retriever and UnifiedQA as a reader.

Our work uses Multi-Task Learning (MTL) to improve Open-Retrieval Conversational Question Answering (ORConvQA). We developed the ORConvQA framework (Chapter 4), which generates several models that address various stages of ORConvQA. We explored and enhanced these aspects of ORConvQA, including:

- **Chapter 5**: The effectiveness and efficiency of dynamic MTL methods on Conversational Question Answering (ConvQA).
- **Chapter 6**: Leveraging MTL for conversational question rewriting and follow-up question identification by sharing a single text generation model like BART and T5.

- **Chapter 7**: Leveraging MTL for combining clarification need classification and generation of asking clarifying questions by sharing a single text generation T5 model.
- **Chapter 8**: Leveraging MTL for integrating the reranking and answer extraction tasks using a single text generation T5 model and Leveraging MTL for integrating the conversation question rewriting, retrieval, and answer extraction tasks using a bi-encoder BERT model.

1.5 Origins of Material

Most of the material presented in this thesis is based on papers that were published in various international conferences during the author's PhD program.

- Chapter 5: In this chapter, we propose a novel hybrid dynamic method that combines Abridged Linear with Loss-Balanced Task Weighting (LBTW) for auxiliary tasks. By integrating these two methods, our approach enables the automatic fine-tuning of task weighting during learning. This allows for the adjustment of task weights based on their relative importance, which improves overall performance. Indeed, our method ensures that the importance of each task is reflected in its weight, resulting in a more accurate allocation of resources during learning. This work appeared first at the SCAI 2020 workshop (co-located with EMNLP 2020) (Kongyoung et al. 2020).
- Chapter 6: In this chapter, we propose a novel approach for employing text generation models (BART and T5) to improve conversational question rewriting and identification of follow-up questions. Specifically, we use Multi-Task Learning (MTL) to simultaneously train our models to rewrite conversational questions and identify follow-up questions. This work has been first published as a full paper at EMNLP 2023 (Kongyoung et al. 2023).
- Chapter 7: In this chapter, we propose a novel hybrid method that combines both generation and selection processes for generating clarifying questions in a conversational setting. Our approach leverages Multi-Task Learning (MTL) by combining the tasks of clarification need classification and question generation to simultaneously identify situations where the user's initial query requires a clarifying question, and generate a set of clarifying questions based on the query and conversation history. This approach allows our method to generate a diverse set of questions that are relevant to the user's queries, improving the effectiveness of the conversational system. This work is in preparation for submission to ACM TOIS.
- Chapter 8: In this chapter, we propose monoQA, a novel approach that employs a text generation model with MTL for both passage reranking and answer extraction. This work has first appeared in the EMNLP 2022 conference (Kongyoung et al. 2022).

1.6 Thesis Outline

The remainder of this thesis is structured as follows:

- Chapter 2 describes the background of developing Open-Retrieval Conversational Question Answering (ORConvQA). We first explore the types of passage retrieval and discuss the classical Information Retrieval (IR) evaluation metrics. Next, we explore Pre-trained Language Models (PLMs), including Encoder-Decoder, Encoder Only, and Decoder Only models. Finally, we present a typical taxonomy for Multi-Task Learning (MTL) approaches, which consists of the Learning Taxonomy and MTL Taxonomy.

- Chapter 3 discusses the related work in the literature addressing the ORConvQA task. We begin by describing the typical approaches used in ORConvQA, and then delve into the Multi-Task Learning techniques previously employed in Conversational Question Answering. In addition, we discuss the typical datasets used in ORConvQA and the evaluation metrics used to assess performance in the ORConvQA task.

- Chapter 4 presents our proposed framework, highlighting the motivation behind each component and the specific tasks we aim to address. We also provide formal definitions for the tasks within the framework, complete with the used terminology and equations.

- Chapter 5 explores the use of dynamic Multi-Task Learning (MTL) methods to improve the effectiveness and efficiency of conversational question answering. We also investigate the use of dynamic MTL methods, including Evolving Weighting, Uncertainty Weighting, and Loss-Balanced Task Weighting, compared to static MTL methods such as the uniform weighting of tasks. Next, we introduce our proposed novel hybrid dynamic method that combines Abridged Linear with Loss-Balanced Task Weighting (LBTW), allowing the automatic fine-tuning of task weighting during learning. We evaluate our proposed method on the QuAC dataset and discuss its performance in comparison to the static MTL baselines.

- Chapter 6 explores using Multi-Task Learning (MTL) to improve the conversational question rewriting and follow-up question identification tasks by employing text generation models (BART and T5) and leveraging their shared learned structure. Our approach shows the potential to outperform traditional methods that treat these tasks separately. We demonstrate the effectiveness of our Multi-Task Learning BART model on the recent LIF dataset, achieving significant improvements over recent baselines.

- Chapter 7 introduces a novel approach for generating and selecting clarifying questions in a conversational setting. Leveraging Multi-Task Learning, we combine the tasks of clarification need classification and question generation to simultaneously identify situations in which the user's initial query requires a clarifying question and generate a set of relevant questions. Our hybrid method uses both generative and selection models and is

evaluated on a recent dataset of mixed-initiative conversational search, showing significant improvements over existing baselines.

- Chapter 8 introduces monoQA, a text generation model that leverages Multi-Task Learning (MTL) to simultaneously perform reranking and answer extraction tasks for the ORConvQA task. By sharing the learned structure of the reranker and reader, our method significantly outperforms strong baselines and achieves better performance on F1, Recall, MRR, and MAP metrics. The proposed MTL model is also twice as fast for inference compared to the individual application of the monoT5 and UnifiedQA models for reranking and answer extraction. Moreover, this chapter also explores the use of dynamic Multi-Task Learning (MTL) methods to improve the effectiveness of the ORConvQA task by combining the three tasks: conversation rewriting, passage retrieval, and answer extraction. We evaluate our proposed method on the OR-QuAC dataset and discuss its performance in comparison to the two-stage pipeline baselines.
- Chapter 9 marks the conclusion of this work, summarising our findings and outlining potential avenues for future research.

Chapter 2

Background

2.1 Introduction

Open-Retrieval Conversational Question Answering (ORConvQA) is a challenging task. The fundamental objective of ORConvQA systems is two-fold: firstly, to retrieve relevant passages from a large collection of texts or documents, and secondly, to generate/extract contextually relevant responses to user queries. By achieving these goals, the ORConvQA systems aim to achieve accurate and natural language responses, effectively bridging the gap between human-like conversational interactions and machine-generated outputs. To accomplish this, machine learning techniques are extensively employed to enhance the performance of ORConvQA systems.

This chapter provides some necessary background for this thesis. First, it focuses on the taxonomy of retrieval approaches, including both sparse and dense methods. The advantages and limitations of each technique are discussed, along with their relevance to the ORConvQA systems. The chapter also discusses Multi-Task Learning (MTL). The capacity of MTL to simultaneously learn numerous tasks aligns directly with the two-fold objective of the ORConvQA systems: retrieving relevant passages and generating/extracting contextually relevant responses to the user's queries. As a result, it is important to understand MTL's benefits and challenges, as it forms an essential component of the strategies deployed in this thesis.

In this chapter, we start with Section 2.2, which presents Sparse Retrieval. Section 2.3 then describes Pre-trained Language Models (PLMs). Section 2.4 focuses on Dense Retrieval & Reranking, followed by an overview of PyTerrier in Section 2.5. Section 2.6 explores Hybrid Sparse and Dense Retrieval techniques. The chapter then reviews Classical IR Evaluation Metrics in Section 2.7 and introduces Multi-Task Learning in Section 2.8. Finally, Section 2.9 gives the concluding remarks for this chapter.

2.2 Sparse Retrieval

Sparse passage or document retrieval approaches involve representing queries and documents as sparse vectors in a high-dimensional space, typically based on the unique words they contain. In this context, each unique word maps to a dimension, making the representation incredibly sparse due to the large number of possible words. These approaches have been widely used in information retrieval tasks, including for passage/document retrieval in ORConvQA systems. Some commonly employed methods in sparse retrieval, such as BM25 (Robertson & Walker 1994), PL2 (Amati 2003), and DPH (Amati et al. 2008). In particular, in this thesis, we will focus on BM25, which has been widely employed in information retrieval tasks, including passage/document retrieval in ORConvQA systems.

BM25 (Best Match 25), was introduced by Robertson & Walker (1994) and is a widely used weighting scheme in Information Retrieval (IR). It is an extension of the TF-IDF (Salton 1971) weighting scheme and is designed to overcome some of the limitations of TF-IDF (Liu et al. 2009). Like TF-IDF, BM25 is a statistical method used to evaluate the relevance of a term to a document in a collection. The BM25 weighting scheme takes into account the frequency of the term in the document, the frequency of the term in the entire collection, and the document length. BM25 uses a probabilistic IR framework to estimate the relevance of a document to a query. It does not directly calculate the probability of a document being relevant to a query but instead generates a relevance score based on the query terms and the document content. This score is derived from three components: a term frequency component, an inverse document frequency component, and a document length normalisation component. The term frequency component is similar to the TF component in TF-IDF, but it is modified to account for the saturation of the score at high term frequencies. The inverse document frequency component is similar to the IDF component in TF-IDF, but it is modified to account for the distribution of term frequencies across the corpus. The document length normalisation component normalises the scores by the length of the document and the average length of all documents in the corpus. The BM25 score for a document d given a query Q, as presented in (Zangerle et al. 2013), is calculated as follows:

$$BM25(Q,d) = \sum_{t \in Q} IDF(t) \cdot \frac{f(t,d) \cdot (k_1 + 1)}{f(t,d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \tag{2.1}$$

where $f(t,d)$ is the frequency of term t in document d, IDF(t) is the inverse document frequency of term t, k_1 and b are free parameters that control the scaling of the term frequency and document length normalisation components, respectively, and $avgdl$ is the average length of documents in the collection.

In Chapter 6, we employ BM25 as a primary approach for passage retrieval, serving as a first-pass method to retrieve relevant passages. In addition, in Chapter 7, BM25 is used as a baseline for clarifying question selection, providing a simple benchmark against which other selection approaches are evaluated.

Limitations of Sparse Retrieval

One limitation of sparse retrieval techniques is their reliance on the occurrence of individual terms in queries and documents. They treat documents and queries as bags of words, disregarding the relationships and co-occurrences between terms. As a result, these techniques may struggle to capture the semantic meaning and context of queries accurately. Sparse models can fail to distinguish between different senses of a word or to consider the broader context in which a query term is used. Moreover, sparse retrieval methods are less adept at handling complex conversational queries that involve multiple intents or sub-questions. They do not easily incorporate historical context and previous interactions, which are essential for effective ORConvQA systems. This limitation is highlighted by the effectiveness of ConvDR (Yu et al. 2021), a Conversational Dense Retrieval system, which learns contextualised embeddings for multi-turn conversational queries. ConvDR outperforms the sparse retrieval approaches for handling complex conversational queries. The advantage of ConvDR is its ability to capture informative context while ignoring the unrelated context in the previous conversation turns. Sparse retrieval models often treat each query independently, disregarding the conversational flow and the need to consider previous interactions to provide accurate and relevant responses. While sparse passage retrieval techniques have been extensively used and have been shown to be effective in various information retrieval tasks (Dai & Callan 2020, Fraser et al. 2018, Mallia et al. 2021, Yang et al. 2021, Zamani et al. 2018), the limitations outlined above motivate the integration of alternative approaches that can capture semantic meaning, contextual understanding, and handle complex conversational queries more effectively. This has led to the development and adoption of dense retrieval techniques, which leverage dense vector representations and neural networks to overcome the shortcomings of sparse models, often based on Pre-trained Language Models.

2.3 Pre-trained Language Models (PLMs)

Pre-trained Language Models (PLMs), such as BERT (Devlin et al. 2019), have transformed the field of natural language processing by learning rich contextual representations of text. PLMs are deep learning models that are trained on large amounts of textual data, typically by employing unsupervised learning techniques. They have significantly advanced the state-of-the-art performance in various language-related tasks, including text classification (Lan et al. 2020, Zhuang et al. 2021), named entity recognition (Darji et al. 2023, Taher et al. 2019), sentiment analysis (Devlin et al. 2019, Raffel et al. 2020, Zhuang et al. 2021), machine translation (Lewis et al. 2019, Raffel et al. 2020), and conversational question answering (Khashabi et al. 2020, Qu

et al. 2020, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Yu et al. 2021). A transformer model is an architectural evolution that has contributed to the success of PLMs. The transformer model is a neural network that can capture long-range dependencies between words in a sentence, which makes it effective at understanding the context of words for natural language processing tasks (Vaswani et al. 2017).

Transformers (Vaswani et al. 2017), indeed, have transformed the field of Natural Language Processing (NLP). Transformer-based models have shown impressive results in diverse tasks, including language translation (Lewis et al. 2019, Raffel et al. 2020), text classification (Devlin et al. 2019, Lewis et al. 2019, Raffel et al. 2020), and text generation (Brown et al. 2020, Radford et al. 2019). The strength of Transformers lies in their flexibility, as they can be adapted and fine-tuned for specific tasks, making them highly versatile and effective in different NLP applications. The Transformer model consists of an encoder and a decoder, each composed of multiple layers of self-attention and feed-forward layers, as shown in Figure 2.1. The encoder processes the input sequence and generates a representation that captures its meaning and structure. The decoder generates the output sequence by attending to the encoder representation and its own previous outputs. The attention mechanism allows the model to focus on the relevant parts of the input and output sequences, and to learn long-range dependencies (Vaswani et al. 2017). Multi-head attention is a variant of the attention mechanism used in the Transformer model. It allows the model to learn multiple representations of the input and output sequences by computing attention multiple times with different weight matrices. This enables the model to capture different aspects of the relationships between the input and output sequences. Multi-head attention ($MultiHead$) is defined as follows:

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,...,\text{head}_h)W^O \tag{2.2}$$

where

$$head_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.3}$$

and

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.4}$$

Here, the Q, K, and V matrices are named the *queries*, *keys*, and *values*, respectively. W_i^Q, W_i^K, W_i^V, and W^O are learnable weight matrices, h is the number of heads, and d_k is the dimension of each head. In addition, *queries*, *keys*, and *values* are the inputs to the attention mechanism. *Queries* are used to represent the output positions that need to be generated. *Keys* are used to represent the input positions that are relevant for generating the output. *Values* are used to compute the weighted sum of the input positions based on their relevance. The attention mechanism computes the similarity between the *queries* and *keys*, and uses it to assign weights to *values*.

In this section, we will delve into the transformer architectures, exploring their varied types and the wide range of applications they can be employed in. In addition, we will provide examples of popular models that leverage these different architectures to showcase their effectiveness in real-world scenarios. Moreover, we will discuss where within our research these transformer architectures are specifically applied.

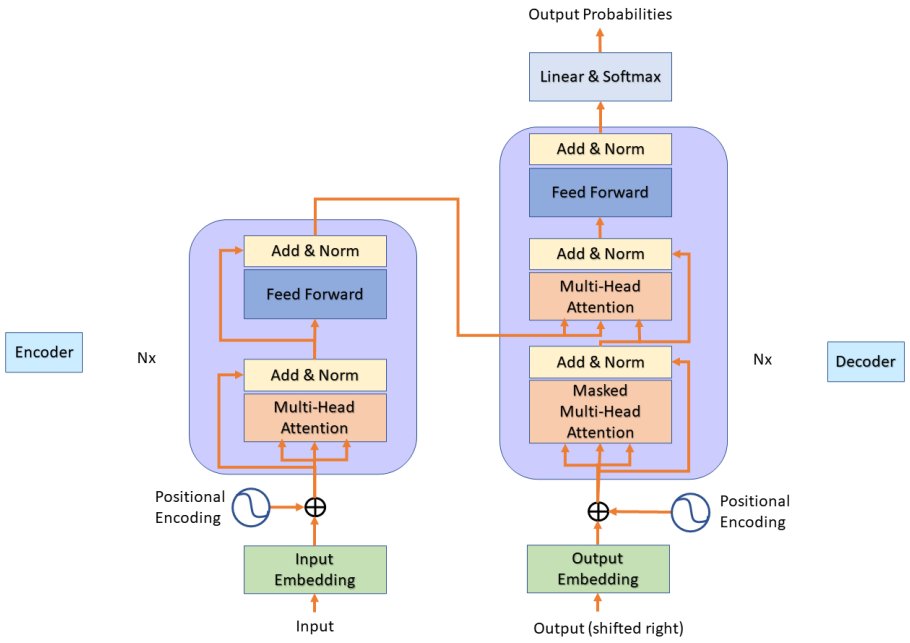


Figure 2.1: The original transformer architecture introduced in Vaswani et al. (2017) is based on the Encoder-Decoder architecture.

2.3.1 Encoder-Decoder

The original transformer architecture introduced in Vaswani et al. (2017) is based on the Encoder-Decoder structure, as illustrated in Figure 2.1, where the encoder is represented on the left of the figure and the decoder is on the right. The encoder takes the input sequence and produces a condensed hidden representation that captures the essential information from the input sequence of tokens. On the other hand, the decoder uses this hidden representation to generate the desired output sequence. During training, the encoder and decoder are jointly optimised to maximise the likelihood of producing the correct output sequence given the input sequence of tokens. This end-to-end training approach ensures the seamless integration of both components for effective sequence processing. Encoder-Decoder Pre-trained Language Models (PLMs), exemplified by models like T5 (Raffel et al. 2020) (Text-to-Text Transfer Transformer) and BART (Lewis et al. 2019) (Bidirectional and Auto-Regressive Transformers), have made significant advances in natural language processing tasks including machine translation, text summarisation, and question answering (Jiang et al. 2022*b*, Khashabi et al. 2020, Lewis et al. 2019, Raffel et al. 2020).

T5 (Text-to-Text Transfer Transformer) (Raffel et al. 2020) is a state-of-the-art language model that uses a unified text-to-text transformer architecture for various natural language processing tasks. It is a pre-trained model that can be fine-tuned on a wide range of tasks, including text classification, question answering, summarisation, and translation. It has also been shown to be usable for other tasks, such as classification (Raffel et al. 2020), document re-ranking (Nogueira, Jiang & Lin 2020) (where it has been shown to outperform BERT-based models) and even arithmetic tasks (Nogueira et al. 2021). These tasks are all handled by taking a sequence of text, and then seeing what token it predicts and with what probability. The T5 model (Raffel et al. 2020) was trained on a large collection of text data, including web pages, books, and articles, and was capable of generating high-quality text outputs. The T5 model achieved state-of-the-art results on various language understanding tasks and had been shown to outperform other language models such as BERT (Devlin et al. 2019) and GPT-2 (Radford et al. 2019). The T5 model is available in different sizes, ranging from T5-Small to T5-11B [1], with larger models having more parameters and higher computational requirements.

In summary, T5 is a text-to-text framework, which is built upon a transformer-based architecture, and it has been shown to be effective on various tasks. In Chapters 6, 7, and 8, we deploy T5 as a question rewriting model. In addition, in Chapter 7, we employ T5 for clarifying need classification and generating clarifying questions. Furthermore, in Chapter 8, we use T5 for passage reranking and for extracting answers.

BART (Bidirectional Auto-Regressive Transformers) (Lewis et al. 2019) is a denoising autoencoder that uses a sequence-to-sequence model for pretraining. It was built on a standard Transformer-based neural machine translation architecture, which allows it to generalise other pretraining schemes like BERT (Devlin et al. 2019), and GPT-2 (Radford et al. 2019). BART has two stages of pretraining: first, text is corrupted with an arbitrary noising function, and then a sequence-to-sequence model is learned to reconstruct the original text. One of the advantages of BART is its flexibility in applying arbitrary transformations to the original text, including changing its length. BART has achieved state-of-the-art results on a range of tasks, including natural language generation, translation, and comprehension (Lewis et al. 2019). BART has been shown to match the performance of other models like RoBERTa (Zhuang et al. 2021) on tasks like GLUE (Wang et al. 2018) and SQuAD (Rajpurkar et al. 2018, 2016), and has achieved new state-of-the-art results on abstractive dialogue, question answering, and summarisation tasks.

In summary, BART is a combination of bidirectional and auto-regressive transformers, along with its denoising pre-training objective. Its ability to capture bidirectional context, generate coherent output, and perform well on tasks such as text generation, text summarisation, and document classification makes BART an effective and adaptable tool for natural language processing. In Chapter 6, we employ BART as a conversational question rewriting and follow-up question identification model.

[1] The T5 model can be found at https://github.com/google-research/text-to-text-transfer-transformer.

The use of Encoder-Decoder PLMs, including T5 and BART, highlights the importance of integrating both encoding and decoding components in addressing complex language tasks. By jointly training the encoder and decoder, these models can effectively capture contextual information and produce meaningful and relevant outputs.

2.3.2 Encoder-Only

In contrast to Encoder-Decoder PLMs such as T5 (Raffel et al. 2020) and BART (Lewis et al. 2019), the Encoder-only architecture is employed when encoding the input sequence is sufficient, without the need for a decoder. In this configuration, the input sequence undergoes encoding to generate a fixed-length representation, which is subsequently used as input for a classifier or regressor to make predictions. These models incorporate a pre-trained encoder that possesses general-purpose capabilities, necessitating the fine-tuning of the ultimate classifier or regressor. This adaptable output nature renders them valuable for a myriad of applications, including: Text classification, Sentiment analysis, and Named entity recognition. Prominent models that adopt this architecture include BERT (Devlin et al. 2019) and ALBERT (Lan et al. 2020).

BERT (Bidirectional Encoder Representations from Transformer) (Devlin et al. 2019) is an effective language representation model that is designed to pre-train deep bidirectional representations from an unlabeled text by jointly conditioning on both the left and right context in all layers. Unlike other language representation models, BERT has a unified architecture across different tasks, with a minimal difference between the pre-trained architecture and the final downstream architecture. BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation described in Devlin et al. (2019). BERT can be fine-tuned with just one additional output layer to create state-of-the-art models for various tasks, such as question answering and language inference, without substantial task-specific architecture modifications. BERT achieved state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE (Wang et al. 2018) score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 (Rajpurkar et al. 2016) question answering Test F1 to 93.2 (1.5 points absolute improvement) and SQuAD v2.0 (Rajpurkar et al. 2018) Test F1 to 83.1 (5.1 points absolute improvement). Moreover, BERT can also be used as the base model for dense passage/document retrieval systems such as ColBERT (Khattab & Zaharia 2020), DPR (Karpukhin et al. 2020), ANCE (DPR-based model) (Xiong et al. 2020), TCT-ColBERT (Lin et al. 2021*b*), and ConvDR (Yu et al. 2021) (see details in Section 3.7.4). It can also be used for conversational question answering models like HAM (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019) (details in Section 3.2.4).

In summary, BERT is an effective Encoder-only pre-trained language model (PLM) that uses deep bidirectional training from unlabeled text. Its unified architecture across tasks and its ability to fine-tune have made it adaptable, achieving state-of-the-art results in various tasks. In addition, BERT is used as a base model for systems in dense passage/document retrieval (Karpukhin

et al. 2020, Khattab & Zaharia 2020, Lin et al. 2021*b*, Xiong et al. 2020, Yu et al. 2021) and conversational question answering (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019), demonstrating its wide-ranging applicability in natural language processing and information retrieval. In Chapter 5, we employ BERT as an answer extraction model. Moreover, BERT serves as a baseline for predicting follow-up questions in Chapter 6.

ALBERT (A Lite BERT) (Lan et al. 2020) is a lite version of BERT, a popular pre-trained language model for natural language processing tasks. ALBERT uses two parameter-reduction techniques to lower memory consumption and increase training speed. The backbone of the ALBERT architecture is similar to BERT in that it uses a transformer encoder. However, ALBERT has significantly fewer parameters than a traditional BERT architecture. The self-supervised loss function focuses on modelling inter-sentence coherence, leading to new state-of-the-art results on various benchmarks. Both single-model and ensemble results indicate that ALBERT improves the state-of-the-art significantly on several benchmarks, achieving a GLUE (Wang et al. 2018) score of 89.4, a SQuAD 2.0 (Rajpurkar et al. 2018) test F1 score of 92.2, and a RACE test (Lai et al. 2017) accuracy of 89.4. Moreover, ALBERT was used as a base model for the Open-Retrieval Conversational Question Answering (ORConvQA) system, introduced by Qu et al. (2020). The experimental results on the OR-QuAC dataset demonstrated that the ALBERT-based ORConvQA system outperformed several state-of-the-art baselines, including DrQA (Chen et al. 2017), a baseline model that uses a document retriever and a reader based on a recurrent neural network (RNN) and BERTserini (Yang et al. 2019), a baseline model that uses a document retriever and a reader based on the BERT architecture.

In summary, ALBERT is an encoder-only Pre-trained Language Model that addresses the challenges of model size and training efficiency. It achieves parameter reduction by factorising the embedding layer and sharing parameters across layers within the encoder. With its transformer-based encoder architecture and competitive results on various benchmarks such as GLUE, SQuAD 2.0, and RACE, ALBERT provides an efficient and effective solution for various tasks. In Chapter 8, we employ the ALBERT-based ORConvQA system (Qu et al. 2020) as a baseline for ORConvQA.

2.3.3 Decoder-Only

Decoder-only Pre-trained Language Models are a type of pre-trained language model that are designed to decode input representations into coherent and contextually appropriate output sequences. They are typically used for natural language generation tasks, such as text summarisation and machine translation. They are also used for other tasks that require the ability to generate text, such as question answering and creative writing (Brown et al. 2020). Some examples of decoder-only language models include GPT-1 (Generative Pre-trained Transformer 1) (Radford et al. 2018), GPT-2 (Radford et al. 2019), and GPT-3 (Brown et al. 2020).

GPT-3 (Generative Pre-trained Transformer 3) (Brown et al. 2020) is a state-of-the-art Pre-trained Language Model developed by OpenAI. It is a neural network-based model that has been trained on large textual data using an unsupervised learning approach. GPT-3 is capable of generating human-like text, completing sentences, paragraphs, and even entire articles. It can also perform a wide range of natural language processing tasks, including machine translation, open-domain question answering, and closed-domain question answering tasks (Brown et al. 2020). GPT-3 has been shown to outperform the baselines on various tasks without the need for fine-tuning on a large dataset, using few-shot learning. In Chapter 7, we use GPT-3 as our baseline for generating clarifying questions, building upon the recent work of Owoicho et al. (2022) who proposed training the model with few-shot learning on TREC CAsT 2021 (Dalton et al. 2021). This approach involves using GPT-3 for generation-based questions, which has shown promising results in the context of asking clarifying questions.

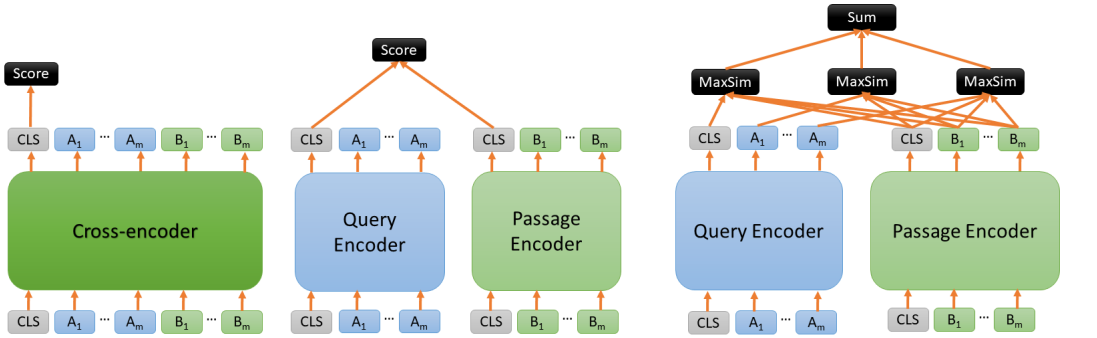
Table 2.1: Comparison of pre-trained transformer model parameter sizes (Casola et al. 2022).

Model	Year	Architecture	Parameters
BERT (Devlin et al. 2019)	2018	Encoder only	BERT-base: 110 M BERT-large: 345 M
RoBERTa (Zhuang et al. 2021)	2019	Encoder only	RoBERTa-base: 125 M RoBERTa-large: 355 M
ALBERT (Lan et al. 2020)	2019	Encoder only	ALBERT-base: 11 M ALBERT-large: 17 M ALBERT-xlarge: 58 M ALBERT-xxlarge: 223 M
GPT-1 (Radford et al. 2018)	2018	Decoder only	110 M
GPT-2 (Radford et al. 2019)	2019	Decoder only	GPT-2: 117 M GPT-2-medium: 345 M GPT-2-large: 774 M GPT-2-xl: 1558 M
GPT-3 (Brown et al. 2020)	2020	Decoder only	175 B
T5 (Raffel et al. 2020)	2019	Encoder-Decoder	T5-small: 60M T5-base: 220 M T5-large: 770 M T5-3B: 2.8 B T5-11B: 11 B
BART (Lewis et al. 2019)	2019	Encoder-Decoder	BART-base: 140 M BART-large: 406 M

In conclusion, based on recent advances in natural language processing (NLP) and Information Retrieval (IR), pre-trained language model like BERT, ALBERT, GPT-3, T5, and BART have demonstrated effectiveness in language understanding and generation. The parameter sizes of pre-trained language models such as BERT, RoBERTa, ALBERT, GPT-1, GPT-2, GPT-3, T5, and

BART are compared in Table 2.1, providing insights into the scale and complexity of these models and their unique architectures. This information is important in understanding the computational and resource requirements when employing these models in real-world applications such as in real-world applications, such as in natural language processing tasks, including machine translation (Lewis et al. 2019, Raffel et al. 2020), sentiment analysis (Lan et al. 2020, Zhuang et al. 2021), and automated content generation (Brown et al. 2020). In the next section, we leverage the capabilities of these pre-trained transformer models discussed in this section to explore Dense Retrieval and Reranking.

2.4 Dense Retrieval & Reranking



(a) Cross-encoder (e.g., BERT) (b) Bi-encoder single representative model (e.g., DPR) (c) Bi-encoder multiple representative model (e.g., ColBERT)

Figure 2.2: Comparison of dense retrieval model architectures (Zhang et al. 2022).

Recent progress in the field of information retrieval has led to the emergence of dense retrieval and reranker approaches that offer better performance than traditional sparse retrieval methods (Khattab & Zaharia 2020, Lin et al. 2021*b*, Ni et al. 2021, Nogueira, Jiang, Pradeep & Lin 2020*a*, Yu et al. 2021). These techniques use deep learning models, which include Pre-trained Language Models (PLMs) (see Section 2.3), to capture semantic and contextual relationships between queries and passages/documents, ultimately resulting in more accurate and context-aware retrieval and reranking outcomes.

In the following, we delve into the different model architectures that are designed for the retrieval and reranking tasks. Figure 2.2 provides a visual comparison of these architectures. As shown in Figure 2.2(a), the cross-encoder model architecture is used as a reranker. On the other hand, dense retrieval models employ bi-encoder model architectures, which are depicted in Figure 2.2(b) and Figure 2.2(c). We explore these architectures in detail in the following section.

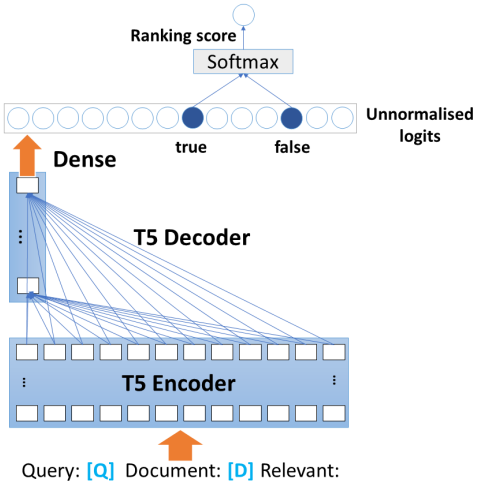


Figure 2.3: Structure of the monoT5 re-ranking model (Zhuang et al. 2022).

2.4.1 Cross-Encoder

The cross-encoder model is often used to rerank in retrieval tasks, by combining the query and document text as input for PLMs like BERT (Lan et al. 2020). It provides a score based on the embeddings of the [CLS] token. This model is effective for handling mismatched vocabulary and semantics between the query and document. However, since it requires high computational resources, it is less practical for large-scale retrieval and is typically used in the second stage of reranking (Nair et al. 2022).

The monoT5 reranking model, as introduced by Nogueira, Jiang, Pradeep & Lin (2020*a*), is a cross-encoder passage re-ranker has been shown to be effective in various information retrieval tasks. It is able to capture and represent the semantic and contextual relationships between queries and passages in a unified manner by employing the encoder-decoder T5 model, as explained in Section 2.3.1. In order to improve its ranking capabilities, Nogueira, Jiang, Pradeep & Lin (2020*a*) employs a methodology that involves concatenating query-document pairs into a single input sequence. The T5 model was fine-tuned using a text generation task to generate the tokens, namely "true" and "false," to indicate the relevance of each pair. Then, the model derives ranking scores from the logits. These logits represent the output values before applying a softmax function and correspond to the "true" and "false" tokens, as shown in Figure 2.3. The monoT5 reranking model (Nogueira, Jiang, Pradeep & Lin 2020*a*) has been shown to outperform a classification-based encoder-only approach such as BERT, especially in a data-poor setting with limited training data. In Chapters 6, 7 and 8, we employ the monoT5 model as the passage reranking approach.

2.4.2 Single Representation Bi-Encoder

Single Representations in Dense Passage Retrieval is a technique where entire passages are represented by a single embedding, usually BERT's [CLS] token (Macdonald et al. 2021), as

shown in Figure 2.2(b). During retrieval, the [CLS] embedding of the query is compared to the [CLS] embeddings of passages using vector similarity measures like cosine similarity or dot product. The passages with the highest similarity scores are ranked as the most relevant. This technique is used to compute the similarity between a query and a passage by identifying the nearest neighbours using a FAISS index (Johnson et al. 2021). The similarity function used in dense retrieval is often cosine or dot product (Xiong et al. 2020). The function calculates the retrieval score $f(\cdot)$ using similarities in a learned embedding space (Karpukhin et al. 2020, Lee et al. 2019, Luan et al. 2021*a*) defined as follows:

$$f(q,d) = sim(E(q,\theta),E(d,\theta)) \tag{2.5}$$

where $E(\cdot)$ is the representation model that encodes the query or document to dense embeddings. The encoder parameter θ provides the main capacity, often fine-tuned from pre-trained transformers, e.g., BERT. The similarity function $(sim(\cdot))$ leverages efficient ANN retrieval and is used to compare the dense embeddings of the query and document, as detailed as follows:

Cosine Similarity: Cosine similarity measures the cosine of the angle between two vectors. The higher the cosine similarity score, the more relevant the document is to the given query:

$$sim(\cdot) = \frac{E(q,\theta)\cdot E(d,\theta)}{\|E(q,\theta)\| \times \|E(d,\theta)\|} \tag{2.6}$$

Dot Product: In the case of normalised vectors (division by magnitude), the dot product is conceptually similar to cosine similarity and effectively measures how aligned the vectors are in the vector space. Then a dot product is performed to measure the similarity between q and d based on their embeddings:

$$sim(\cdot) = E(q,\theta)^T \cdot E(d,\theta) \tag{2.7}$$

Thakur et al. (2021) found that the choice of similarity function, either cosine similarity or dot product, can have a significant impact on the performance of dense models in information retrieval. The cosine similarity model was found to prefer shorter documents over longer ones, while the dot product model primarily retrieves longer documents. The study also found that the performance of the two models varied across different datasets, with the dot product model achieving the biggest improvement on TREC-COVID (Voorhees et al. 2021). Therefore, the choice of similarity function should be carefully considered depending on the nature and needs of the specific task. In this thesis, this insight guided our decision to select cosine similarity for our models, considering its suitability for the used datasets which have the passage size at most 384 tokens (Qu et al. 2020).

The advantage of using Single Representations is that it reduces the computational cost of indexing and retrieval, as only one embedding is required for each passage (Macdonald et al. 2021). However, the disadvantage is that the meaning of an entire passage is assumed to be represented by a single embedding (Macdonald et al. 2021), which may not always be accurate.

Several approaches have been proposed to enhance dense retrieval performance using the bi-encoder architecture, which consists of separate question and passage encoders. This architecture applies neural contextualised language models, such as T5 or BERT, to encode both questions and passages into low-dimensional vectors before computing their relevance scores Lin, Yang & Lin (2020), Ni et al. (2021), Xiong et al. (2020).

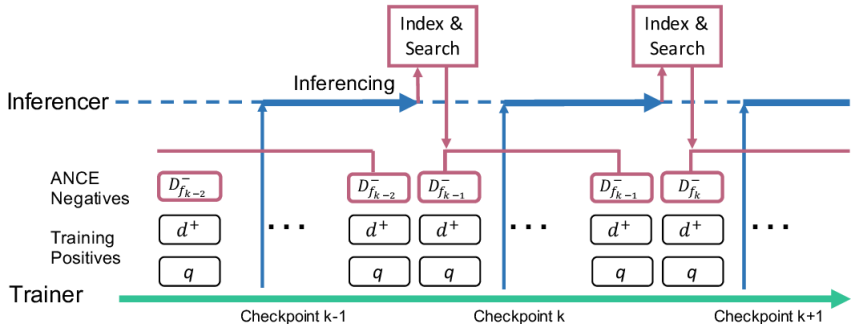


Figure 2.4: ANCE Asynchronous Training. Figure taken from (Xiong et al. 2020).

One approach is ANCE (Approximate nearest neighbour Negative Contrastive Estimation) (Xiong et al. 2020), which is a learning mechanism for dense retrieval that selects hard training negatives globally from the entire collection, using an asynchronously updated ANN (Approximate Nearest Neighbor) index. As shown in Figure 2.4, in the ANCE model, two main components are involved: the Trainer and the Inferencer. The Trainer is responsible for learning the representation by using negatives from the ANN index. It capitalises on the efficient storage and retrieval capabilities of the ANN index to sample relevant negative examples, thereby enhancing the model's ability to discriminate between similar documents. On the other hand, Inferencer plays an important role in updating the document representations in the collection. It uses a recent checkpoint of the trained model to refresh and refine the encodings of the documents. By incorporating the latest information and insights gained from the trained model, the Inference component ensures that the document representations remain accurate and up-to-date. Together, the Trainer and Inferencer work in tandem to continuously improve the performance and effectiveness of the ANCE model. The Trainer learns from negative examples provided by the ANN index, while the Inferencer keeps the document representations aligned with the latest knowledge obtained from the trained model. This dynamic and iterative process enables the ANCE model to adapt and evolve, delivering robust and accurate results in dense text retrieval tasks. Moreover, ANCE has been shown to outperform all sparse retrieval methods and even outperforms DPR (Karpukhin et al. 2020) in passage retrieval for OpenQA (Xiong et al. 2020).

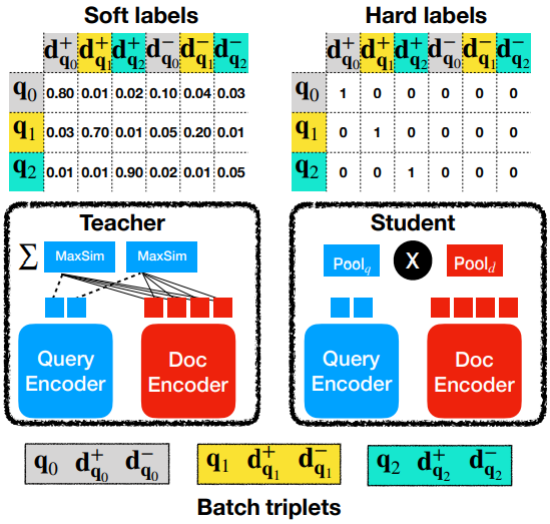


Figure 2.5: The process of distilling dense representations for ranking involves a close integration between the teacher and student models. Figure taken from (Lin et al. 2021*b*).

TCT-ColBERT (Tightly-Coupled Teacher ColBERT) (Lin, Yang & Lin 2020, Lin et al. 2021*b*) is a bi-encoder model that applies knowledge distillation to improve the ColBERT model (Khattab & Zaharia 2020) (see Section 2.4.3). TCT-ColBERT distils the knowledge from ColBERT's expressive MaxSim operator for computing relevance scores into a simple dot product, thus enabling a single-step ANN search. The insight is that during distillation, tight coupling between the teacher model and the student model enables more flexible distillation strategies and yields better learned representations, as presented in Figure 2.5. TCT-ColBERT has been shown to outperform its baselines including sparse retrieval methods, ANCE, ColBERT, and bi-encoder (PoolAvg), in both MRR@10 and NDCG@10 for MS MARCO and TREC2019 DL (Lin, Yang & Lin 2020).

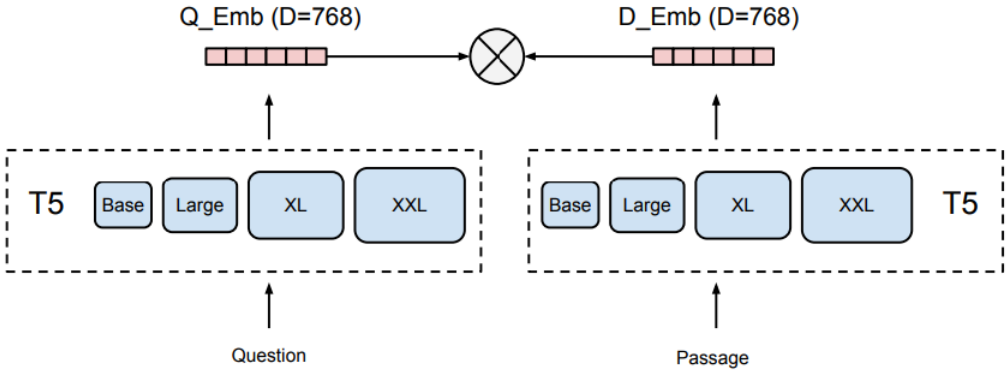


Figure 2.6: Architecture of Generalisable T5-based dense Retrievers. Figure taken from (Ni et al. 2021).

On the other hand, the GTR (Generalisable T5-based dense Retrievers) (Ni et al. 2021) model is a scaled-up dual encoder model with a fixed-size dot-product bottleneck layer. GTR is a neural

retrieval model that leverages pre-trained T5 (Raffel et al. 2020) models of different sizes as the backbone encoder and performs multi-stage training on web-mined question-answer pairs and search datasets, as shown in Figure 2.6. GTR has been shown to be effective on a variety of retrieval tasks, including the BEIR benchmark (Thakur et al. 2021). In the BEIR benchmark, GTR outperformed other dense retrieval models, such as DPR (Karpukhin et al. 2020), ANCE (Xiong et al. 2020), and ColBERT (Khattab & Zaharia 2020). In Chapter 6 of this thesis, we employ ANCE and TCT-ColBERT as the baseline models for clarifying question selection. These models serve as reference points to compare and evaluate the effectiveness of our proposed approach in the selection of clarifying questions using GTR.

2.4.3 Multiple Representation Bi-Encoder

Multiple Representations in Dense Passage Retrieval is a technique where each token in a passage is represented by its own embedding (Macdonald et al. 2021). This technique is used to compute the similarity between a query and a passage by identifying the nearest neighbours using a FAISS (Johnson et al. 2021) index. Multiple Representations are a type of dense retrieval approach that has emerged recently in the field of information retrieval. In multiple representation dense retrieval models, such as ColBERT (Khattab & Zaharia 2020) (Contextualized Late Interaction over BERT), each token within a passage is assigned a specific embedding as shown in Figure 2.2(c). This approach enables the model to capture the nuanced meanings and relationships of individual words within the context of the passage. The advantage of using Multiple Representations is that it provides a more fine-grained representation of the meaning of a passage, as each token is represented by its own embedding. This can lead to more accurate retrieval results, especially for longer passages. However, the disadvantage is that it increases the computational cost of indexing and retrieval, as multiple embeddings are required for each passage (Macdonald et al. 2021).

The introduction of multiple representation models has expanded the repertoire of dense retrieval approaches, providing additional flexibility and potential for improved retrieval performance. By capturing the semantic relationships at the token level, these models offer a more comprehensive representation of the text, enhancing the accuracy and relevance of retrieval results.

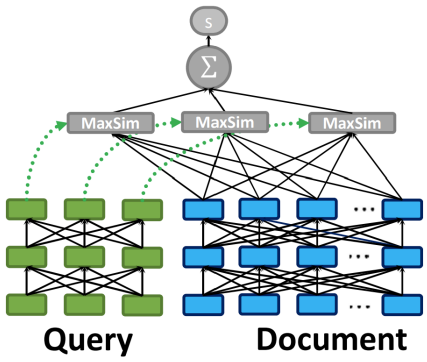


Figure 2.7: Schematic diagram of ColBERT, a late interaction paradigm. Figure taken from (Khattab & Zaharia 2020).

ColBERT (Khattab & Zaharia 2020) is an approach that adopts a late interaction model, where queries and documents are encoded independently and then interacted to compute relevance scores (see Figure 2.7)). This approach has several advantages over early interaction models, which compute the relevance score between the query and the document at the same time. ColBERT has been trained on a large MS-Marco corpus, and it uses the BERT model as its encoder. BERT (see Section 2.3.2) is a deep bidirectional transformer model. ColBERT has been shown to outperform other neural search models on a variety of passage retrieval tasks and outperforms BM25, DPR (Karpukhin et al. 2020), and other baselines (Khattab & Zaharia 2020).

These approaches, including ANCE, ColBERT, TCT-ColBERT, and GTR, contribute to the advancement of dense retrieval techniques and have demonstrated their effectiveness in improving retrieval performance. In Chapter 6 of this thesis, we employ ANCE, ColBERT, and TCT-ColBERT as the baseline models for clarifying question selection. These models serve as reference points to compare and evaluate the effectiveness of our proposed approach in the selection of clarifying questions using GTR.

2.5 PyTerrier

PyTerrier (Macdonald & Tonellotto 2020) is a framework for expressing IR experiments with composable pipelines. It allows complex retrieval pipelines to be specified in a declarative rather than procedural fashion, using standard operators to combine objects representing retrieval building blocks called transformers. PyTerrier differs from other IR frameworks in that it uses Python as a high-level language for operationalising experiments, similar to deep machine learning platforms such as Tensorflow [2] and PyTorch [3]. In addition, PyTerrier targets IR platforms as backends in order to execute and evaluate retrieval pipelines, and can automatically optimise the retrieval pipelines to increase their efficiency for a particular IR platform backend.

[2] https://www.tensorflow.org/ [3] https://pytorch.org/

Table 2.2: PyTerrier operators for combining transformers. Table taken from (Macdonald & Tonellotto 2020).

Operators	Name	Description
>>	then	Pass the output from one transformer to the next transformer
+	linear combine	Sum the query-document scores of the two retrieved results lists
*	scalar product	Multiply the query-document scores of a retrieved results list by a scalar
**	feature union	Combine two retrieved results lists as features
|	set union	Make the set union of documents from the two retrieved results lists
&	set intersection	Make the set intersection of the two retrieved results lists
%	rank cutoff	Shorten a retrieved results list to the first K elements
^	concatenate	Add the retrieved results list from one transformer to the bottom of the other

Table 2.2 lists the PyTerrier operators for combining transformers. The table includes the operator name, a brief description of the operator, and its functionality. In this thesis, namely Chapters 6-8, we use the PyTerrier (Macdonald & Tonellotto 2020) platform for indexing and retrieving passages.

2.6 Hybrid Sparse and Dense Retrieval

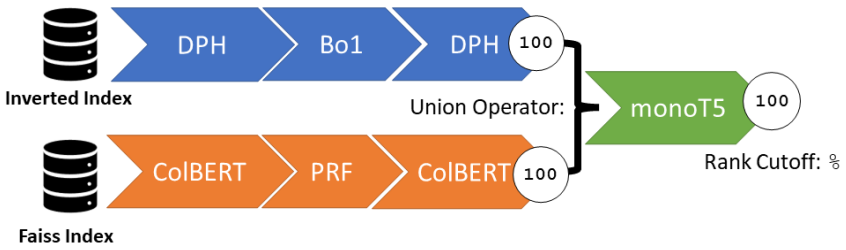


Figure 2.8: Hybrid of sparse and dense retrieval. Figure taken from (Wang, MacAvaney, Macdonald & Ounis 2021*a*).

Hybrid retrieval approaches combine both sparse and dense retrieval methods to leverage the strengths of each approach. Dense retrieval models are good at finding documents that are semantically similar to a given query. Sparse retrieval models are good at finding exact matches, even for long documents. Hybrid models combine the strengths of both dense and sparse retrieval models, to achieve better performance than either model alone. Several approaches (Lin & Ma 2021, Lin, Yang & Lin 2020, Luan et al. 2021*b*, Macdonald & Tonellotto 2020, Wang, Zhuang & Zuccon 2021, Wang, Wu, Wang, Macdonald & Ounis 2020) have been proposed to build hybrid models that combine the strengths of both dense and sparse retrieval models. In most cases, a dense retrieval model is trained separately and its scores are interpolated with those of a sparse model. This interpolation can be achieved by using techniques such as score interpolation or score combination. For example, Figure 2.8 presents a hybrid sparse and dense retrieval

architecture introduced by Wang, MacAvaney, Macdonald & Ounis (2021*a*). This approach combines sparse retrieval using DPH (Amati et al. 2008) with Bo1 (Amati & Van Rijsbergen 2002) query expansion and ColBERT-PRF (Wang, Macdonald, Tonellotto & Ounis 2021) dense retrieval on the ColBERT FAISS (Johnson et al. 2021) index. The retrieved passages/documents are further re-ranked using the monoT5 model (Nogueira, Jiang, Pradeep & Lin 2020*a*) (see Section 2.4.1). Following the Pyterrier notation as presented in Table 2.2, let I_{dense} and I_{sparse} denote the dense index and PyTerrier's sparse index of a given corpus, respectively. The hybrid of sparse and dense retrieval system as presented in Figure 2.8 can be defined as follows:

$$\begin{aligned} Ret_{\text{ColBERT-PRF}}(I_{dense}, k) \mid DPH_{w/QE}(I_{sparse,k}) \\ \text{»} monoT5(I_{sparse}, k) \end{aligned} \tag{2.8}$$

where $Ret_{\text{ColBERT-PRF}}(I_{dense}, k)$ is ColBERT dense retrieval.

This hybrid system aims to enhance retrieval performance by capturing both lexical similarity and semantic relationships, so as to improve the accuracy and relevance in passage/document retrieval tasks.

These efforts in developing hybrid retrieval models enable ORConvQA systems to benefit from the semantic modelling capabilities of dense retrieval models while also addressing their limitations, such as high computational cost and limited ability to generalise to new domains (Lin & Ma 2021, Luan et al. 2021*b*, Wang, Zhuang & Zuccon 2021). Following Wang, MacAvaney, Macdonald & Ounis (2021*a*), by effectively combining the strengths of dense and sparse retrieval models, hybrid models can offer more comprehensive and accurate retrieval results, enhancing the performance of ORConvQA systems in capturing both lexical and semantic aspects of queries and passages. In Chapter 7, we employ a hybrid approach that combines sparse and dense passage retrieval methods in order to achieve a more comprehensive and accurate information retrieval process for our ORConvQA system.

2.7 Classical IR Evaluation Metrics

Evaluation is an important part of information retrieval (IR) systems, as it provides a way to measure the effectiveness and performance of the system. Several classical evaluation metrics are commonly used in IR research including precision, recall, F1 score, MAP (Mean Average Precision), MRR (Mean Reciprocal Rank), and NDCG (Normalized Discounted Cumulative Gain). We discuss these metrics in the following section:

- **Precision** and **Recall** are two fundamental metrics used to evaluate the performance of a retrieval system. Precision measures the proportion of relevant documents among the retrieved documents, while Recall measures the proportion of relevant documents that are

retrieved. Precision and recall are calculated as follows:

$$Precision = \frac{\text{Number of Relevant Documents Retrieved}}{\text{Total Number of Retrieved Documents}} \tag{2.9}$$

$$Recall = \frac{\text{Number of Relevant Documents Retrieved}}{\text{Total Number of Relevant Documents}} \tag{2.10}$$

- **MAP** (Mean Average Precision) is a measure of the average precision over a set of queries (Voorhees 2003). It is calculated as the average of the precision values at each relevant document in the ranking:

$$MAP = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{num}_i} \sum_{j=1}^{\text{num}_i} \text{Precision@}j \tag{2.11}$$

  where Q is the set of queries, num_i is the number of relevant documents for query Q_i, and j iterates over the ranking of documents for query Q_i, from the most relevant to the least relevant. Precision@j represents the precision at rank j in the ranking for query Q_i.

- **MRR** (Mean Reciprocal Rank) measures the effectiveness of a retrieval system in returning the most relevant results for a query higher in the ranking (Radev et al. 2002, Voorhees 2003). MRR is defined as the average of the reciprocal ranks of the first relevant document retrieved over a set of queries. It is particularly useful for evaluating the effectiveness of a system in cases where only one relevant document is expected for a given query. MRR is considered to be a more robust metric than precision and recall, as it takes into account the order of the retrieved documents. While MRR is useful when only one relevant document is expected per query, whereas MAP is suitable for scenarios with multiple relevant documents (Voorhees et al. 1999). Both metrics assess the effectiveness of retrieval systems but focus on different aspects: the rank of the first relevant document (MRR) and precision across the ranking (MAP). The formula for calculating the MRR is as follows:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{2.12}$$

  where Q is the set of queries, and rank_i is the rank of the first relevant document retrieved for the i-th query.

- **NDCG** (Normalised Discounted Cumulative Gain), proposed by Järvelin & Kekäläinen (2002), is a measure of the quality of a ranking of documents (Voorhees 2003). It takes into account both the relevance of the documents and their position in the ranking. NDCG can be used in various scenarios where the order of relevant retrieved documents matters, such as web search, recommendation systems, and conversational search. NDCG is calculated as follows:

$$NDCG@k = \frac{1}{Z_k} \sum_{i=1}^{k} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)} \tag{2.13}$$

where k is the number of documents that are included in the calculation, rel_i is the relevance of the document at position i, and Z_k is a normalisation factor that ensures that the NDCG score is between 0 and 1.

NDCG is particularly useful in evaluating retrieval systems that return a ranked list of documents, where the relevance of the documents varies. For example, in web search, some documents may be highly relevant to the user's query, while others may be only marginally relevant or not relevant at all. In this case, NDCG provides a way to evaluate the effectiveness of the retrieval system in ranking the relevant documents higher in the list, while downgrading the importance of less relevant documents.

In Open-Retrieval Conversational Question Answering (ORConvQA) systems, the goal is to retrieve relevant passages or documents that provide answers to a given query or question. These systems often use retrieval techniques, such as sparse retrieval (e.g., TF-IDF, BM25) or dense retrieval (e.g., TCT-ColBERT), to identify passages or documents that are likely to contain the answer to the query. Once the passages or documents have been retrieved, they need to be ranked based on their relevance to the query. This is where NDCG can be useful. NDCG takes into account both the relevance of the passages and their position in the ranking. The relevance of a passage or document is typically measured by a relevance score, which reflects the degree to which the passage or document answers the query. By using NDCG, Open-Retrieval Conversational Question Answering (ORConvQA) systems can evaluate the effectiveness of their retrieval techniques in ranking the passages or documents based on their relevance to the query, while taking into account the order of the retrieved passages (Dalton et al. 2020, 2021).

2.8 Multi-Task Learning

Multi-Task Learning (MTL) (Ruder 2017) is an approach in machine learning that aims to improve the performance of multiple related tasks by jointly learning them in a single model. In traditional single-task learning, each task is typically treated as an independent problem and trained separately. However, in many real-world scenarios, there exist inherent relationships and dependencies among different tasks that can be leveraged to enhance learning efficiency and generalisation.

In this section, we explore Multi-Task Learning (MTL) and compare it with other learning approaches such as Transfer Learning (Yang et al. 2020), Prompt-based Learning (Liu et al. 2023), Multi-Label Learning (Zhang & Zhou 2014), and Multi-view Learning (Sun et al. 2017, Wang et al. 2011, Zhao et al. 2017). By examining the distinctions between MTL and these approaches,

we gain insights into how MTL offers unique advantages and benefits in jointly learning multiple related tasks, as is needed in the ORConvQA systems.

2.8.1 Learning Taxonomy

In the field of machine learning, various approaches and techniques have been developed to tackle different learning tasks. This section provides an overview of several learning paradigms that are relevant to the research conducted in this thesis, including their connections and relevance to the concept of Multi-Task Learning (MTL). By exploring these learning paradigms, we can gain insights into their applicability and potential synergies with MTL in addressing complex real-world challenges.

- **Transfer Learning** (Yang et al. 2020) aims to leverage knowledge acquired from one task or domain to improve learning on another related task or domain. It recognises that models trained on large-scale datasets can capture generic features and knowledge that can be reused in different contexts. By transferring learned knowledge, models can potentially benefit from reduced training time, improved performance, and enhanced generalisation capabilities. Transfer learning techniques include fine-tuning pre-trained models, using feature extraction layers, and domain adaptation methods (Yang et al. 2020).

- **Multi-label Learning** (Zhang & Zhou 2014) deals with scenarios where each instance can be associated with multiple labels simultaneously. This learning paradigm is particularly useful in domains where instances can have multiple attributes or labels. It requires models to learn to predict multiple output variables, where each label may have varying degrees of relevance or importance. Applications of multi-label learning can be found in text classification, image annotation, and recommendation systems (Zhang & Zhou 2014).

- **Multi-View Learning** (Sun et al. 2017, Wang et al. 2011, Zhao et al. 2017) refers to learning tasks that involve multiple distinct sets of features or views for each instance. These views provide complementary information about the data, and the goal is to learn a unified representation or model that effectively exploits the information from all views. Multi-view learning is commonly used in multimedia analysis, where an instance can have different representations, such as text, images, or audio, each capturing different aspects of the data (Sun et al. 2021).

- **Prompt-based Learning** (Liu et al. 2023) is a new paradigm in natural language processing (NLP) that involves pre-training a language model to predict text based on a given prompt. Unlike traditional supervised learning, which trains a model to take in an input x and predict an output y as $P(y|x)$, prompt-based learning is based on language models that model the probability of text directly. The process involves modifying an input x into a textual string prompt x' that is used to generate text. Prompt-based learning has been

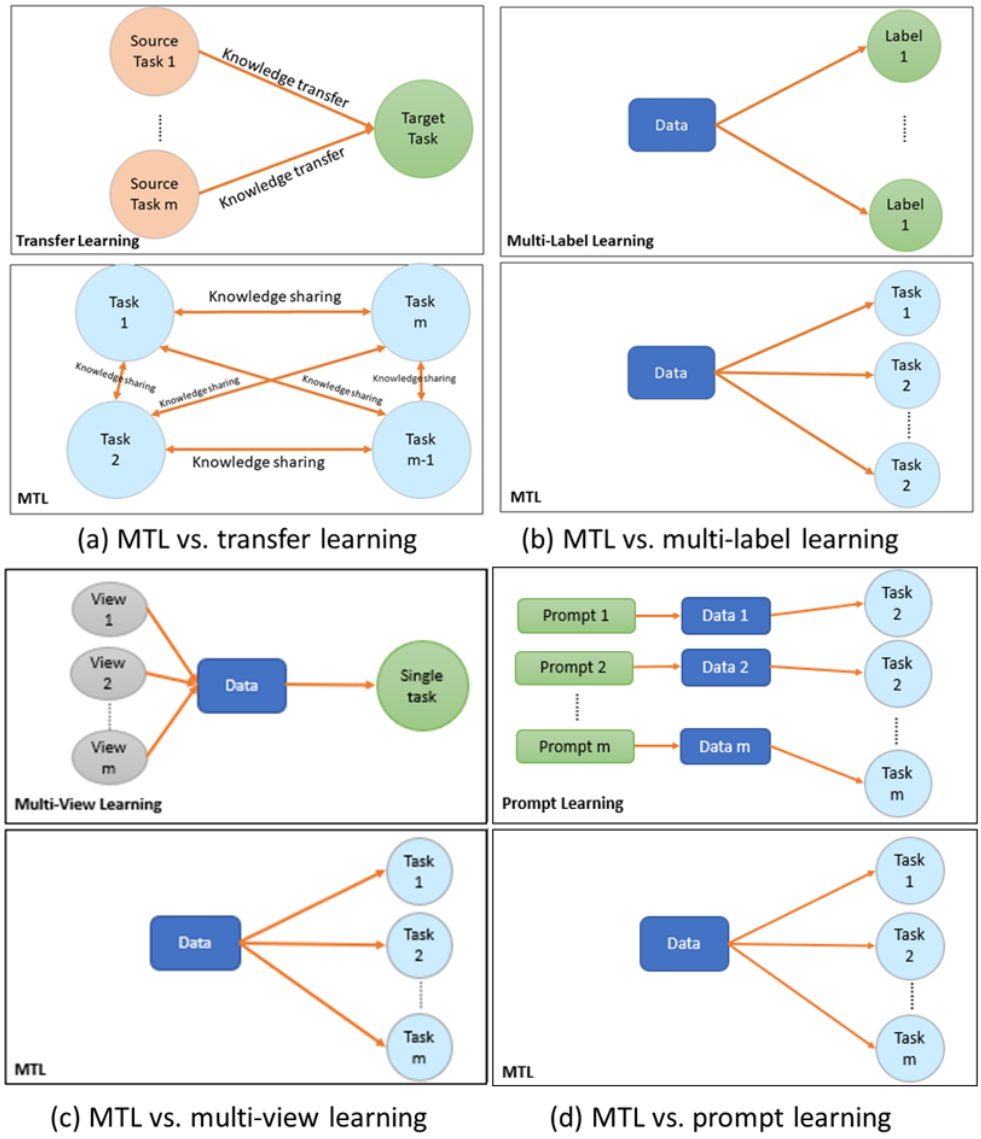


Figure 2.9: Comparison of Multi-Task Learning (MTL) and Other Learning Paradigms. Figure adapted from (Zhang & Yang 2021).

applied to a variety of NLP tasks and has shown promising results in few-shot and zero-shot learning scenarios (Brown et al. 2020).

- **Multi-Task Learning** (Crawshaw 2020, Zhang & Yang 2021) involves training a model to perform multiple related tasks simultaneously, with the idea that learning multiple tasks jointly can lead to improved performance on each individual task. By sharing information and representations across tasks, models can leverage the commonalities and dependencies between tasks to enhance learning. Multi-task learning can provide benefits such as better generalisation, improved efficiency, and increased robustness (Ruder 2017).

Figure 2.9 presents a comprehensive comparison of Multi-Task Learning (MTL) with other learning paradigms, such as Transfer Learning, Multi-Label Learning, Multi-View Learning, and Prompt-based Learning. Each paradigm exhibits distinct approaches and objectives (Zhang & Yang 2021).

In Figure 2.9 (a), Transfer Learning and MTL both leverage knowledge from one task to benefit another. Transfer Learning takes knowledge from an initial task (source) and applies it to a different but related task (target), often by fine-tuning a pre-trained model (Yang et al. 2020). In contrast, MTL simultaneously trains a model on multiple tasks, sharing information across them to boost performance on all tasks. In addition, whilst Transfer Learning focuses on using past knowledge for a new task, Multi-Task Learning optimises several tasks together (Zhang & Yang 2021).

In addition, Figure 2.9 (b) shows that both Multi-Label Learning and MTL handle multiple outputs, but in different ways. Multi-Label Learning predicts several labels for one instance, like tagging a movie as both "Action" and "Adventure". On the other hand, MTL trains a model on multiple tasks at once, such as identifying objects in a photo while also classifying the scene (Zhang & Zhou 2014). Hence, while Multi-Label Learning focuses on multiple labels for one task, Multi-Task Learning aims to improve performance across several related tasks by training them simultaneously (Zhang & Yang 2021).

Figure 2.9 (c) also compares Multi-View Learning and MTL, which are both advanced machine learning paradigms that use multiple sources of data or objectives, but they differ in their goals and application. Multi-View Learning focuses on exploiting data from different views or sources (like audio and video) for the same task, aiming to build a comprehensive representation of data (Wang et al. 2011). In contrast, Multi-Task Learning involves training a model on multiple related tasks simultaneously to improve performance on each, by leveraging shared patterns or features. Hence, while Multi-View capitalises on diverse data sources for a single task, Multi-Task Learning optimises several tasks together, often through shared model parameters (Zhang & Yang 2021).

Moreover, Figure 2.9 (d) presents the difference between Prompt-based Learning and MTL. Prompt-based learning and MTL are two techniques that can be used to improve the performance of machine learning models. Prompt-based learning uses carefully designed inputs to guide a model's output, while multi-task learning trains a model on multiple related tasks simultaneously (Liu et al. 2023, Zhang & Yang 2021).

In conclusion, Figure 2.9 presents a comparison, focusing on the unique characteristics of Multi-Task Learning (MTL) in contrast to other learning paradigms including Transfer Learning, Multi-Label Learning, Multi-View Learning, and Prompt-based Learning. In this thesis, we propose to develop an effective ORConvQA by leveraging MTL. Compared to other learning paradigms, MTL can help ORConvQA systems to simultaneously optimise multiple related tasks, leverage shared knowledge structures, and improve generalisation capability, resulting in

enhanced performance in ORConvQA. In Chapters 5, 6, 7, and 8, we show how MTL allows to improve the performance of ORCOnvQA on various tasks, including answer extraction, follow-up question identification, conversational question rewriting, passage retrieval, passage reranking, clarification need classification, and asking clarifying question.

2.8.2 MTL Taxonomy

As mentioned in Section 2.8, Multi-Task Learning (MTL) is a machine learning approach that aims to jointly train a model on multiple related tasks, leveraging the shared knowledge and dependencies between them. By learning from multiple tasks simultaneously, MTL can improve generalisation, enhance performance on individual tasks, and reduce the need for separate models for each task (Crawshaw 2020). To effectively implement MTL, several important aspects need to be considered, including the optimisation strategy methods and parameter sharing approaches. These aspects are essential in determining how the MTL model learns and shares information across tasks.

In this section, we provide an overview of the optimisation strategy methods in MTL, specifically focusing on static vs dynamic task weighting. We also discuss the parameter sharing techniques employed in MTL, distinguishing between hard and soft parameter sharing. Understanding these foundational aspects of MTL taxonomy will pave the way for a comprehensive understanding of multi-task learning and its application in various domains.

2.8.2.1 Optimisation Strategy Methods in MTL

Multi-Task Learning (MTL) involves training models on several tasks simultaneously. The optimisation strategy, which determines the importance assigned to each task during training, is important for training MTL models on multiple tasks. Task weighting can be done in two ways: static task weighting and dynamic task weighting. Static task weighting assigns fixed weights to each task during training, while dynamic task weighting assigns weights to tasks that are updated during training. We discuss static and dynamic task weighting in detail in the remainder of this section:

Static task weighting involves assigning fixed weights to each task throughout the training process. These weights can be based on prior knowledge about the relative importance of tasks or set equally to treat all tasks equally, such as setting all of them to 1.0 (Yeh & Chen 2019), or setting their sum to 1 (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019). The total loss function of this method is defined as follows:

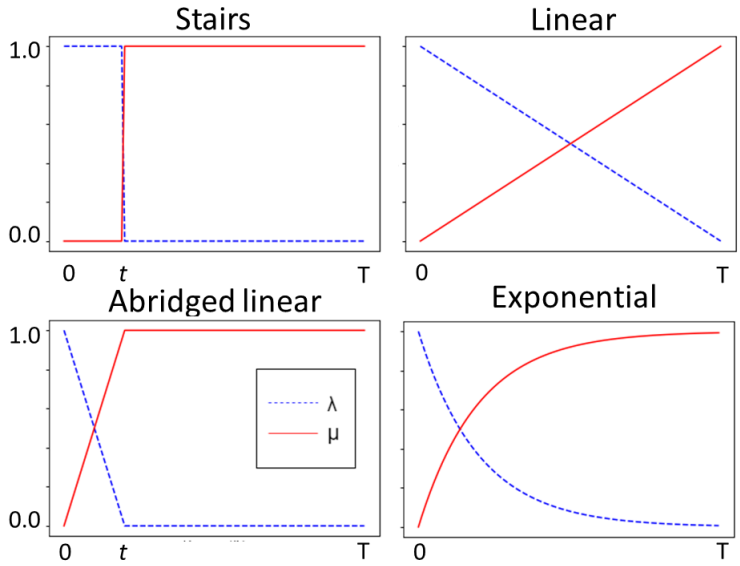


Figure 2.10: Dynamic Evolving Weighting approaches, Figure adapted from (Belharbi et al. 2016).

$$\mathcal{L}_{total} = \mu \mathcal{L}_{ans} + \lambda \sum_{a \in A} \mathcal{L}_a \tag{2.14}$$

where A is the set of auxiliary tasks, μ is the weight for the main task and λ is the weight for A. Static task weighting provides a straightforward and intuitive approach to MTL. However, it may not fully capture the varying complexities and contributions of different tasks, potentially limiting the model's ability to optimally allocate resources during training (Crawshaw 2020).

Dynamic task weighting adaptively adjusts the importance assigned to each task based on its performance or difficulty. This approach allows the model to allocate more resources and focus on challenging or more informative tasks during training. Dynamic task weighting can be implemented using various techniques, such as reinforcement learning (Liu, Liang & Gitter 2019) or adaptive loss scaling (Chen et al. 2018). By dynamically updating task weights, the model can adapt and prioritise the learning process to maximise overall performance across multiple tasks. Examples of dynamic approaches are Evolving Weighting (Belharbi et al. 2016), Loss-Balanced Task Weighting (Liu, Liang & Gitter 2019), and Uncertainty Weighting (Kendall et al. 2018), discussed further below.

- **Evolving Weighting:** Belharbi et al. (2016) proposed to evolve the loss weighting during the training steps according to a *schedule*. A training step is defined as the number of batches of the training data, such that the total number of steps is the number of batches multiplied by the number of training epochs. Four different schedules were proposed. Figure 2.10 gives an overview of how the four schedules vary the weights of the main and auxiliary tasks – μ and λ, respectively – across the training steps. These four schedules are described below:

Stairs schedule: The initial emphasis is on the auxiliary task, with $\mu = 0$ and $\lambda = 1$. At a given training step t, $\mu = 1$ and $\lambda = 0$.

Linear schedule: The weight of the auxiliary task decreases linearly at each training step, such that the auxiliary weight $\lambda = 1$ tends to 0; in contrast, the weight of the main task increases linearly, i.e. $\mu = (1 - \lambda)$. In particular, given that the total number of steps T is known in advance, $\mu_t = \frac{t}{\mathrm{T}}$, where μ_t is the weight of the main tasks at training step t. This approach is typically employed when the aim is to incrementally enhance the importance of the main task in a linear manner throughout the training process.

Abridged Linear schedule: In a linear schedule, μ rises over the full training schedule to step T. This may not place sufficient emphasis on the main task during training. Instead, in the Abridged Linear schedule the weight on the auxiliary task λ falls linearly to 0 by a threshold step t_τ. After t_τ, all emphasis is on the main task (i.e. $\mu = 1$).

Exponential schedule: The weights evolve exponentially to the step number, i.e. $\mu = exp(\frac{-t}{\sigma})$, where t is the current number of training steps, and σ is the slope, as shown in Figure 2.10.

- **Loss-Balanced Task Weighting (LBTW) (Liu, Liang & Gitter 2019):** This MTL method aims to reduce *negative transfer* by using the task-specific loss to balance the different auxiliary tasks. Negative transfer is when the performance of the task is decreased by Multi-Task Learning compared to the Single-Task Learning. This method employs the *loss ratio* between the current loss and the initial loss of each task to adjust the task's weight. The task with the loss ratio closest to 1 needs to contribute more to the total loss. By increasing the weight of the task with the loss ratio that is closest to 1, this method attempts to *balance* the task importance.

- **Uncertainty Weighting (Kendall et al. 2018):** This method is the most often used Multi-Task Learning approach, which is a weighting strategy that consists in analysing the uncertainty of each task. In this method, each of the task's weights is adjusted by deriving a multi-task loss function when maximising the Gaussian likelihood (Ruder 2017).

Both static and dynamic task weighting have their advantages and trade-offs in MTL (Crawshaw 2020). Static task weighting offers simplicity and interpretability, making it easier to define and control the relative importance of tasks. It is particularly useful when prior knowledge about task importance is available or when all tasks are expected to contribute equally. However, it may not effectively capture variations in task difficulty or adapt to changing task dynamics during training. On the other hand, dynamic task weighting enables the model to flexibly allocate resources to tasks based on their performance or inherent complexity. This adaptability enhances the model's ability to learn and allocate resources efficiently, leading to potentially improved

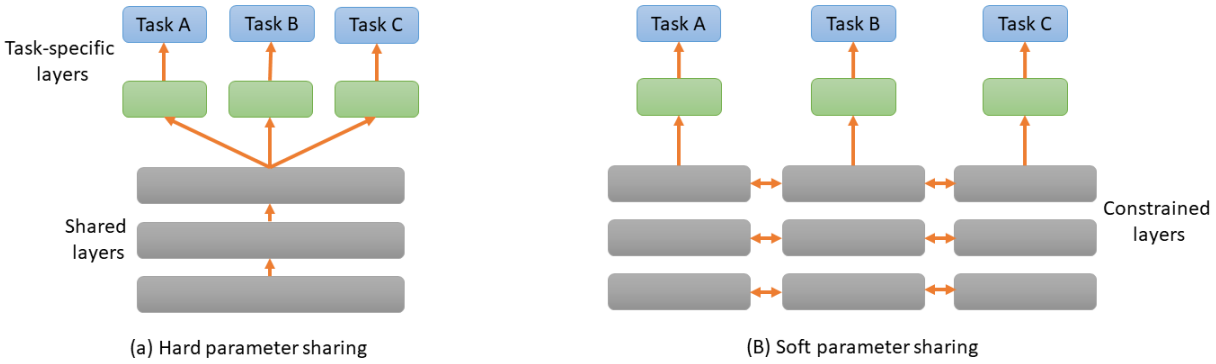


Figure 2.11: Comparison of Hard and Soft Parameter Sharing in Multi-Task Learning (Ruder 2017).

performance. However, dynamic task weighting approaches introduce additional complexity in terms of implementation and may require additional computational resources.

2.8.2.2 Parameter Sharing in MTL

Multi-Task Learning (MTL) involves sharing and reusing parameters in a model across various tasks. The way the model shares these parameters markedly impacts its capability to leverage shared knowledge and relationships between tasks. Figure 2.11 presents two common methods for parameter sharing in MTL, which are hard parameter sharing and soft parameter sharing (Ruder 2017).

Hard parameter sharing is a commonly used approach for Multi-Task Learning (MTL) within neural networks, which can be traced back to the research conducted in 1993 by Caruana (1993). This approach involves using shared layers or components across all tasks in the MTL model. The shared layers are responsible for capturing common features and representations that are relevant to multiple tasks. By sharing parameters, the model is forced to learn a unified representation that is shared among all tasks, as shown in Figure 2.11 (a). Hard parameter sharing enables efficient knowledge transfer among tasks, particularly when they share similar inputs or patterns. It simplifies model complexity and boosts training and inference efficiency. Hard parameter sharing effectively minimises overfitting. As Baxter (1997) found, the overfitting risk for the shared parameters, where N is the task count, is significantly lower than for task-specific parameters. This is intuitive: As the model learns more tasks concurrently, it develops a representation fitting all tasks, thus reducing the likelihood of overfitting.

Soft parameter sharing (Ruder 2017) enables each task to have its own set of parameters while still allowing for some degree of sharing, as presented in Figure 2.11 (b). In this approach, the model incorporates regularisation techniques (Duong et al. 2015) or track norm (Yang & Hospedales 2017) to encourage the parameters of different tasks to be similar or to converge towards a shared space. Soft parameter sharing offers flexibility in capturing both shared and task-specific information (Worsham & Kalita 2020). It allows the model to adapt to variations in

task characteristics and provides the capacity to learn task-specific representations. However, it can increase the model's complexity and training time, as each task has its own set of parameters to optimise.

The decision to use either hard or soft parameter sharing depends on the nature of the tasks and the amount of shared knowledge between them. Hard parameter sharing is simpler and more common, but soft parameter sharing allows more flexibility and specialisation for each task. Both approaches aim to improve the generalisation and representation of the network by learning from multiple tasks (Worsham & Kalita 2020). In this thesis, following Liu et al. (2015), Liu, He, Chen & Gao (2019), Xu et al. (2019*a*), we employ a hard parameter sharing MTL approach because this network type reduces the risk of overfitting (Ruder 2017).

2.9 Conclusions

In this chapter, we explored various aspects that lay the background for effective Open-Retrieval Conversational Question Answering (ORCOnvQA) leveraging Multi-Task Learning (MTL). We began by delving into sparse retrieval (Section 2.2), which involved the retrieval of relevant passages based on term frequencies and document relevance scores. Sparse retrieval approaches, such as BM25, have been widely used and provide a solid baseline for information retrieval systems. Next, we discussed the significance of Pre-trained Language Models (PLMs) (Section 2.3) in the field of natural language processing (NLP) and information retrieval (IR). PLMs, such as BERT (see Section 2.3.2), T5 (see Section 2.3.1), and BART (see Section 2.3.1), have revolutionised language understanding by capturing contextual information and semantic meaning. These models serve as tools for improving the performance of Open-Retrieval Conversational Question Answering systems. Dense retrieval (Section 2.4) has emerged as another important approach to information retrieval. Dense retrieval models, such as ANCE (see Section 2.4.2), ColBERT (see Section 2.4.3), TCT-ColBERT (see Section 2.4.2), and GTR (see Section 2.4.2), leverage dense vector representations to capture semantic similarities between queries and passages. The integration of dense retrieval techniques complements the limitations of sparse retrieval and enhances the accuracy and relevance of retrieved passages. We then explored the concept of hybrid sparse and dense retrieval (Section 2.4), which combines the strengths of both approaches. By leveraging the complementary aspects of sparse and dense retrieval, hybrid models aim to achieve more comprehensive and precise retrieval results in ORConvQA. Section 2.7 provided a detailed overview of the different evaluation metrics that are commonly used in IR and which are typically used in ORConvQA. Evaluation metrics, including Precision, Recall, MAP, MRR, and NDCG, are commonly used to compare the performance of the different IR systems. Lastly, we explored the potential of MTL (Section 2.8) in ORConvQA. MTL allows for joint training of multiple related tasks, enabling knowledge sharing and improved generalisation. By leveraging MTL, we can enhance the performance of the ORConvQA systems.

Overall, in order to develop an effective ORConvQA leveraging MTL, it is important to have a solid understanding of the background and approaches associated with Sparse Retrieval, PLMs, Dense Retrieval, Hybrid Dense and Sparse Retrieval, Classical Evaluation Metrics in IR, and MTL. This comprehensive background knowledge sets the stage for the subsequent chapters, where we delve into specific methodologies and experiments to explore the benefits and challenges of multi-task learning in the context or ORConvQA. With this comprehensive background knowledge, we are now ready to delve into the related work in the next chapter. Indeed, Chapter 3 examines existing research and methodologies in the field of Open-Retrieval Conversational Question Answering.

Chapter 3

Related Work

3.1 Introduction

As previously discussed in Section 1.3, the aim of this thesis is to develop an Open-Retrieval Conversational Question Answering (ORConvQA) system using Multi-Task Learning (MTL). Recall that, as stated in Section 1.1, ORConvQA systems differ from traditional search engines by allowing users to interact in a more natural and conversational manner. For example, instead of submitting a search query like "Famous scientists" to a search engine, users can ask, "Who are some famous scientists?" and follow up with questions like, "What are their notable discoveries?" based on the system's previous responses. This conversational approach offers a more intuitive and user-friendly method for information retrieval.

However, as outlined in Section 1.2, effectively answering the users' questions in ORConvQA comes with several challenges, including:

- **Ambiguities in conversational questions:** Questions in conversations can be ambiguous or unclear, requiring the system to use prior conversation context or ask for more details to provide accurate answers.
- **Retrieve relevant passages:** In a large-scale collection of passages, efficient and accurate retrieval of relevant passages is important.
- **Identify the most relevant passages:** Among the passages retrieved, determining which passages are the most relevant in the context of the ongoing conversation is a challenging task.
- **Extract answers:** Once a list of relevant candidate passages is formed, extracting the most accurate and relevant answers represents the final challenge.

To address the challenges of ORConvQA, this thesis investigates several sub-tasks, including: i) Conversational Question Answering: to gauge the current question in the context of the

conversation history and to provide a relevant answer; ii) Follow-up Question Identification: to identify questions that are follow-ups to previous questions, helping in context understanding; iii) Conversational Question Rewriting: to reformulate a concise conversational question into a fully specified query that existing information retrieval systems can adequately handle; iv) Clarification Need Classification: to determine when a user's query is ambiguous and may require additional clarification; v) Asking Clarifying Questions: to interact with the user to gain additional context or clarification; vi) Passage Retrieval: to efficiently search through a large collection of documents to retrieve relevant passages; vii) Passage Reranking: to reorder the list of retrieved passages based on their relevance to the ongoing conversation.

In addition, given the multifaceted challenges and tasks required to effectively address ORConvQA, this thesis argues that Multi-Task Learning (MTL) has a good potential as an integrated solution. Recall from Chapter 2, Section 2.8, that MTL has become a popular method for simultaneously tackling multiple related tasks within a single model (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019). Indeed, multi-task learning can enhance generalisation performance, reduce overfitting, and improve efficiency by encouraging the model to learn more general features that are relevant across multiple tasks while sharing parameters between them (Collobert & Weston 2008, Ruder 2017). Through Multi-Task Learning (MTL), our proposed system aims to integrate these sub-tasks to develop a more effective ORConvQA system.

Therefore, to distinguish our main contributions from the existing work, in this chapter, we explore all seven aforementioned tasks in the literature by providing their definitions, datasets, evaluation metrics, as well as describing the existing approaches. In addition, we identify the gaps in the literature and discuss how MTL can be leveraged to address them.

The remainder of this chapter is organised as follows:

- Section 3.2 presents an overview of Conversational Question Answering (ConvQA) exploring how to gauge the current question within the conversation history and provide relevant answers;
- Section 3.3 describes Follow-up Question Identification (FID) identifying questions that are follow-ups to previous questions, thereby helping in context understanding;
- Section 3.4 presents Conversational Question Rewriting (QR), which emphasises the reformulation of conversational questions into queries that existing information retrieval systems can process effectively;
- Section 3.5 discusses Clarification Need Classification (CNC), which is concerned with how to determine when a user's query is ambiguous and may need further clarification;
- Section 3.6 describes Asking Clarifying Questions, an important task for interacting with users to gain additional context.

- Section 3.7 presents Passage Retrieval, exploring techniques for retrieving relevant passages from large document collections;
- Section 3.8 discusses Passage Reranking, which involves reordering retrieved passages based on their relevance to the ongoing conversation;
- Section 3.9 presents Multi-Task Learning in ORConvQA, exploring how to combine the tasks for a more effective ORConvQA system and identifying the research gaps that this thesis aims to address;
- Finally, Section 3.10 gives the concluding remarks for this chapter, summarising the identified research gaps from Section 3.9.

3.2 Conversational Question Answering (ConvQA)

Conversational Question Answering (ConvQA) (Choi et al. 2018) is a subset of Machine Reading Comprehension (MRC) (Rajpurkar et al. 2018, 2016) where questions are asked in the context of ongoing conversations. ConvQA simplifies the Open-Retrieval Conversational Question Answering (ORConvQA) task by providing a relevant passage from which the answer is extracted. To effectively tackle the ConvQA task, a system must effectively handle the conversation history to understand and answer the current question accurately. In the following sections, we provide the ConvQA task definition in Section 3.2.1. In Section 3.2.2, we discuss the ConvQA dataset. Then, we explore how to evaluate the ConvQA task in Section 3.2.3. Finally, the existing approaches are reviewed in Section 3.2.4.

3.2.1 ConvQA Task Definition

Following Choi et al. (2018), we describe the ConvQA task as follows: given a passage p, a conversation history H_k consisting of a list of k questions and ground truth answer pairs, i.e. $H_k = [\langle q, a \rangle]$, and a new query q_{k+1}, the task is to predict answer a_{k+1} by predicting answer span indices i, j within passage p. Table 3.1 exemplifies the ConvQA task, showing an example passage p, and conversation history of length $k = 2$ with corresponding questions and answers; In particular, in response to question q_3, the aim of a ConvQA system is to correctly predict the right answer a_3 from all possible sentences in p.

3.2.2 ConvQA Datasets

In this section, we provide a detailed comparison of well-known datasets for Machine Reading Comprehension (MRC) tasks including SQuAD [1] (Rajpurkar et al. 2018), CoQA [2] (Reddy et al.

[1] https://rajpurkar.github.io/SQuAD-explorer/ [2] https://stanfordnlp.github.io/coqa/

Table 3.1: An example dialog from the ConvQA dataset.

p	In 1934 he batted .344 with 18 home runs, 104 RBI, 102 runs scored and 192 hits in 138 games. After a disappointing final season with the White Sox which saw Simmons bat just .267 with 16 home runs and 79 RBI in 128 game (first time in his 11-year career he did not reach .300+ & 100 RBI) he rebounded by hitting .327 with 13 home runs, 112 RBI and 96 runs scored in 1936 for the Detroit Tigers. In 1937 he struggled again, this time with the Washington Senators, batting just .279 with 8 home runs and 84 RBIs in 103 games. He rebounded with a stellar season in 1938, batting .302 with 21 home runs and 95 RBI in just 125 games for Washington. His 21 home runs that year gave Simmons the distinction of being the first player to hit 20 home runs in a year for the Senators. CANNOTANSWER
q_1	Where was he playing in 1933?
a_1	CANNOTANSWER
q_2	What did he do between 1933 and 1938?
a_2	In 1934 he batted .344 with 18 home runs, 104 RBI, 102 runs scored and 192 hits in 138 games.
q_3	Did he lead the league in hitting?
a_3	After a disappointing final season with the White Sox

2019), and QuAC [3] (Choi et al. 2018). According to Yatskar (2019), these datasets are similar in using Wikipedia articles as their sources, asking questions about a provided passage, and extracting the answer from the text or providing no answer (e.g., $a_1 : CANNOTANSWER$ in the example of Table 3.1). The three datasets differ in their handling of unanswerable questions, dialogue, and abstractive answers:

- Unanswerable questions: SQuAD 2.0 contains the most diverse set of unanswerable questions, while QuAC focuses on questions that could plausibly be answered by the passage. CoQA, on the other hand, contains only a few unanswerable questions.
- Dialogue: SQuAD is a single-turn dataset, while QuAC and CoQA both contain dialogues. In CoQA, the questions often drill into the details of a topic, covering up to 60% of the context sentences. In QuAC, on the other hand, the questions often shift to new topics, covering less than 30% of the context sentences.
- Abstractive answers: Abstractive answers refer to responses that are not directly extracted from the provided passage. Both QuAC and CoQA contain the same rate of yes/no questions (ie., questions where the answer is either yes or no). QuAC has no abstractive answers, while CoQA includes a small number of predominant insertions, which are additional words or phrases in the answers that are not directly extracted from the text but are inserted to make the response more coherent or complete.

In the following, we describe each dataset in detail:

SQuAD (Stanford Question Answering Dataset) (Rajpurkar et al. 2018, 2016) is a dataset for training and evaluating machine reading comprehension models. Each item in this dataset consists of a passage from a Wikipedia article and a set of questions about the passage. The

[3] https://quac.ai/

answer to each question is a span of the passage. One unique feature of SQuAD is that it includes unanswerable questions. This introduces the challenge of not only answering questions but also determining when a question cannot be answered based on the given passage. SQuAD is a single-turn dataset, meaning that each question is answered with a single sentence from the context. The dataset contains over 100,000 question-answer pairs, and it is divided into a training set and a development set. SQuAD is a widely used benchmark dataset for evaluating the performance of question answering systems.

QuAC (Question Answering in Context) (Choi et al. 2018) is a widely-used Conversational Question Answering (ConvQA) dataset that encompasses various types of questions, including both non-factoid and factoid questions. This diverse question set challenges models to reason and extract relevant information from the context in order to provide accurate answers. includes three auxiliary tasks: Yes/No Prediction, Follow-up Question Identification, and Unanswerable Prediction. These auxiliary tasks add complexity and depth to the dataset, facilitating an exploration of ConvQA. However, the dialogues in QuAC often shift topics, thereby increasing the complexity of context understanding for models. This dataset also contains unanswerable questions, though these questions are specifically designed to focus on information that could be plausibly present in the passage. Unlike CoQA, QuAC does not contain abstractive answers beyond "yes" or "no" responses.

As discussed in Section 1.2, our focus in this thesis is on multi-turn conversation question answering. Therefore, we have chosen not to use the SQuAD dataset. The two large-scale ConvQA datasets, QuAC (Choi et al. 2018), and CoQA (Reddy et al. 2019), have facilitated further research on this task. As mentioned above, the differences between these datasets are that the questions in CoQA are predominantly factoid in nature, while most questions in QuAC are non-factoid. Moreover, QuAC also contains three auxiliary tasks; in contrast, CoQA only provides an Unanswerable prediction task as an auxiliary task. Hence, due to the presence of multiple auxiliary tasks, our MTL study focuses on the QuAC dataset in Chapter 5. Meanwhile, in Chapters 6, 7, and 8, we use datasets adapted from the QuAC dataset, such as OR-QuAC (Qu et al. 2020), LIF (Kundu et al. 2020), and CANARD (Elgohary et al. 2019), which are specifically tailored to the tasks and objectives of those chapters.

3.2.3 Evaluation of ConvQA Systems

The evaluation of Conversational Question Answering (ConvQA) systems involves assessing their ability to accurately answer questions within the context of ongoing conversations. The word-level F1 and the human equivalence (HEQ) scores are the metrics typically used in the literature. The Word-level F1 score measures the overlap between the predicted and the actual responses at the word level, while an EM score is used to evaluate whether the system's response is exactly the same as the actual response. The details of these three metrics are as follows:

1. **Word-level F1**, commonly used in Machine Comprehension and in the ConvQA tasks (Choi et al. 2018, Rajpurkar et al. 2018, 2016), evaluates the overlap between the system's prediction and the ground truth answer span. The word-level F1 score is calculated as follows:

$$\text{Precision} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3.1}$$

   where Precision is the fraction of words in the predicted answer that are also in the ground-truth answer and Recall is the fraction of words in the ground-truth answer that are also in the predicted answer.

2. **The Human Equivalence Score (HEQ)** is used to evaluate the percentage of examples for which the deployed model's F1 is equivalent to or higher than the human word-level F1. This metric comprises two components: HEQ-Q and HEQ-D. HEQ-Q is calculated at the question level, assessing whether the model's F1 score for each individual question is at least as high as the human F1 score. HEQ-D, on the other hand, is computed at the dialogue level, evaluating whether the model's F1 score across an entire dialogue equals or outperforms the human performance. The QuAC (Choi et al. 2018) challenge defines human performance to have an HEQ-Q and HEQ-D of 100.

3. **Exact Match (EM)** calculates the percentage of questions for which the model's answer matches the ground truth answer exactly. It is a stringent measure of correctness (Rajpurkar et al. 2018, 2016).

In this thesis, Chapters 5 and 8 employ word-level F1, HEQ-Q, and HEQ-D as evaluation metrics for assessing performance in conversational question answering tasks. However, in Chapters 6 and 7, these metrics are not used as these chapters focus on information retrieval (IR) tasks, which require different evaluation metrics (see Section 2.7).

3.2.4 Approaches for ConvQA

Different studies have employed various approaches for handling conversation history. For example, Reddy et al. (2019), Zhu et al. (2018), have appended the preceding question-answer pairs to the current question. On the other hand, Qu, Yang, Qiu, Croft, Zhang & Iyyer (2019), Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer (2019), Yeh & Chen (2019) all adopted a history selection mechanism. This mechanism selectively incorporates relevant parts of the conversation history, rather than using all conversation history. In addition, several studies have directly integrated the conversation history into neural language models like BERT (as introduced in Section 2.3.2). For example, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer (2019) introduced the Positional History Answer Embedding (PosHAE) approach, which uses a feature vector to encode the answer's position within the conversation history relative to the current enquiry. Similarly, Choi et al.

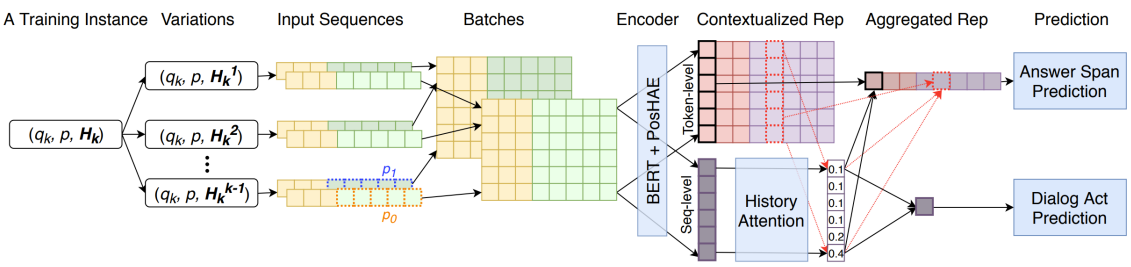


Figure 3.1: An architecture of the HAM model. Figure taken from (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019).

(2018), Yeh & Chen (2019) employed a Context Feature to signify historical answers within the passage. We describe the existing ConvQA models in detail as follows:

HAM (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019) proposed a solution for the ConvQA task by using the prior answers to understand and answer the current question. In this system, there were three features including Positional History Answer Embedding (PosHAE introduced by (Qu, Yang, Qiu, Croft, Zhang & Iyyer 2019)), History Attention mechanism (HAM), and Multi-Task learning (MTL) as illustrated in Figure 3.1. First, PosHAE was conducted by using BERT to encode the history of the answer and its position in the conversation. Second, HAM was the method that applies a single-layer feed-forward neural network to estimate the weight for answering the current question. Last, to predict the answer with another classification task (dialogue act prediction in QuAC) they applied MTL with weighting each prediction model to increase the performance of ConvQA. In addition, HAM has been shown to outperform the baselines including BiDAF++ (Seo et al. 2017), FlowQA (Huang et al. 2019), and BERT PosHAE (Qu, Yang, Qiu, Croft, Zhang & Iyyer 2019), on the ConvQA QuAC (Choi et al. 2018) dataset. As a result, we use HAM as our strongest baseline in Chapter 5, particularly when evaluating the performance of our proposed Multi-Task Learning models in the ConvQA task.

UnifiedQA (Khashabi et al. 2020) is a single pre-trained question answering (QA) model. It was developed to handle different of QA formats, such as extractive span selection (Rajpurkar et al. 2018, 2016), abstractive QA (Kočiskỳ et al. 2018, Reddy et al. 2019), multiple choice (Mihaylov et al. 2018), and Yes/No QA (Clark, Etzioni, Khashabi, Khot, Mishra, Richardson, Sabharwal, Schoenick, Tafjord, Tandon, Bhakthavatsalam, Groeneveld, Guerquin & Schmitz 2020), without relying on format-specific prefixes. UnifiedQA was evaluated across 20 diverse QA datasets spanning 4 different formats. The results demonstrated that UnifiedQA performs on par with or even outperforms dataset-specific expert models trained for individual formats. In particular, the evaluation showed that UnifiedQA performs almost as well as the best single dataset experts and, in some cases. Furthermore, UnifiedQA demonstrated strong generalisation on 12 unseen datasets, highlighting its adaptability and effectiveness in handling diverse QA tasks. As a result, we use UnifiedQA as our strongest baseline in Chapter 8, particularly when evaluating the performance of our proposed Multi-Task Learning models in the ConvQA task.

Conversation History:

Q1: What does Sheldon have to do with the Israeli press?

A1: he proceeded with parallel plans to publish a free daily newspaper to compete with Israeli, a newspaper he had co-founded in 2006

Q2: Was he successful at the paper?

A2: The first edition of the new newspaper, Israel Hayom, was published on July 30, 2007.

Candidate question	Label
Was he the owner of the paper?	**Valid**
Did he make changes at the paper?	**Invalid**
Did they put out and records between 2006-2008?	**Invalid**

Figure 3.2: An example of dialogue and candidate follow-up questions in the follow-up question identification task (Kundu et al. 2020).

3.3 Follow-up Question Identification (FID)

Follow-up question identification is the task of determining if a candidate question is related to previous questions in a conversation (Kundu et al. 2020). It involves detecting references and connections between the candidate follow-up question and the ongoing conversation. In this chapter, we first define the Follow-up Question Identification (FID) task in Section 3.3.1. We then discuss the FID datasets in Section 3.3.2. In Section 3.3.3, we explore how to evaluate the FID task. Finally, we review the existing approaches for the FID task in Section 3.3.4.

3.3.1 FID Task Definition

Following Kundu et al. (2020), we consider the following inputs: a conversation history H_k consisting of a list of k previous questions and ground truth answer pairs, i.e. $H_k = [\langle q, a \rangle]$ and a candidate follow-up question q_c. Given these inputs, the task we address is to predict whether or not the candidate follow-up question q_c is a valid follow-up question.

Figure 3.2 exemplifies the follow-up question identification task, showing a history of length $k = 2$ with the corresponding questions and answers. In particular, as this task is a binary classification task, the aim of a follow-up question identification approach is to classify a question q_c as a *valid* follow-up question or as *invalid*.

3.3.2 FID Dataset

The LIF dataset (Kundu et al. 2020) is a dataset for learning to identify follow-up questions in ongoing conversations. It has been developed using the QuAC (Choi et al. 2018) dataset, which assigns each question one of three categories: should ask, could ask, or should not ask a

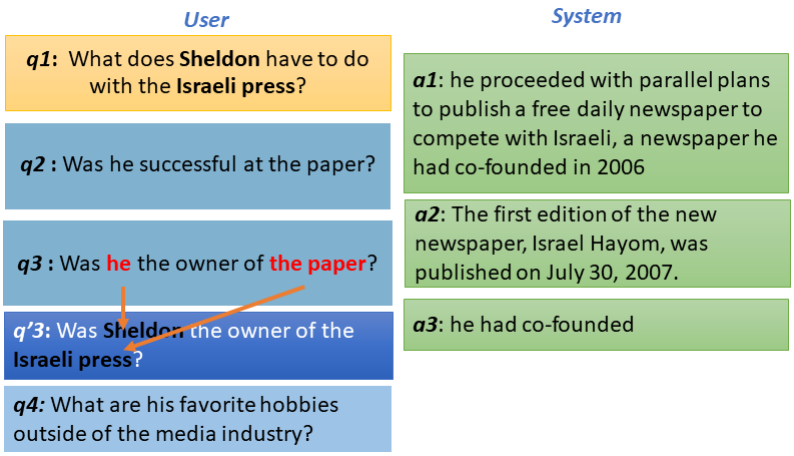


Figure 3.3: An illustrative example of dialogue (Elgohary et al. 2019, Kundu et al. 2020).

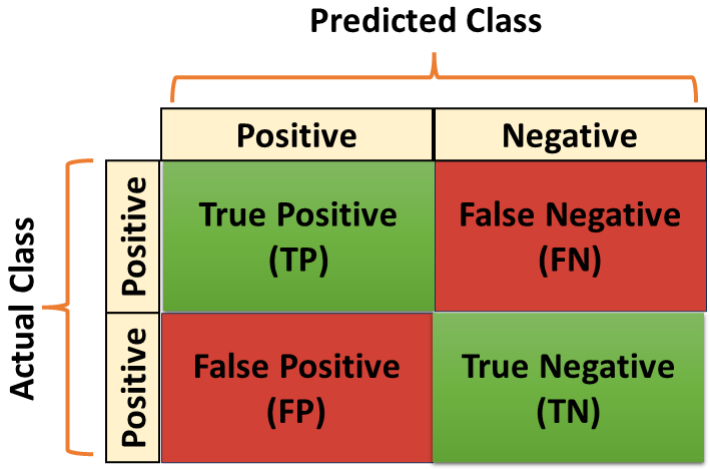


Figure 3.4: A binary confusion matrix illustrating systematic and traditional notations. The green and red colours represent correct rates/counts and incorrect rates/counts in the confusion matrix, respectively. Figure adapted from (Powers 2020).

follow-up question. In Figure 3.3, a follow-up question (q_3) is classified as *valid* if it can be linked to the previous conversation (q_1, a_1, q_2, a_2), else it is classified as *invalid* (e.g., q_4). The *invalid* instances of the LIF dataset were constructed using the should ask follow-up question instances. The LIF dataset contains 126,632 instances for training, 5,861 instances for development, and 5,992 instances for Test-I, which includes candidates from both other conversations and the same conversation. Additionally, there are 5,247 instances for Test-II, which includes candidates from other conversations only, and 2,685 instances for Test-III, which includes candidates from the same conversation only.

3.3.3 Evaluation of FID Systems

The evaluation of Follow-up Question Identification (FID) systems assesses their capability to accurately identify whether a given question is a follow-up to a previous query within a

conversation. This evaluation aims to measure the system's performance in maintaining the conversational context and identifying semantic connections between questions. As introduced in Section 2.7, Precision and Recall are two fundamental metrics used to evaluate the performance of a retrieval system. In this section, we recall these two metrics, as well as the F1-measure and Macro F1, for assessing a system in the classification task rather than retrieval task. As presented in Figure 3.4, True Positive (TP) refers to the number of cases that are correctly predicted as positive, while False Positive (FP) refers to the number of cases that are incorrectly predicted as positive. True Negative (TN) refers to the number of cases that are correctly predicted as negative, while False Negative (FN) refers to the number of cases that are incorrectly predicted as negative. The details of these metrics are as follows:

1. **Precision** is a measure of the accuracy of a system's positive predictions. It is defined as the proportion of true positive (TP) cases (i.e., cases that are correctly predicted as positive) out of all positive predictions ($TP+FP$) made by the system (Powers 2020). Precision is defined as follows:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{3.2}$$

2. **Recall** is a measure of the completeness of a system's positive predictions. It is defined as the proportion of true positive cases (i.e., cases that are correctly predicted as positive) out of all actual positive cases (Powers 2020). Recall is defined as follows:

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3.3}$$

   For example, if a model is predicting whether or not a candidate follow-up question is a valid follow-up question within a conversation dialogue, precision measures the percentage of the candidate follow-up questions that the model correctly identifies valid if the question is actually valid. On the other hand, Recall measures the percentage of the candidate follow-up questions that the model correctly identifies as valid out of all the actually valid follow-up questions in the evaluation set (Kundu et al. 2020). Moreover, a model with a high precision score is likely to make accurate positive predictions. However, by being cautious and making fewer positive predictions, it might miss some actual positives, resulting in a lower Recall. On the other hand, a model with high recall correctly identifies most positives. However, predicting too many instances as positive can create many false positives, which lowers Precision. To achieve a balanced assessment of a model's performance, Precision and Recall are often combined into a single measure, such as F1, to get a more comprehensive understanding of a model's performance.

3. **F1** (van Rijsbergen 1979) is a measure of a system's overall performance that combines both Precision and Recall. It is defined as the harmonic mean of precision and recall, and ranges from 0 to 1, with higher values indicating better performance. F1 is often used

in natural language processing to evaluate the effectiveness of a system in identifying relevant information and is particularly useful when the classes are imbalanced or when both Precision and Recall are important (Powers 2020). The F1-measure is defined as follows:

$$\text{Precision} = \frac{2 \cdot TP}{2 \cdot (TP + FP + FN)} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3.4}$$

4. **Macro F1** (Pillai et al. 2017) is a type of F measure that is computed on a set of instances in multi-label problems. It is defined as the average of the single-label F1 measures computed for each label and gives the same weight to each label. The macro F1-measure is defined as follows:

$$\text{macro F1} = \frac{1}{n} \cdot \sum_{i=1}^{m} F1_i \tag{3.5}$$

where m is the number of classes and $F1_i$ is the F1 score for class i.

In this thesis, we use Precision, Recall, F1 score, and macro F1 as evaluation metrics for the follow-up question identification task in Chapter 6. These metrics are used to assess the performance of the model in terms of its accuracy, completeness, and balance. In other words, the marco F1 score is a variation of the F1 score that is calculated by averaging the F1 scores for each class, regardless of the class imbalance (Kundu et al. 2020).

3.3.4 Approaches for FID

There have been several approaches to address the task of FID in the literature. A number of rule-based approaches (Bertomeu et al. 2006, Kirschner & Bernardi 2007) have been proposed in the literature to address the follow-up question identification task. Such approaches deploy rules to identify if the candidate question contains a reference to previous questions in the conversation, e.g. through a definite description (Kirschner & Bernardi 2007), an ellipsis (Bertomeu et al. 2006), or an anaphoric pronoun (Bertomeu et al. 2006) that links the candidate follow-up question to the previous conversation. Instead of using a rule-based approach as in (Bertomeu et al. 2006, Kirschner & Bernardi 2007), Kirschner & Bernardi (2009) proposed statistical machine learning models, namely Logistics Regression models, where the TF-IDF values of the terms in the questions of the dialogue are used as the models' features for detecting the follow-up question. More recently, Kundu et al. (2020) presented a three-way attention pooling network to identify whether the follow-up question is related to the conversation history with the user. As illustrated in Figure 3.5, this network takes as input three sequences of text: the conversation history, the associated passage, and a candidate follow-up question. The network then uses an attention mechanism to learn the interactions between these three inputs of text. The attention mechanism allows the network to capture both topic continuity and topic shift, which are important factors in determining the suitability of a follow-up question. Unlike the rule-based methods in (Bertomeu

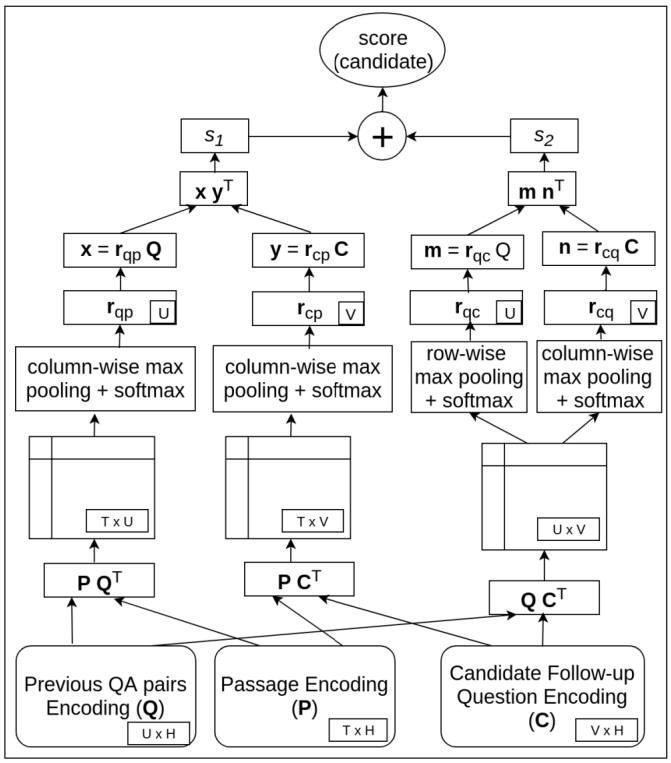


Figure 3.5: Architecture of the three-way attentive pooling network. Figure taken from (Kundu et al. 2020).

et al. 2006, Kirschner & Bernardi 2007), Kundu et al. (2020)'s approach can also make use of the associated answer passage. The three-way attentive pooling network has been by Kundu et al. (2020) was shown in Kundu et al. (2020) to outperform the rule-based methods, a logistic regression model using the TF-IDF values of the terms not only in the questions of the dialogue like (Kirschner & Bernardi 2009), but also the terms in the passage. The three-way attentive pooling approach has also been shown to outperform neural network-based models such as BiLSTM, CNN, and BERT (as introduced in Chapter 2, Section 2.3.2). In Chapter 6, we use both the three-way attentive pooling network and BERT as our baselines since the rule-based and statistical machine learning models have been shown by Kundu et al. (2020) to be much less effective for this task.

3.4 Conversational Question Rewriting (QR)

The task of conversational question rewriting aims to transform a concise question in a conversational context into a fully specified and context-independent query that can be effectively processed by an existing information retrieval (IR) system (Mele et al. 2021).

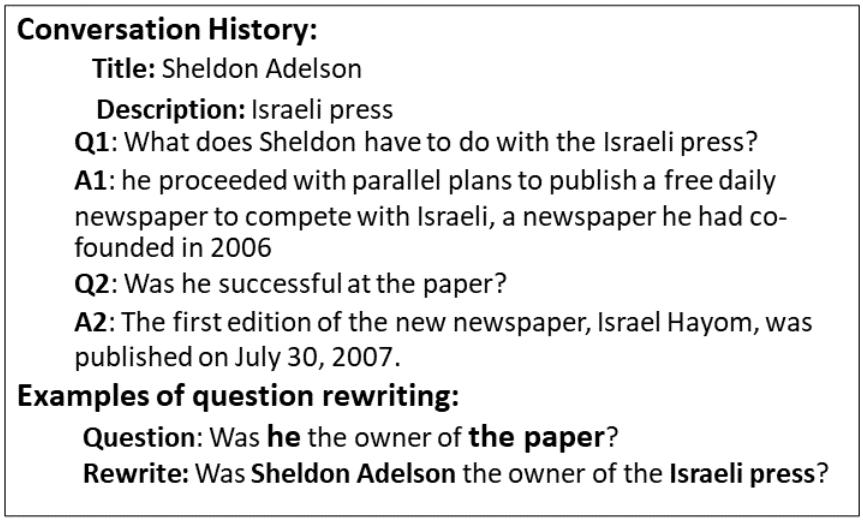


Figure 3.6: An example of dialogue from the conversational question rewriting task (Elgohary et al. 2019).

3.4.1 QR Task Definition

Following Elgohary et al. (2019), given a conversation history H_k consisting of a list of k questions and a list of ground truth answer pairs, i.e $H_k = [\langle q, a \rangle]$, the task is to generate a rewrite q'_m for the next question q_m based on H_k. Because q_m is part of the conversation, its meaning frequently includes references to parts of H_k. A valid q'_m should be self-contained: i.e. a correct answer to q'_m without the history H_k is a correct answer to q_m with the history H_k.

Figure 3.6 exemplifies the conversational question rewriting task, showing a history of length k=2 with the corresponding questions and answers. The question q_m omits the title of the article and the first question (replacing the pronoun "he" with Sheldon Adelson and replacing "the paper" with Israeli press). Hence, to address this task, the system needs to resolve any omission by using history H_k.

3.4.2 QR Dataset

The CANARD dataset (Elgohary et al. 2019) is derived from the QUAC (Question Answering in Context) dataset (see details in Section 3.2.2). The data collection process involved eliciting paraphrases from human crowd-workers to make previously conversational questions unambiguously answerable. The main characteristic of CANARD is that it provides a pair-wise mapping between ambiguous and context-enriched questions, which can be used to train models for conversational question rewriting. The CANARD dataset contains 40,527 questions and their corresponding context-independent rewrites, covering 65 topics from the QUAC dataset. For example, in Figure 3.3, the question q'_3 "Was Sheldon the owner of the Israeli press?" is a rewrite of q_3 "Was he the owner of the paper?", based on the conversation history (q_1, a_1, q_2, a_2).

As mentioned in Section 1.4, our intuition is to improve the system's response accuracy and relevance by combining follow-up question identification and conversational question rewriting tasks. In Chapter 6, we conduct experiments using the LIF (see Section 3.3.2) and CANARD (see Section 3.4.2) datasets, which are recent adaptations of the well-known QuAC Conversational QA dataset (Choi et al. 2018).

3.4.3 Evaluation of QR Approaches

The system's effectiveness is typically evaluated using the BLEU (Papineni et al. 2002) and ROUGE (Lin 2004) scores. BLEU measures the closeness between the system-generated rewrite and the actual rewrite in terms of n-gram precision. On the other hand, ROUGE provides a recall-based measure, providing insight into how much of the actual rewrite is captured in the system-generated rewrite. The details of these metrics are described below:

1. **BLEU (Bilingual Evaluation Understudy) score** (Papineni et al. 2002), primarily used to assess machine translation quality, is also valuable for evaluating responses in conversational question answering (Elgohary et al. 2019). This metric compares the system's output to one or more reference responses, gauging the quality based on similarity. Scores range from 0 to 1, where 1 represents a perfect match with the reference. This score is computed by considering n-gram overlaps, an n-gram being a sequence of 'n' words, between the system's output and the reference response(s) for various 'n' values (usually ranging from 1 to 4). The BLEU score incorporates both precision, the proportion of words in the system's output that match the reference, and brevity, penalising outputs that are excessively short compared to the reference. While the BLEU score is widely used for evaluating machine translation system, it does have limitations; for example, it does not account for the semantic meaning of the generated text (Papineni et al. 2002).

2. **ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score** (Lin 2004) is a set of measures used to evaluate the quality of automatic summarisation and machine translation outputs. The measures are based on comparing the n-gram overlap between the generated summary and the reference summary. ROUGE has four different measures: ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S. ROUGE-N measures the n-gram overlap between the generated summary and the reference summary, while ROUGE-L measures the longest common subsequence (LCS) between the two summaries. ROUGE-W is a weighted version of LCS that takes into account the length of the LCS, and ROUGE-S is a skip-bigram measure that counts the number of skip-bigrams that appear in both summaries. The ROUGE scores range from 0 to 1, with a higher score indicating a better match between the generated summary and the reference summary. ROUGE has been widely used in the evaluation of summarisation and machine translation systems, and its effectiveness has been demonstrated in large-scale evaluations (Lin 2004). ROUGE can

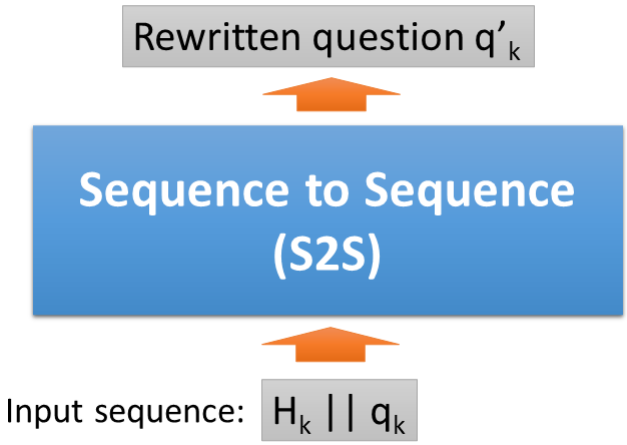


Figure 3.7: Architecture of the sequence-to-sequence model for the conversational question rewriting task. Figure inspired by (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*).

also be used to evaluate the quality of conversational question rewriting systems (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Mele et al. 2021, Ren et al. 2018, Vakulenko, Longpre, Tu & Anantha 2021, Vakulenko, Voskarides, Tu & Longpre 2021, Voskarides et al. 2020, Yu et al. 2020). In this task, the goal is to rewrite a question in a way that makes it easier for a retrieval system to retrieve relevant passages. ROUGE can be used to measure the lexical overlap between the rewritten question and the original question. A higher ROUGE score indicates that the rewritten question is more similar to the original question, which suggests that existing information retrieval systems can adequately handle it. This is because the rewritten question that is more similar to the original question is more likely to contain the same keywords or phrases as the original question. This means that it is more likely to match the keywords or phrases in the documents that are relevant to the original question.

In this thesis, we use the BLEU and ROUGE scores as evaluation metrics for assessing performance in conversational question rewriting in Chapter 7.

3.4.4 Approaches for QR

Several approaches have been proposed to address the linguistic characteristics of human conversation, such as anaphora and ellipsis (see Section 3.3.4). Ren et al. (2018) introduced sequence-to-sequence models like LSTM (Hochreiter & Schmidhuber 1997) and GRU (Cho et al. 2014) for context-aware conversational query rewriting. Yu et al. (2020) proposed methods to generate weak supervision data from large sets of ad-hoc search sessions using rules and self-supervised learning. They fine-tuned the GPT-2 (Radford et al. 2019) model with this data and demonstrated its superior performance over the state-of-the-art in the TREC CAsT 2019 (Dalton et al. 2019) track. Another GPT-2 based model, Transformer++ (Vakulenko, Longpre, Tu & Anantha 2021), has been trained on the CANARD dataset (see Section 3.4.2) to rewrite the

current question considering the previous five conversation turns. Vakulenko, Voskarides, Tu & Longpre (2021) conducted a comparison study of question-rewriting approaches using the TREC CAsT 2019 and 2020 datasets (Dalton et al. 2019, 2020). They evaluated GPT-2-based models (Transformer++ and self-supervised learning) as well as a related-term classification method called QuReTeC against original user questions and human-rewritten questions. The results showed that using Transformer++ to rewrite the current question with related terms predicted by QuReTeC outperformed the existing state-of-the-art methods. Recently, Lin, Yang, Nogueira, Tsai, Wang & Lin (2020*a*) proposed using neural sequence-to-sequence (S2S) models for the conversational question reformulation task, such as T5 (see Section 2.3.1), GPT-2, LSTM, BERT (see Section 2.3.2), and UniLM (Dong et al. 2019). The S2S model took the original question q_k and its context H_k as input and generated the rewritten question $q\prime_k$ as output, as shown in Figure 3.7. In order to optimise the parameters within the S2S models, Lin, Yang, Nogueira, Tsai, Wang & Lin (2020*a*) employed supervised learning for training the model to generate predicted tokens ($q\prime_k$) by utilising the ground truth output ($\hat{q}_k$) tokens. Among these S2S models, the T5 model demonstrated a superior performance compared to neural network-based models such as LSTM, GPT-2, BERT, and UniLM on the CANARD (Elgohary et al. 2019) and CAsT 2019 (Dalton et al. 2019) datasets. However, while previous work, such as (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Mele et al. 2021, Ren et al. 2018, Vakulenko, Longpre, Tu & Anantha 2021, Vakulenko, Voskarides, Tu & Longpre 2021, Voskarides et al. 2020, Yu et al. 2020), focused on conversational question rewriting; however, they did not address the task of follow-up question identification. Recall from Section 1.3 that our intuition is that by combining follow-up question identification and conversational question rewriting, the system's response accuracy and relevance can be enhanced. Indeed, we argue that by identifying connections between the user's questions, addressing ambiguities, and leveraging the conversation's context, the system can refine its understanding of the user intent and can provide more precise and relevant responses. In Chapter 6, our text generation models leverage the Multi-Task Learning of the conversational question rewriting and classification tasks to identify whether a question is a follow-up to the previous question and, accordingly, reformulate a question using the dialogue context. To the best of our knowledge, no prior work has inherently combined both tasks to more effectively address ambiguity in conversational questions.

3.5 Clarification Need Classification (CNC)

The Clarification Need Classification (CNC) task (Aliannejadi et al. 2021) aims to address the challenge of identifying when a user's query is ambiguous or lacks clarity, necessitating further clarification to generate accurate and contextually relevant answers.

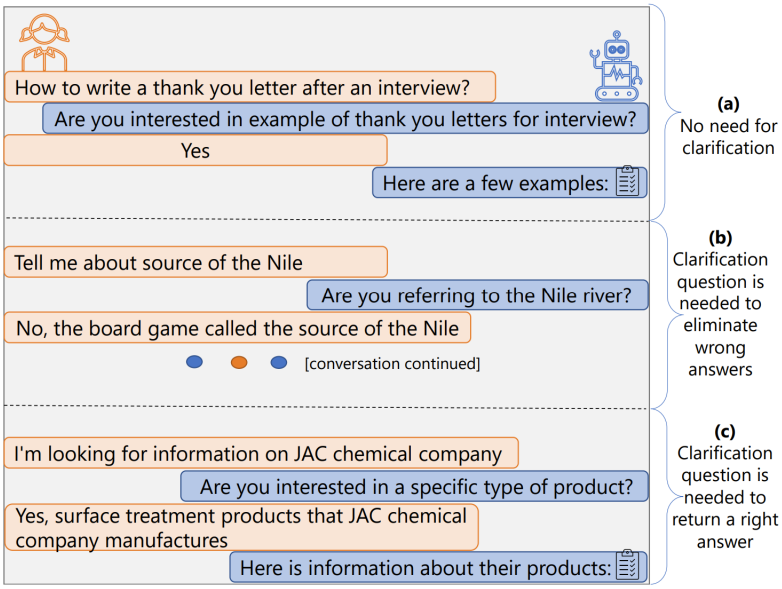


Figure 3.8: Examples of the need for clarification questions: (a) depicts a clear user question that requires no further clarification; (b) and (c) present scenarios where the user's questions are ambiguous, necessitating a clarifying question from the system. Figure taken from (Aliannejadi et al. 2021).

3.5.1 CNC Task Definition

Following Aliannejadi et al. (2021) and Owoicho et al. (2022), the task of clarification need classification is defined as follows: given a current utterance u_k, and a conversation history H_k – consisting of a list of $k-1$ utterances and the corresponding response text pairs, i.e. $H_k = [\langle u_i, r_i \rangle]_{i=1}^{k-1}$ – the task is to determine whether the system should ask a clarifying question in response to user utterance u_k. An example of the need for clarification questions is shown in Figure 3.8. The user question "How to write a thank you letter after an interview?" (Figure 3.8) is a clear question, which requires no further clarification. The system can directly answer this question without asking the user any clarifying questions. In contrast, the user questions "Tell me about the source of the Nile" and "I'm looking for information on JAC chemical company" are ambiguous, as there are many sources of the Nile river and many JAC chemical companies. The system would hence need to ask the user to clarify their question before providing an answer.

3.5.2 Dataset for CNC

The ClariQ dataset (Aliannejadi et al. 2020*a*, 2021) provides a benchmark for the task of Clarification Need Classification. The task is to predict the necessity of asking clarifying questions given a user request, and the ClariQ dataset includes a set of conversational user requests and a set of questions (i.e., question bank) that contains all collected questions on all the topics. ClariQ introduced a module called "Understanding User Request", which takes a user request as input and returns a score from 1 (no need for clarifying questions) to 4 (cannot provide

Table 3.2: Distribution of Instances in the ClariQ Dataset for Clarification Need Classification.

Sets	Labels			
	1	2	3	4
Train	676	3546	3474	1480
Development	49	854	914	496
Test	317	2478	1465	239

Rank	Creator	Model Name	Precision	Recall	F1
1	TAL ML	Roberta+++	**59.81**	**65.57**	**60.70**
2	Cactusjam	Roberta+Stats	59.63	59.02	54.16
3	TAL ML	Roberta++	52.90	55.74	52.53

Table 3.3: Clarification Need Prediction on the ConvAI3 leaderboard.

any answers without user clarification) indicating the necessity of asking clarifying questions. In Chapter 7, we use the ClariQ dataset for training and evaluating the performance of our proposed system in Clarification Need Classification.

In the ClariQ dataset, as summarised in Table 3.2, the instances are distributed across different sets, each labelled with values 1, 2, 3, or 4. The ClariQ dataset is a comprehensive and well-curated resource for clarification need classification, providing a wide range of examples that can be used to train and evaluate models.

3.5.3 Evaluation CNC Approaches

The evaluation of Clarification Need Classification (CNC) systems involves assessing their ability to accurately determine whether a given question requires additional clarification or context. We use typical classification metrics such as precision, recall, and F1, in line with ClariQ (Aliannejadi et al. 2020*a*). These metrics have been previously described in Section 3.3.3.

3.5.4 Approaches for CNC

In this section, we describe approaches, which will be used as the baselines for Chapter 6. ConvAI3 (Aliannejadi et al. 2020*b*) is a Conversational AI challenge series that includes the ClariQ challenge, which aims to generate clarifying questions for open-domain dialogue systems. The ClariQ challenge involves the task of clarification need classification, which is the process of identifying the level of clarification needed for a given user request. The collected dataset for the challenge includes an initial user request in conversational form, a set of possible clarifying questions, and a user answer for each question. The label for each user request reflects the level of clarification needed, ranging from 1 to 4. The participants are required to return a score from 1 to 4 indicating the necessity of asking clarifying questions for a given user request.

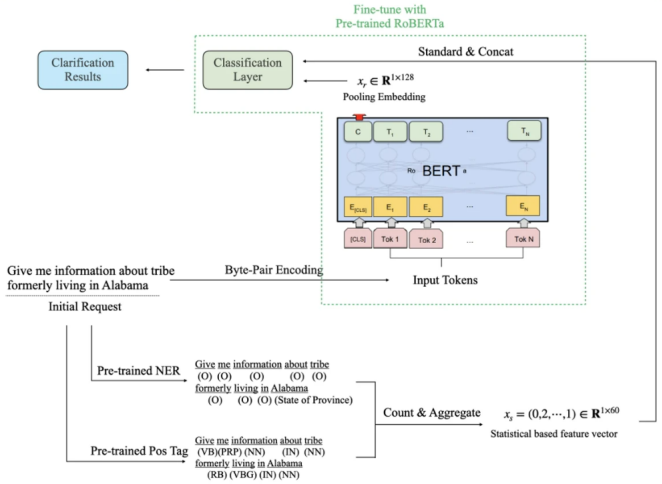


Figure 3.9: A system overview of Roberta+++ by TAL ML. Figure taken from (Li et al. 2020).

Table 3.3 presents the results of the participation approaches on the ClariQ (Aliannejadi et al. 2021) test set. Roberta+++ by TAL ML (Li et al. 2020) has the highest precision, recall, and F1 scores (59.81, 65.57, and 60.70, respectively).

Roberta+++ incorporates both the user utterance and the user feedback into its clarification need prediction, as depicted in Figure 3.9. Moreover, it uses Named Entity Recognition (NER) and Part of Speech Tagging (PoS Tagging) by employing a count and aggregation process, followed by connecting with the Roberta model's representation as inputs to the classification layer (Li et al. 2020). Due to the best performance of Roberta+++, we have chosen to use it as the baseline for clarification need classification in Chapter 7.

3.6 Asking Clarifying Questions

The task of Asking Clarifying Questions is a typical component of Open-Retrieval Conversational Question Answering (ORConvQA) systems. This task addresses the challenge of generating or selecting contextually appropriate questions that seek clarification from the user when their query is ambiguous or lacks clarity, enabling the system to provide accurate and relevant answers.

3.6.1 Asking Clarifying Questions Task Definition

Following Aliannejadi et al. (2021) and Owoicho et al. (2022), the task of asking clarifying questions is defined as follows: given a current utterance u_k, and a conversation history H_k – consisting of a list of $k-1$ utterances and the corresponding response text pairs, i.e. $H_k = [\langle u_i, r_i \rangle]_{i=1}^{k-1}$ – the task is to predict (generate or select) a clarifying question c_k that clarifies the current utterance u_k, and receive feedback, u_{k_2}, from the user for the clarifying question c_k. An example of the asking clarifying question task is shown in Figure 3.10. It consists of

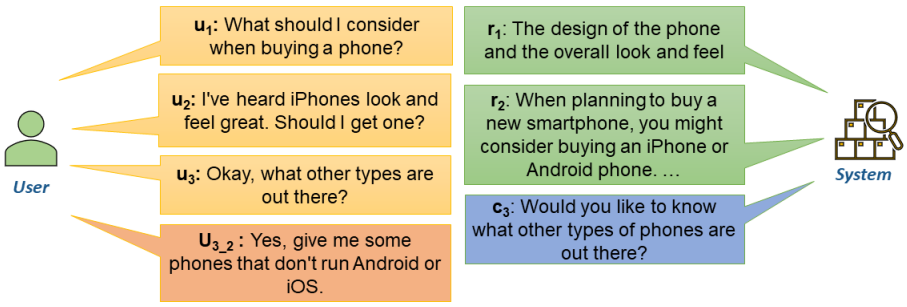


Figure 3.10: An example of dialogue from the conversational question rewriting task (Elgohary et al. 2019).

a conversation history of length $k = 3$ (two previous pairs of utterances and response texts). In order to predict a clarifying question c_3, the asking clarifying questions system needs to either select a list of candidate questions in a pool of questions C or to generate new questions by leveraging the conversation history H_k to specify the user's intent and correctly predict a relevant clarifying question c_3 that is specific to the user's utterance u_3. The user's response to the clarifying question, u_{3_2}, can then be used to further inform and improve the system's performance.

3.6.2 Asking Clarifying Questions Datasets

In the following, we described two commonly used datasets for the task of asking clarifying questions.

The Qulac dataset (Aliannejadi et al. 2019) is a collection of clarifying questions in an information retrieval (IR) setting. Its main purpose is to help conversational systems better understand the users' information needs by proactively asking clarifying questions. The dataset was collected through crowdsourcing in four steps: defining topics and their corresponding facets, collecting candidate clarifying questions for each query, assessing the relevance of the questions to each facet, and collecting new questions for those facets that require more specific questions. The dataset consists of 198 topics, each coupled with a facet, resulting in 141 faceted topics and 57 ambiguous topics. There are 762 facets and 2,639 questions in total, with an average of 3.85 facets per topic and 9.49 terms per question. The dataset also includes relevance judgments at the facet level, borrowed from the TREC Web track (Clarke et al. 2011).

The ClariQ dataset (Aliannejadi et al. 2020*a*, 2021) is a dataset, which is dedicated to the problem of asking clarifying questions in open-domain dialogue systems. Unlike the Qulac dataset, which focused on single-turn conversations and contained only a limited number of topics, the ClariQ dataset includes both single- and multi-turn conversations and covers a much wider range of approximately 300 various topics. The main purpose of the ClariQ dataset is to provide a benchmark for evaluating the quality of clarifying questions in open-domain dialogues and to study when and which clarifying questions should be asked given the current context of the

conversation. The dataset is also intended to be used as a resource for training and testing various conversational agents and neural models. To collect the ClariQ dataset, Aliannejadi et al. (2020*a*) used a combination of crowdsourcing and existing datasets. They first converted topics into conversational requests using the Qulac dataset, and then markedly extended it by crowdsourcing more data through Human Intelligence Task (HIT) on Amazon Mechanical Turk. They asked the workers to imagine themselves acting as a conversational agent where an imaginary user had asked them about a topic. The ClariQ dataset contains approximately 15,000 single-turn conversations and 1.5 million multi-turn conversations, making it one of the largest datasets of its kind. The main characteristic of the dataset is that each inquiry to the system should be in conversational form, and the need for clarification should be predetermined as a label for each inquiry in the collection. In addition, each clarifying question should be reasonable, coherent with the inquiry, and address multiple facets of every ambiguous request.

In this thesis, we chose not to use the Qulac dataset because its questions are ac-hoc keyword queries, and we focus on conversational-like queries. Hence, in Chapter 7, where we propose a Multi-Task Learning of Clarification Need Classification and the generation for asking clarifying question model, we train our proposed model using the training and development sets of ClariQ. This is necessary as CAsT 2022 (see details in 3.7.2) only provides an evaluation set for clarifying questions. Note that we do not use the ClariQ dataset to evaluate our proposed model because it does not support the generation-based asking of clarifying questions, which aligns with our research objectives.

3.6.3 Evaluation of Asking Clarifying Questions Approaches

To evaluate the effectiveness of asking clarifying questions, we follow CAsT 2022 (Owoicho et al. 2022) and apply P@1, and assess performance based on three criteria. These criteria include Relevance, which measures whether the question logically follows from previous utterances in the conversation; Novelty, which assesses whether the question adds new information to the conversation; and Diversity, which considers the number of options provided by the question. We have previously provided an explanation of the Precision metric in Section 2.7.

3.6.4 Approaches for Asking Clarifying Questions

Recall from Section 1.2 that to address the ambiguity of conversational questions in OR-ConvQA, previous works (Aliannejadi et al. 2020*a*, 2021, Owoicho et al. 2022) have proposed the use of asking clarifying questions. Asking clarifying questions is an approach employed in mixed-initiative conversational search systems to enhance the search experience for users (Keyvan & Huang 2022, Zamani et al. 2022). These systems combine both machine and human initiative to better understand the user's information needs and provide more accurate search results. By asking clarifying questions, the system aims to elicit further details from the users, allowing

for a refined interpretation of their intents and improving the system's understanding of their information needs. In addition, users have the opportunity to provide additional information and feedback to the system, contributing to the improvement of the search results (Keyvan & Huang 2022, Zamani et al. 2022). In general, asking clarifying questions in a mixed-initiative approach creates a more natural and interactive search experience, resembling a human-like conversation (Krasakis et al. 2020). Research on asking clarifying questions in conversational search has explored both generation and selection approaches.

Generation-based approaches involve creating clarifying questions tailored to the user's query and context. Template-based slot filling methods have been proposed, such as the approach by Coden et al. (2015), which defines question templates like "Did you mean _ or _?" However, these templates may not be applicable to all queries. To address this, Generating Clarifying Questions (Zamani et al. 2020) identified more general clarification question templates, including "What would you like to know about _?" from search logs. They also proposed weakly-supervised and reinforcement learning models for generating clarifying questions based on their template-based slot filling approach. TG-ClariQ (Wang & Li 2021) introduced a model that selects a template question from a set of candidates and fills it in with words from a slot vocabulary. Furthermore, Owoicho et al. (2022) fine-tuned text generation models, specifically T5 (see Section 2.3.1) and GPT-3 (see Section 2.3.3), to generate clarifying questions. However, generating clarifying questions using a generation-based approach requires a diverse and high-quality dataset, which can be challenging to obtain. Evaluating the quality of generated questions can also be difficult due to the reliance on human annotations or online experimentation.

Instead, selection-based approaches involve choosing pre-determined questions from a pool of options based on the user's query. These questions are typically created by experts and pre-approved to ensure quality (Aliannejadi et al. 2021, 2019). Prior approaches (Aliannejadi et al. 2021, 2019, Ou & Lin 2020, Owoicho et al. 2022, Rao & Daumé III 2018) have applied classical information retrieval techniques such as BM25 (see Section 2.2) to retrieve and rank pre-determined questions. For example, Rao & Daumé III (2018) proposed a neural network model that leverages the expected value of perfect information to rank clarification questions in three StackExchange domains. NTES-ALONG (Ou & Lin 2020) retrieves candidate clarifying questions using BM25 and re-ranks them using a fine-tuned ELECTRA (Clark, Luong, Le & Manning 2020) model to estimate the relation between a query and a clarifying question. Owoicho et al. (2022) described a system that generates a candidate set of questions from a question bank using sentence similarity and then applies a fine-tuned BERT model for pairwise ranking. However, pre-determined questions may not cover a wide range of queries and intents, limiting the system's robustness and versatility.

Instead, in this thesis (Chapter 7), we take advantage of both the generation and selection. In doing so, our hybrid approach is able to generate a better set of questions and ensure that the selected question is relevant to the user's query. The goal of our proposed system is to

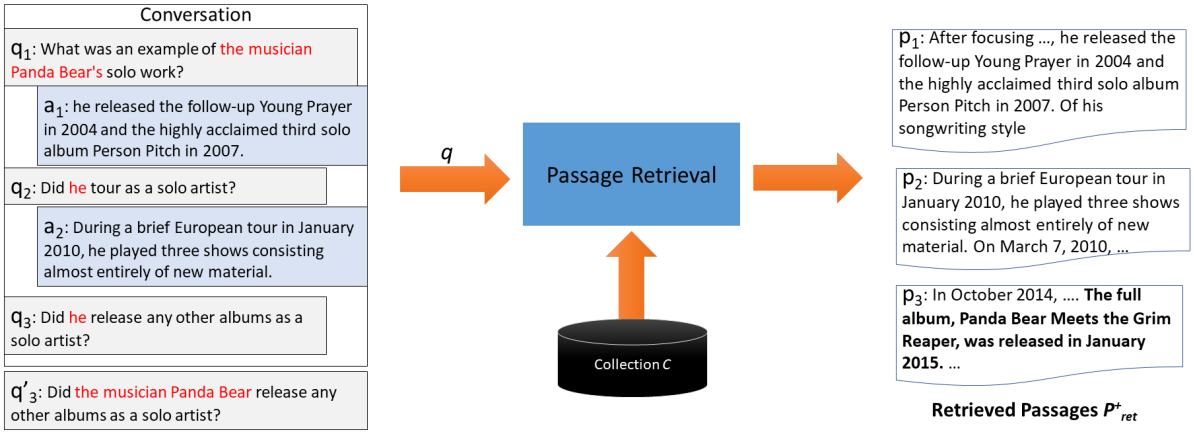


Figure 3.11: An example of the passage retrieval task for ORConvQA

improve the effectiveness of the task of asking clarifying questions. To the best of our knowledge, no previous research has combined both approaches to more effectively address unclear or ambiguous questions by asking clarifying questions.

3.7 Passage Retrieval

Conversational Passage Retrieval is a fundamental component of ORConvQA systems. This task tackles the challenge of effectively searching through a large collection of documents to identify relevant passages that contain information related to the ongoing conversation.

3.7.1 Passage Retrieval Task Definition

Following Croft et al. (2010), given a query q and a text collection C, the task is to identify a list of n relevant passages $P^{+}_{ret} = [p_1, p_2, ..., p_n]$ corresponding to the query q. These passages then serve as input to later more complex models. The query q can be either the ongoing user's question q_k along with its conversation history H_k, or the reformulated question q'_k.

An example of the passage retrieval task for Open-Retrieval Conversational Question Answering is shown in Figure 3.11, showing a history of length $k = 2$ with the corresponding questions and answers. In this task, the system takes the query q as an input to retrieve a list of $n = 3$ relevant passages denoted as $P^{+}_{ret} = [p_1, p_2, p_3]$ from the passage corpus C. The query q can be either the ongoing user's question q_3 along with its conversation history $H_2 = [\langle q_1, a_1 \rangle, \langle q_2, a_2 \rangle]$, or the reformulated question q'_3.

3.7.2 Passage Retrieval Datasets

Research in the field of Conversational Search has advanced over the years, with the creation of a series of reusable datasets by the TREC Conversational Assistance Track (CAsT). Each year,

the TREC CAsT datasets (2019, 2020, 2021, and 2022) have introduced fresh complexities and challenges to better mimic real-world scenarios and improve the performance of conversational search systems. Moreover, to address the challenges of ORConvQA, datasets, such as OR-QuAC (Qu et al. 2020) and OR-CoQA (Qu et al. 2021), have been introduced. In the following, we provide a detailed description of each dataset:

TREC CAsT 2019 (Dalton et al. 2019) was an initiative to facilitate Conversational Information Seeking (CIS) research and to create a large-scale reusable test collection for conversational search systems. The task required effective response selection that requires understanding a question's context (the dialogue history). The primary focus was placed on a system understanding the users' information needs in a conversational format and finding relevant passages leveraging conversational context. The data collection included pre-determined conversation trajectories (paths) and passage responses. There were 20 topics with 173 turns and 29,571 assessments. The judgments were based on how well the system's response met the information need of the user, with scores ranging from 0 (fails to meet) to 4 (fully meets). One of the main characteristics of TREC CAsT 2019 was the use of conversational context to improve search accuracy. The task required systems to understand the context of the conversation and use it to select the most relevant response. This was a departure from traditional information retrieval systems that rely on keyword matching and do not take into account the conversational context. The goal was to encourage research on conversational search systems that can better serve users in real-world scenarios.

TREC CAsT 2020 (Dalton et al. 2020) focused on conversational search challenges. The goal of the 2020 task was to satisfy a user's complex information need expressed through multi-turn conversational queries/utterances by retrieving and ranking passages from MS MARCO and Wikipedia. The 2020 edition of CAsT had 25 information needs (topics) with an average length of 8.6 utterances, for a total of 216 turns. This is slightly shorter than the 2019 edition, which averaged 9.5 turns. The topics are based on multi-turn information-seeking sessions from a commercial search engine and reflect the organisers' vision of user behaviour for the conversational search systems of the future, while also being grounded in real information needs and current search behaviour. The reference sessions are constructed by filtering raw web search sessions to conversational-alike search sessions, using the procedure to obtain about 20,000 "QA-Gen" sessions as described in Rosset et al. (2020). The 2020 topics have more complexity than the previous year's, with turns requiring previous result context being particularly challenging. In this edition, CAsT aimed to support natural conversations between a person and a search engine to satisfy information needs and support complex information tasks.

TREC CAsT 2021 (Dalton et al. 2021) focused on conversational search techniques. The goal of the task is to satisfy a user's complex information need expressed through multi-turn conversational queries/utterances. In this edition, the passages come from a document corpus by retrieving and ranking documents and passages from MS MARCO (Nguyen et al. 2016),

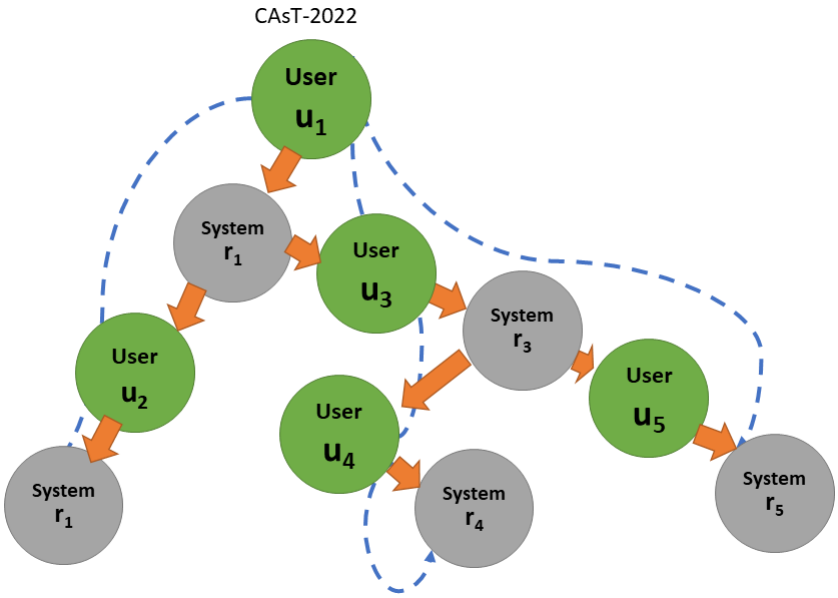


Figure 3.12: Example of a CAsT 2022 dialogue tree with 1 main topic, 3 sub-topics, and 5 user utterances. Figure inspired by (Owoicho et al. 2022).

Wikipedia - the KILT dump, and news from the Washington Post V4 collection. Compared to previous years, CAsT 2021 had some important changes. One of the changes is that every turn has a single manually selected canonical response passage result representing a previous system response. This evolved from CAsT 2020 when only some turns were manually selected and others automatically added from the baselines. CAsT 2021 manual results provide consistency between automatic and manual runs. The canonical results are used more, with higher query dependence on previous system responses. CAsT 2021 has 26 information needs (topics) with an average length of 9.2 utterances, for a total of 239 turns. In comparison, the CAsT 2020 topics are slightly shorter with an average of 8.6 utterances per topic. The topics in 2021 are based on real user needs from information-seeking sessions in Bing. The organisers manually reviewed and filtered sessions to ensure they have meaningful trajectories that are then manually rewritten to make them conversational. The final topics reflect diverse types of exploratory information needs while also being grounded in real information needs that have content available in the target collection. The main characteristic of CAsT 2021 is that it introduced more diverse types of interactions and increased dependence on previous system responses. The turns introduce simple forms of user revealment, reformulation, and explicit feedback if the previous canonical response is not relevant. This makes the task a bit more realistic by having varying types of user interactions.

TREC CAsT 2022 (Owoicho et al. 2022) is the fourth year of the TREC Conversational Assistance Track, which focuses on evaluating Conversational Passage Ranking (ConvPR) for information seeking. CAsT 2022 aims to take conversational search to the next level with new additions and improvements. The topics are more realistic and dynamic, and the evaluation

metrics have been updated to better reflect the conversational nature of the task. In addition to ConvPR, a new sub-task on response generation has been introduced, which involves generating natural language responses to user queries. TREC CAsT 2022 also includes a mixed-initiative (MI) sub-task. This sub-task builds on the main task using the same collection and topics but with the added feature of mixed-initiative responses. Mixed-initiative responses are included in trajectories, which provide the system with a chance to ask the user a question to clarify the information need, ask for feedback, or elicit the task. This new addition aims to make the track more interactive and realistic. Participants have the option to submit MI utterances at every point of the conversation and receive a user response. This represents a first step for the track beyond "user ask, system reply", albeit on predefined fixed trajectories. The mixed initiative is incorporated into the canonical system responses, and the outcome of this sub-task could be used in the main phase. TREC CAsT 2022 has a total of 18 information needs (topics) with an average length of 11.39 user utterances and an average of 2.7 sub-topics. The topics follow a "tree" structure with distinct conversational paths, with a maximum of nine distinct conversational paths and a minimum of one. Each topic starts off with a common query but branches off at various points in the conversation as the topic unfolds. As depicted in Figure 3.12, user utterance u_1 can appear in all sub-topics, including (1) $u_1 : r_1 : u2 : r2$; (2) $u_1 : r_1 : u_3 : r_3 : u_4 : r_4$; and (3) $u_1 : r_1 : u_3 : r_3 : u_4 : r_5$. However, for the purpose of evaluating performance in the conversational search task, u_1 can only be evaluated in sub-topic (1) (Owoicho et al. 2022). The topics are divided into two sets: a training set of 10 topics and a test set of 8 topics. There are a total of 205 user utterances, including vague, ambiguous, or user responses to system questions.

In this thesis, we evaluate our proposed conversational question rewriting model in Chapter 6 using the TREC CAsT 2019 and TREC CAsT 2020 datasets. Chapter 7 uses the TREC CAsT 2021 dataset to train the negative feedback analysis model, which helps analyse the sentiment of user feedback in the task of asking clarifying questions. Furthermore, in our evaluation of the hybrid approach for generating and selecting clarifying questions in a mixed-initiative conversation search, we use the TREC CAsT 2022 dataset.

OR-QuAC: This dataset has been introduced by Qu et al. (2020), adapting the well-known QuAC (Choi et al. 2018) dataset to an open-retrieval setting. This dataset is an aggregation of three existing datasets consisting of (1) the QuAC dataset (see Section 3.2.2), which is an information seeking dataset, (2) the CANARD dataset (see Section 3.4.2), which contains questions that humans have re-written from questions in the QuAC dataset, and (3) the Wikipedia corpus, a large collection of over 11 million passages, which are used as the knowledge source for actually answering a given question. The OR-QuAC dataset has 5,644 dialogues with 40,527 questions, and it is intended to facilitate research on ORConvQA. The main characteristic of the OR-QuAC dataset is that it is self-contained, meaning that the initial questions in a conversation have been replaced with their rewrites from CANARD. This makes the dialogues easier to interpret in an open-retrieval setting. Overall, the OR-QuAC dataset offers a unique set of challenges

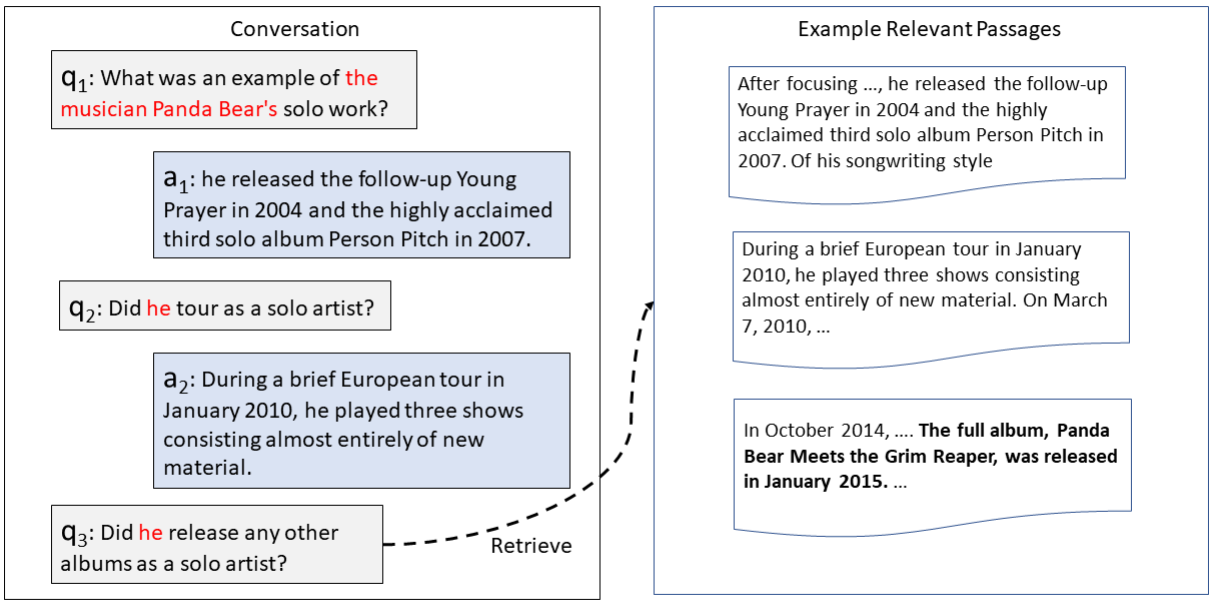


Figure 3.13: An example dialog and relevant passages from the ORConvQA dataset (Qu et al. 2020).

and opportunities for developing functional conversational search systems. An example of the ORConvQA task selected from the OR-QuAC dataset is shown in Figure 3.13, consisting of a relevant passage (shown in bold), and a conversation history of length $k = 2$ (two previous pairs of questions and answers). In order to answer question q_3, the ORConvQA system needs to retrieve a list of relevant passages in C and leverage the conversation history to understand and correctly predict an answer a_3 from all relevant passages in C.

OR-CoQA: Qu et al. (2021) introduced this dataset by aggregating the CoQA (Reddy et al. 2019) dataset with the Wikipedia corpus from the OR-QuAC dataset. In contrast to OR-QuAC, the gold passages for each question are not included in the OR-CoQA dataset. Moreover, unlike OR-QuAC, there are no manually rewritten questions in the OR-CoQA dataset. The dataset includes 1,521 training dialogues, 100 development dialogues, and a test set that is not publicly available. The main characteristic of OR-CoQA is that it offers freeform answers generated by crowdsourcing, which makes the conversations more natural and challenging for conversational question answering systems. In contrast to OR-QuAC, the gold passages for each question are not included in the OR-CoQA dataset. Moreover, unlike in the OR-QuAC dataset, there are no manually rewritten questions in the OR-CoQA dataset. As a result, we do not use OR-CoQA for training our proposed model for addressing the tasks of passage reranking and answer extraction in Chapter 8.

In Chapters 8, to evaluate our proposed model to address the tasks of passage reranking and answer extraction, we selected two datasets: OR-QuAC and OR-CoQA described above. Both of

these datasets are extractive Question Answering (QA) datasets. However, the OR-CoQA dataset can be also considered as a generative question answering dataset because it contains both span and freeform answers. Indeed, in this thesis, we focus on extractive QA only.

Finally, Table 3.4 summarises the used datasets in this thesis providing information on the chapters in which each dataset is used and the specific tasks they are employed to tackle.

Table 3.4: Summary of Datasets for Open-Retrieval Conversational Question Answering

Dataset	Fullname	Conversational	Multi-Turn	Retrieval	Tasks	Chapter	Comment
SQuAD (Rajpurkar et al. 2018, 2016)	Stanford Question Answering Dataset	/	x	x	- Question Answering	-	- Extractive Question Answering
MS-Marco (Nguyen et al. 2016)	Microsoft Machine Reading Comprehension	x	x	/	- Passage/Document retrieval	-	
QuAC (Choi et al. 2018)	Question Answering in Context	/	/	x	- Conversational Question Answering - Follow-up question identification - Yes/No prediction - Unanswerable	5	- Extractive Question Answering
CoQA (Reddy et al. 2019)	A Conversational Question Answering Challenge	/	/	x	- Conversational Question Answering - Yes/No prediction	-	- Extractive Question Answering - Abstractive QA
LIF (Kundu et al. 2020)	Learning to Identify Follow-up Questions	/	/	x	- Follow-up question identification	6	
CANARD (Elgohary et al. 2019)		/	/	x	- Conversational Question Rewriting	6,7,8	
Qulac (Aliannejadi et al. 2019)		x			- Asking Clarifying Question	-	
ClariQ (Aliannejadi et al. 2020*a*, 2021)		/	/	/	- Asking Clarifying Question	7	
CAsT 2019 (Dalton et al. 2019)		/	/	/	- Conversational Question Rewriting - Conversational Search	6	
CAsT 2020 (Dalton et al. 2020)		/	/	/	- Conversational Question Rewriting - Conversational Search	6	
CAsT 2021 (Dalton et al. 2021)		/	/	/	- Conversational Question Rewriting - Conversational Search	7	
CAsT 2022 (Owoicho et al. 2022)		/	/	/	- Conversational Question Rewriting - Conversational Search - Conversational Question Answering - Asking Clarifying Question	7	
OR-QuAC (Qu et al. 2020)		/	/	/	- Conversational Question Rewriting - Question Answering - Passage Retrieval	8	
OR-CoQA (Qu et al. 2021)		/	/	/	- Conversational Question Rewriting - Question Answering - Passage Retrieval	8	- No Qrels
NQ (Kwiatkowski et al. 2019)	Natural Questions	/	x	/	- Document Retrieval - QA	-	
QReCC (Anantha et al. 2021)		/	/	x	- Conversational Question Rewriting	-	CANARD+CAsT+NQ
INSCIT (Wu et al. 2022)	Information-Seeking Conversations with Mixed-Initiative Interactions	/	/	/	- Conversational Question Answering - Conversational Question Rewriting - Conversational Search - Asking Clarifying Question	-	

3.7.3 Evaluation of Passage Retrieval Approaches

The evaluation of Conversational Passage Retrieval aims to assess the effectiveness of systems in accurately retrieving relevant passages from a large document collection in response to user queries. We use MAP, MRR, Recall, and NDCG, as previously discussed in Section 2.7, for evaluating this task.

- **Mean Average Precision (MAP):** This metric is calculated by averaging the precisions at different ranks. Precision is the fraction of relevant passages that are retrieved in the top k results.
- **Mean Reciprocal Rank (MRR):** This metric is calculated by averaging the reciprocal ranks of the relevant passages. The reciprocal rank of a passage is 1 divided by its rank.
- **Recall:** This metric measures the fraction of relevant passages that are retrieved.
- **Normalised Discounted Cumulative Gain (NDCG):** This metric is similar to MAP, but it gives more weight to the top results. NDCG is calculated by normalising the discounted cumulative gain (DCG) by the ideal DCG.

In this thesis, Chapter 6, which proposes a model to address the tasks of follow-up question identification and conversational question rewriting, uses MAP, MRR, NDCG and Recall@1000 as metrics. In Chapter 7, which proposes a model to address the tasks of clarification need classification and asking clarifying questions, we adopt the NDCCG@3, MAP@1000, MRR@1000, and Recall@1000 metrics, as used in CAsT 2022 (Owoicho et al. 2022). In Chapters 8, which proposes a model to address the tasks of passage reranking and answer extraction, following (Qu et al. 2020, Yu et al. 2021), we use MAP@10, MRR@5 and Recall@5. The notation "@k" signifies that the evaluation metric is computed considering the top k ranked results. For instance, "MAP@10" signifies that the Mean Average Precision is calculated based on the top 10 retrieved items, while "MRR@5" indicates that the Mean Reciprocal Rank is determined using the reciprocal rank of the top 5 relevant passages.

3.7.4 Approaches for Passage Retrieval

Unlike dense retrieval models, discussed in Section 2.4, such as ANCE, ColBERT, TCT-ColBERT, and GPR, which only focus on ad-hoc queries, ConvDR (Yu et al. 2021) stands out as a dense retrieval system designed for conversational search. ConvDR, shown in Figure 3.14, introduced by Yu et al. (2021), learns contextualised embeddings for multi-turn conversational queries and retrieves documents solely using embedding dot products. It is designed to address the challenge of adapting dense retrieval models to conversational search, where queries are often ambiguous and context-dependent. ConvDR uses a teacher-student framework, where a teacher model is trained on a large dataset and a student model is trained to mimic the

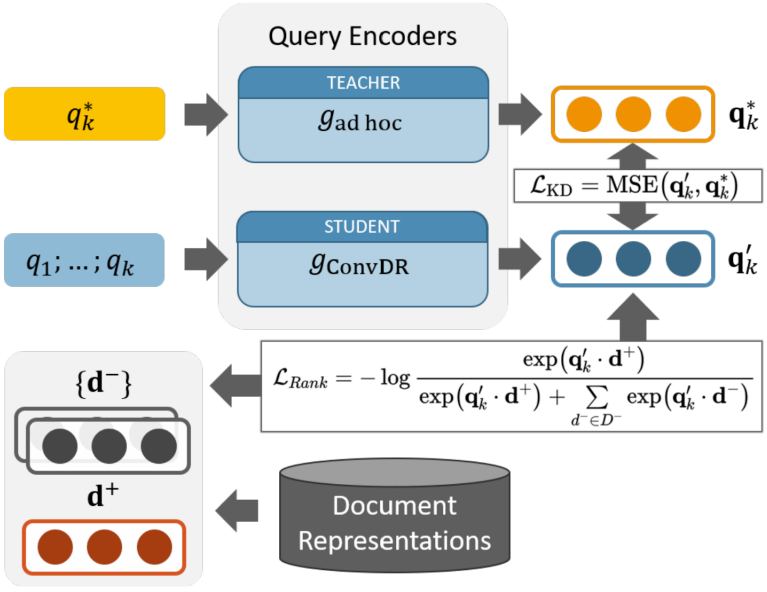


Figure 3.14: Framework of the ConvDR model. Figure taken from (Yu et al. 2021).

teacher's behaviour with limited labelled data. This approach allows ConvDR to effectively capture informative context from the conversation history while ignoring the unrelated context in previous conversation rounds, making it more effective as conversations evolve while previous systems (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*b*, Vakulenko, Longpre, Tu & Anantha 2021) may get confused by the increased noise from previous turns. In the experiments on TREC CAsT 19-20 (see Sections 3.7.2 and 3.7.2) and the OR-QuAC dataset (see Section 3.7.2), ConvDR outperformed the baselines in both few-shot and fully-supervised settings. On CAsT-19 (see Section 3.7.2), ConvDR outperformed the best participating system, CFDA_CLIP_RUN7 (Dalton et al. 2019), which is a well-designed system with state-of-the-art sparse retrieval and neural IR approaches. On CAsT-20 (see Section 3.7.2), ConvDR outperformed every baseline by a large margin except h2oloo_RUN2 (Dalton et al. 2019), which uses a dense-sparse hybrid retrieval model followed by a T5 ranking model with T5-based query reformulation. On OR-QuAC (see Section 3.7.2), ConvDR outperformed all previous methods by huge margins, including the current state-of-the-art baseline, the ALBERT-based ORConvQA system (Qu et al. 2020). These results demonstrate the effectiveness of ConvDR in capturing informative context while ignoring the unrelated context in previous conversation rounds.

To address the ORConvQA task, prior works (Liang et al. 2022, Qu et al. 2021, 2020) have adopted a three-stage architecture, including a retriever, a reranker, and a reader to extract the answers as illustrated in Figure 3.15. First, the retriever retrieves the top K relevant passages from the collection based on a question and its conversation history. The reranker and the reader then respectively rerank and identify an answer in the top K passages. We also adopt this three-stage architecture in our proposed model in Chapter 8.

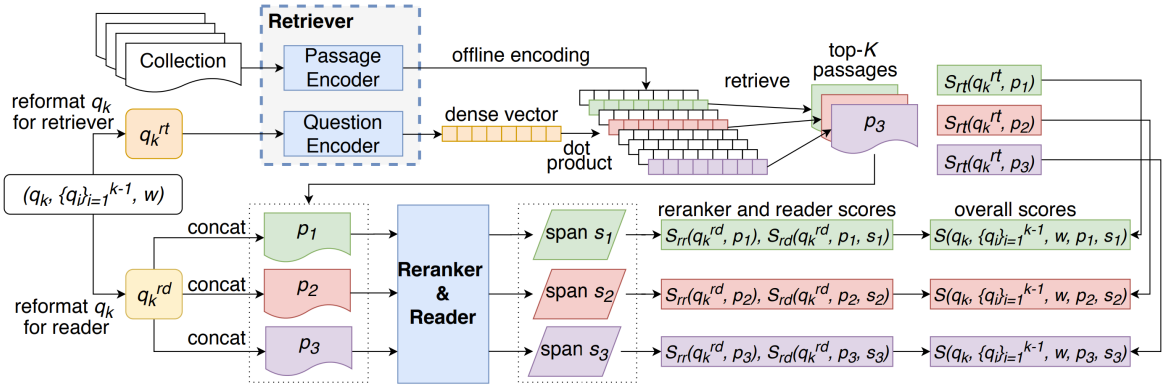


Figure 3.15: Architecture of the end-to-end ORConvQA model. Figure taken from (Qu et al. 2020).

However, in order to investigate the effectiveness of the cross-encoder reranker, we consider a two-stage pipeline including a retriever and a reader, as a baseline for comparison with our system. For the retriever, existing works (Liang et al. 2022, Qu et al. 2021, 2020, Xiong et al. 2020, Yu et al. 2021) have focused on using bi-encoder dense retrieval models (consisting of a question encoder and a passage encoder), which apply neural contextual language models, such as ALBERT or BERT, for encoding the question and passage into low-dimensional vectors and computing their relevance scores as detailed in Section 2.4. For example, Yu et al. (2021) proposed ConvDR, which encodes the question and its history in a dense vector learned with a teacher-student model to mimic a dense representation of the manually rewritten question. ConvDR has also been shown to outperform other retriever models for conversational search such as sparse BM25, and bi-encoders using ALBERT (Qu et al. 2020) or BERT (Karpukhin et al. 2020, Xiong et al. 2020). In Chapter 8, due to the good effectiveness of bi-encoder dense retrievers for passage retrieval, we adapt this type of retrieval models as our retriever. We also consider other recent existing bi-encoder passage retrievers such as TCT-ColBERT (see Section 2.4.2) and CQE (Lin et al. 2021*a*) as baseline passage retrievers.

3.8 Passage Reranking

Passage reranking is an important phase in the ORConvQA pipeline, aimed at refining the order of retrieved passages to enhance the relevance and accuracy of the answers provided to the user queries. The datasets used for evaluating passage reranking align closely with those used for passage retrieval, as previously discussed in Section 3.7.2. Similarly, the evaluation metrics employed for passage reranking can be referenced from the context of passage retrieval (see Section 3.7.3). In addition, we also discuss the passage reranking approaches in Section 3.8.2 as it is the step after passage retrieval.

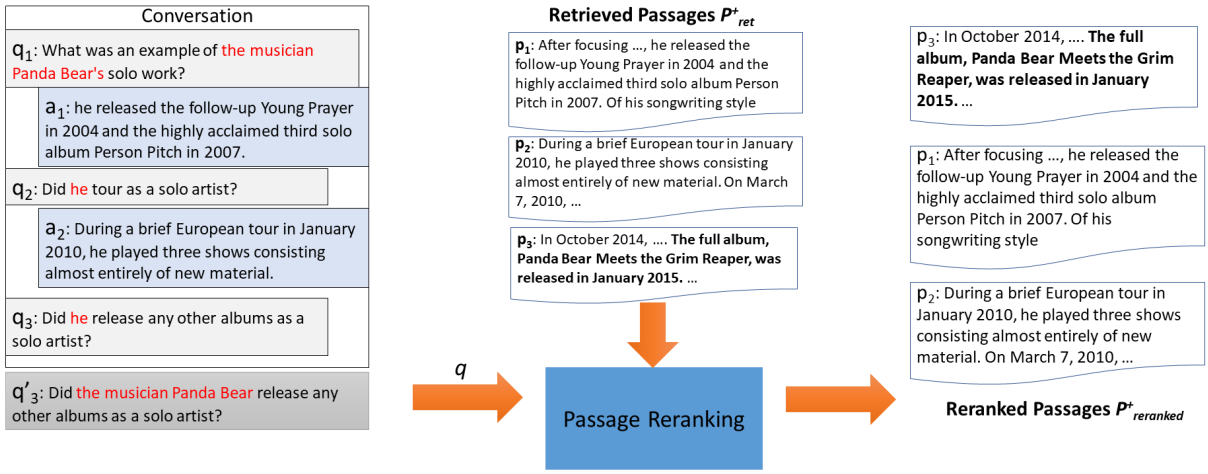


Figure 3.16: An example of the passage reranking task for ORConvQA

3.8.1 Passage Reranking Task Definition

Following Nogueira & Cho (2019), given a query q, a passage corpus C, and an initial list of n retrieved passages $P^+_{ret} = [p_1, p_2, ..., p_n]$, the task aims to refine and reorder these passages based on their relevance to the query q and to produce a more accurate and relevant ranked list $P^+_{reranked} = [p'_1, p'_2, ..., p'_n]$. The query q can be either the ongoing user's question q_k along with its conversation history H_k, or the reformulated question q'_k.

An example of the passage reranking task for Open-Retrieval Conversational Question Answering is depicted in Figure 3.16, illustrating a conversation history of length $k = 2$ with the corresponding questions and answers. In this task, the system takes both the query q and the retrieved passages denoted as $P^+_{ret} = [p_1, p_2, p_3]$ from the previous stage (passage retrieval). The query q can be either the ongoing user's question q_3 along with its conversation history $H_2 = [\langle q_1, a_1 \rangle, \langle q_2, a_2 \rangle]$, or the reformulated question q'3. The objective of this task is to reorder the retrieved passages P^+_{ret} based on their relevance to the query q into $P^+_{reranked} = [p_3, p_1, p_2]$.

3.8.2 Approaches for Passage Reranking

As previously discussed in Section 2.4.1, monoT5 is a passage reranking model. This section highlights its practical application, in particular, by the top-ranked in TREC Conversational Assistance Track (CAsT) 2020-2022 (see Section 3.7.2) participants, who have effectively used monoT5 to improve the relevance of retrieval systems in conversational search contexts. For example, h2oloo (Dalton et al. 2020, 2021), ASCFDA (Dalton et al. 2020), WaterlooClarke (Dalton et al. 2021), udel_fang (Owoicho et al. 2022) and HEATWAVE (Owoicho et al. 2022) employ monoT5 as a ranker. These real-world implementations demonstrate monoT5's robustness and its ability to enhance the precision of information retrieval in response to conversational queries. As a result, in Chapters 6, 7 and 8, we employ the monoT5 model as the passage reranking approach.

3.9 Multi-Task Learning in ORConvQA

Recall from Section 1.2 that this thesis aims to develop an effective ORConvQA system by leveraging Multi-Task Learning (MTL). Multi-Task Learning (MTL) in Conversational Question Answering involves training a single model to handle multiple related tasks simultaneously. In the context of ORConvQA, these tasks may include conversational question answering (Section 3.2), follow-up question identification (Section 3.3), conversational question rewriting (Section 3.4), asking clarifying questions (Section 3.6), and passage retrieval/reranking (Sections 3.7 and 3.8).

Most existing (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Yeh & Chen 2019) ConvQA models that leverage Multi-Task Learning (MTL) use a static form (see details in Section 2.8.2.1) with unchanged tasks' weights during the training epochs. For instance, the recently-proposed History Attention Mechanism (HAM) model (see Section 3.2.4) applied Multi-Task Learning in order to improve the effectiveness of conversational QA. However, the tasks' weights in the model were unchanged during the training state and emphasise the main task. Similarly, FlowDelta (Yeh & Chen 2019) is a ConvQA model that also employed a static MTL method, which sets all tasks' weights equal to one. In the static MTL methods used in HAM and FlowDelta, all of the tasks' weights have not been adjusted throughout the learning phase. As a result, training resources could be diverted to unnecessary tasks with a negative impact on the performance of the learned models (**Gap 1**). In this thesis, in Chapter 5, we include these static MTL methods as baselines, but we also introduce a dynamic MTL approach specifically designed for ConvQA. Our proposed model aims to enhance the effectiveness of the ConvQA task through dynamic MTL. To the best of our knowledge, no prior work has addressed the use of dynamic MTL methods for ConvQA.

On the other hand, MTL methods have been effectively implemented in existing Conversational QA works (Qu et al. 2021, 2020, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Xu et al. 2019*b*, Yeh & Chen 2019). However, in these works, all tasks correspond to the answer span prediction and its auxiliary tasks, which are typically classification tasks. In contrast, recent works (Ide & Kawahara 2021, Jiang et al. 2022*b*, Lee et al. 2022*b*) have adopted a more diverse MTL approach, sharing the learner for both classification and text generation tasks. This approach has resulted in improvements in passage ranking/re-ranking and answer generation. Their models, based on T5 (see Section 2.3.1), focus on generating answers while ranking passages. Similarly, in Chapter 6, we adopt the MTL paradigm in our model. However, while existing approaches have primarily focused on generating answers (text generation task) and passage ranking (classification task), our present work aims to leverage classification tasks for more effective retrieval (**Gap 2**). In Chapter 6, we take advantage of an MTL for Conversational QA. The goal of our proposed models is to improve the effectiveness of both the follow-up question classification and the conversational question rewriting tasks. To the best of our knowledge, no prior work has inherently combined both tasks to effectively identify ambiguous questions. In our proposed models, we employ text generation models including BART (see Section 2.3.1) (shown to be usable for multi-task learning both classification and text generation tasks) and T5 (Raffel

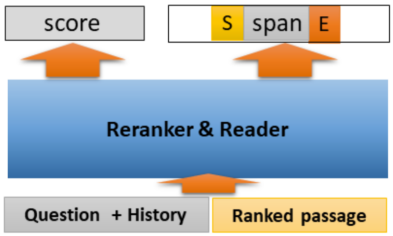


Figure 3.17: Overview of reranker and extractive reader.

et al. 2020) (the state-of-art model for question rewriting, as mentioned in Section 3.4.4).

Moreover, MTL methods have been effectively implemented in various conversational search approaches (Qu et al. 2021, 2020, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Xu et al. 2019*b*, Yeh & Chen 2019). However, all of the tasks in this prior work leverage MTL by sharing the network structure between an extractive reader and its auxiliary tasks, which are typically classification tasks, such as a yes/no question prediction or a follow-up question prediction (**Gap 3**). For example, a number of previous studies (Qu et al. 2021, 2020) have adopted an MTL approach in order to effectively answer the questions posed by the users. Instead, in Chapter 7, we leverage MTL by combining the clarification need classification and the generation of clarifying question to share a single text generation model. This allows to simultaneously determine when, for the current utterance, the system needs to ask a clarifying question and generate a set of clarifying questions based on the user's query and conversation history.

In addition, MTL has been employed in order to efficiently answer the questions posed by the users (Qu et al. 2021, 2020). In this manner, the network structure is shared between the reranker and the reader as shown in Figure 3.15. Doing this, existing works (Qu et al. 2021, 2020) also typically approach reranking and *extractive* reading as classification tasks, with two fully-connected layers (one for the reranker and reader, respectively) added to find an answer span for the retrieved passages (start/end positions) as well as to predict the relevance score of the question to the passage as shown in Figure 3.17. In this thesis, in Chapter 8, we use the Multi-Task Learning of the reranker and the *extractive* reader as our strongest baseline. On the other hand, Nogueira, Jiang, Pradeep & Lin (2020*b*) proposed monoT5, a text generation model, which was fine-tuned to generate the tokens "true" or "false" depending on whether the document is relevant or not to the query. Indeed, the monoT5 model (see Section 2.4.1) has been shown to outperform BERT-based models in passage reranking (Nogueira, Jiang, Pradeep & Lin 2020*b*). In addition, many studies (Karpukhin et al. 2020, Khashabi et al. 2020, Lewis et al. 2020, Raffel et al. 2020) have focused on developing a generative reader which is fine-tuned as a text generation model to extract the answer from the passage. In particular, Khashabi et al. (2020) introduced the UnifiedQA model, which has been shown to yield impressive performances on many extractive QA datasets. However, no existing work has combined monoT5 (passage reranker) and UnifiedQA (reader) to share a single text generation model that directly extracts the answers for the users instead of predicting the start/end positions in a retrieved passage

(**Gap 4**). Moreover, we show that compared to using monoT5 and UnifiedQA separately, a joint learning can enhance the learning efficiency and prediction accuracy of a model for the ORConvQA task, since by sharing the learning model the reranker and reader can simultaneously predict the answer and reranking score. Indeed, a joint learning by sharing a single model trained using MTL reduces the memory needs and speeds up inference (Standley et al. 2020, Sun et al. 2020). In Chapter 8, we combine the effective monoT5 (to rerank the retrieved passages) and the UnifiedQA (to extract the answer from the highest scored passage) models into a strong baseline. To the best of our knowledge, no prior work has combined monoT5 and UnifiedQA by sharing a single text generation model, in order to directly extract the answers instead of predicting the start/end positions in a retrieved passage.

On the other hand, existing works (Qu et al. 2021, 2020), that employ Multi-Task Learning (MTL) for the reranker and the extractive reader have not considered integrating the retriever component into their model. These systems typically handle ambiguities in conversational questions by concatenating the user's current question with its conversational history. This concatenation approach can miss important context and changes in the conversation, which might lead to less accurate answers (Yu et al. 2021). To address this limitation, as previously described in Section 3.4, the conversational question rewriting approach can be employed. Furthermore, these existing systems often adopt a static approach to MTL, with fixed task weights during training. This static allocation of task weights may divert training resources to unnecessary tasks with a negative impact on the performance of the learned models. This represents a gap in the field (**Gap 5**), where there is a lack of dynamic, integrated MTL approaches that can effectively integrate the learning process across all components of ORConvQA systems – namely, the conversational question rewriting, retriever, and reader. Note that, to simplify the model's complexity, we initially focus on these three components, omitting the reranker. To the best of our knowledge, no prior work has combined these three tasks by sharing a uniform model, in order to retrieve relevant passages and extract the answers in retrieved passages simultaneously.

3.10 Summary

In this chapter, we discussed the related work in Open-Retrieval Conversational Question Answering (ORConvQA) providing an in-depth review of numerous tasks, each of which contributes to the improvement of ORConvQA systems. The surveyed areas covered several tasks: Conversational Question Answering (ConvQA), Follow-up Question Identification (FID), Conversational Question Rewriting (QR), Clarification Need Classification (CNC), Asking Clarifying Questions, Passage Retrieval, and Passage Reranking.

ConvQA extends the concept of traditional question answering to a multi-turn conversational setting, promoting more natural interactions. Follow-up Question Identification focuses on determining questions that continue the dialogue coherently. In contrast, Conversational Question

Table 3.5: A summary of models and their approaches used in Open-Retrieval Conversational Question Answering.

Model	QA	Dialogue	Retrieval	MTL	Based model	Chapter	Comment
BiDAF++ (Seo et al. 2017)	/	x	x	x	LSTM	-	-
SDNet (Zhu et al. 2018)	/	x	x	x	RNN	-	-
FlowQA (Huang et al. 2019)	/	x	x	x	Bi-LSTM	-	-
BERT HAE (Qu, Yang, Qiu, Croft, Zhang & Iyyer 2019)	/	/	x	x	BERT	-	-
HAM (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019)	/	/	x	/	BERT	5	Static MTL baseline
Three-way attentive pooling network (Kundu et al. 2020)	x	/	x	x	Bi-LSTM	6	Follow-up question identification baseline
Discriminative+Generative BART (Ide & Kawahara 2021)	x	/	x	/	BART	6	MTL baseline
ANCE (Xiong et al. 2020)	x	x	/	x	RoBERTa/BERT	7	Clarifying question selection baseline
ColBERT (Khattab & Zaharia 2020)	x	x	/	x	BERT	7	Clarifying question selection baseline
TCT-ColBERT (Lin, Yang & Lin 2020, Lin et al. 2021*b*)	x	x	/	x	BERT	7	Clarifying question selection baseline
monoT5 (Nogueira, Jiang, Pradeep & Lin 2020*a*)	x	x	/	x	T5	6,7,8	- Reranker (Chapters 6,7) - Clarifying question selection baseline
GTR (Ni et al. 2021)	x	x	/	x	T5-encoder	7	Proposed clarifying question selection
GPT-3 (Brown et al. 2020)	/	/	x	x		7	Clarifying question generation baseline
miniLM (Wang, Wei, Dong, Bao, Yang & Zhou 2020)	x	x	/	x		7	Clarifying question selection baseline
ConvDR (Yu et al. 2021)	x	/	/	/	DRP (Karpukhin et al. 2020)	8	- First-pass retriever(Chapter 8) - MTL Baseline (Chapter 9)
UnifiedQA (Khashabi et al. 2020)	/	x	x	x	T5	8	Question answering baseline

Rewriting reformulates a conversational question in the context of the previous conversation history, improving the understanding of the question. Asking Clarifying Questions aims to resolve any ambiguities in the initial query, refining the system's responses. Passage Retrieval involves efficiently and effectively searching through large document collections to identify relevant passages, ensuring an alignment with the ongoing conversation context. Passage Reranking further refines the quality of retrieved passages by adjusting their order based on relevance.

In order to effectively address the challenges of ORConvQA, Multi-Task Learning (MTL) has emerged as an integrated solution. By leveraging MTL, the diverse tasks of ORConvQA, including ConvQA, FID, QR, Asking Clarifying Questions, Passage Retrieval, and Passage Reranking, can be integrated to develop a cohesive and effective ORConvQA system.

Each of these tasks contributes to the development of ORConvQA systems. Different approaches can be combined to create more efficient and effective systems. Table 3.5 provides a summary of models and their approaches used in Open-Retrieval Conversational Question Answering.

Our approach to building an effective ORConvQA system is based on the combination of different novel approaches. Indeed, we identified the following general gaps in this chapter:

Gap 1 Current Conversational Question Answering (ConvQA) models using Multi-Task Learning (MTL) lack a dynamic adjustment of task importance during learning. As a result, training resources could be diverted to unnecessary tasks with a negative impact on the performance of the learned models. To improve the effectiveness of MTL for ConvQA, we propose a novel method, called Hybrid Task Weighting, which focuses on adjusting the tasks' weights by modelling the difference between these weights, while still prioritising the main task.

Gap 2 Effective integration between identifying follow-up questions and rewriting conversational questions is currently lacking. This integration allows the system to better understand user intent, address ambiguities, and leverage the context of the conversation.

Gap 3 There is a lack of a comprehensive approach for asking clarifying questions in prior works, as they have either focused solely on generating clarifying questions using generative models or selecting from pre-determined pools. In addition, no prior work has effectively leveraged Multi-Task Learning to simultaneously determine when clarifying questions are needed and generate relevant questions based on the user's initial query and conversation history. By combining both the generation and selection approaches in a uniform framework, we can produce a better set of questions and ensure that the selected question is relevant to the user's query.

Gap 4 There is a lack of an integrated approach that leverages multi-task learning to combine reranking and answer extraction (conversational question answering), sharing a single text generation model for directly extracting answers for the users instead of predicting the start/end positions in a retrieved passage. By sharing the learning model, the reranker and reader can simultaneously predict the answer and reranking score. As a result, this joint learning method can enhance both the learning efficiency and prediction accuracy of a model for the ORConvQA task.

Gap 5 There is no integrated approach using multi-task learning to combine the three tasks: conversational question rewriting, passage retrieval, and answer extraction. In addition, the existing systems use Multi-Task Learning (MTL) in a static approach with fixed task weights during training, which can divert training resources to unnecessary tasks and negatively impact the performance of the learned models.

In the next chapter, we formally introduce our proposed framework, which serves as a solution to bridge these gaps, so as to effectively address the challenges of ORConvQA.

Chapter 4

ORConvQA Framework

4.1 Introduction

In Section 1.3, we stated the hypothesis that an effective Open-Retrieval Conversational Question Answering (ORConvQA) system can be built through the use of Multi-Task Learning (MTL), which combines multiple related tasks. In Chapters 2 and 3, we provided background and reviewed the existing work in the field of ORConvQA. In particular, Chapter 3 discussed the details of the ORConvQA subtasks, including Conversational Question Answering (ConvQA) or Answer Extraction (introduced in Section 3.2), Follow-up Question Identification (FID) (introduced in Section 3.3), Conversational Question Rewriting (QR) (introduced in Section 3.4), Clarification Need Classification (CNC) (introduced in Section 3.5), Asking Clarifying Questions(introduced in Section 3.6), and Passage Retrieval/Reranking (introduced in Sections 3.7 and 3.8). In addition, in Section 3.10, we identified five gaps in the current approaches for ORConvQA and the advancements needed to address the ORConvQA task effectively, namely:

Gap 1 states that existing ConvQA works employing Multi-Task Learning (MTL) lack the capability to dynamically adjust task importance during training.

Gap 2 states the need for improved integration between follow-up question identification and conversational question rewriting to enhance response's accuracy and relevance by better understanding the user's intent, addressing ambiguities, and leveraging the conversation context.

Gap 3 states that prior works on asking clarifying questions have either focused on generating clarifying questions or selecting from pre-determined pools, and that no prior work has effectively leveraged MTL to simultaneously determine when clarifying questions are needed and generate relevant questions.

Gap 4 states that prior works on passage reranking and answer extraction have not effectively leveraged MTL to combine the two tasks, and that no prior work has used a single text generation model to directly extract answers for users instead of predicting the start/end positions in a retrieved passage.

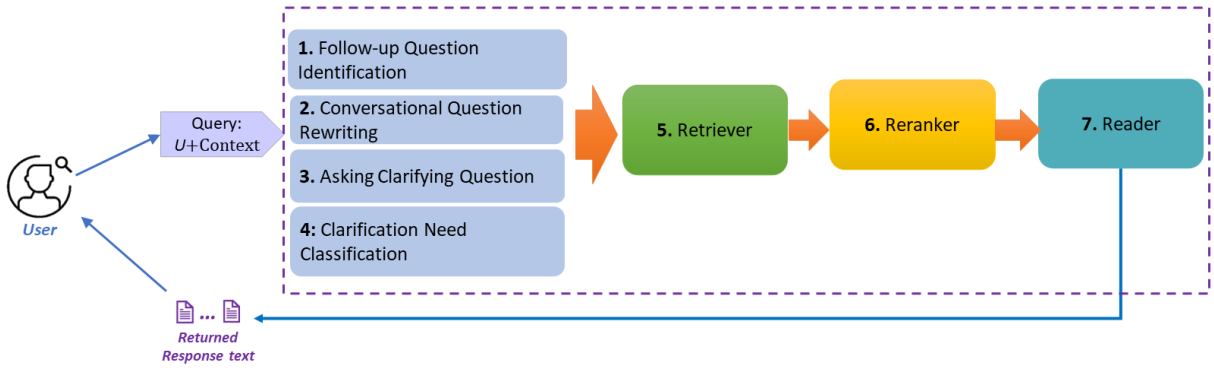


Figure 4.1: An overview of our proposed framework.

Gap 5 states current ORConvQA systems do not effectively integrate conversational question rewriting, passage retrieval, and answer extraction, and they use MTL in a static approach with fixed task weights during training.

This chapter introduces our proposed ORConvQA framework in Section 4.2. The framework aims to bridge these five gaps laid out in Section 3.10 and improve the performance of ORConvQA. Next, in Section 4.3, we present five new methods within our proposed ORConvQA framework, each addressing a different aspect of ORConvQA. Section 4.4 explains how we combine the multiple related tasks using MTL for each of these methods to effectively address the ORConvQA task. Section 4.5 summarises the chapter and provides conclusions.

4.2 Framework Overview

This thesis aims to effectively address the task of Open-Retrieval Conversational Question Answering (ORConvQA) by leveraging Multi-Task Learning (MTL). To address the tasks introduced in Chapter 3, we propose an ORConvQA framework. Figure 4.1 illustrates our proposed ORConvQA framework, which consists of seven main components: (1) Follow-up Question Identification; (2) Conversational Question Rewriting; (3) Asking Clarifying Questions; (4) Clarification Need Classification; (5) Retriever; (6) Reranker; and (7) Reader i.e. the answer extraction component. Specifically, to address the ambiguities of the users' questions for the ORConvQA task, we employ components (1)-(4) as introduced in Section 1.2. These components help us refine the user's query. Subsequently, we pass the refined query to components (5)-(7) for the retrieval, reranking, and answer extraction processes, thereby providing the user with the requested answer.

In Figure 4.1, each box corresponds to a single component. We start by receiving the user's query and conversation history. The ORConvQA system then identifies the need for follow-up questions or conversational question rewriting to address the possible ambiguities in the user's input. The system also performs clarification need classification to determine whether asking clarifying questions is required. If necessary, contextually relevant clarifying questions are

Table 4.1: Notations used in this thesis.

Notation	Definition
q_t	A question at turn t
a_t	An answer at turn t
q_k	A current question
q'_k	A rewritten question of the current question q_k
a_k	An answer to current question q_k
H_k	A conversation history i.e $H_k = [\langle q, a \rangle]$
q_c	A candidate follow-up question
q	q_k and H_k, or q'_k
u_k	A current utterance
c_k	A clarifying question
n_k	An important level for asking a clarifying question
C	A passage collection

generated. Based on the clarified query, relevant passages are retrieved, and then the retrieved passages are reranked to prioritise the most relevant ones. Finally, the system extracts the final answer from the top-ranked passage and provides it as a response to the user. This comprehensive approach leverages Multi-Task Learning to jointly learn and optimise the performance of each component in the ORConvQA system, so as to enhance the overall performance of the deployed system.

Table 4.1 lists the notations used in this thesis. In particular, we explicitly define the terms "question", "query", and "utterance" as they will be used in the following sections and chapters. Following Zamani et al. (2022), an *utterance* is a general term that refers to any spoken or written expression produced by a speaker in a conversation. A *question* is a specific type of utterance that seeks information or clarification from another participant in the conversation. A *query*, on the other hand, (a term used in information retrieval) refers to a request for information from a database or search engine. In the context of conversational information seeking, a query is typically a text-based input that is used to initiate a conversation with a conversational system or search component.

Next, we summarise the functionality of each component and define its specific input and output.

1. **Follow-up Question Identification:** This component is responsible for identifying whether or not the utterance/question is a follow-up question. It helps the system determine if additional information is needed to provide a more accurate and relevant response to the utterance/question. The component takes the current user's question q_k and its conversation history H_k as input then outputs a boolean value indicating whether the question q_{k+1} is linked to the previous conversation history or not. In particular, we define a follow-up

question identification function $FID(\cdot)$, which takes an input, as follows:

$$FID(q_k, H_k, \theta) \rightarrow \text{(True (valid) /False (invalid))} \tag{4.1}$$

where θ represents the learnable parameters of the model. The model is then fine-tuned to predict the target class depending on whether the candidate question is a valid/invalid follow-up to the previous question or not. To evaluate the models in the Follow-up Question Identification task, we use the three test sets of the LIF dataset (Kundu et al. 2020), as described in Section 3.3.2. Since the Follow-up Question Identification task is a binary classification task, we evaluate performances using classical classification metrics, namely precision, recall, F1 and Macro-F1 (see Section 3.6.3). Indeed, following Kundu et al. (2020), reporting Macro-F1 enables the accuracy of topic shift detection to be measured, while F1 focuses solely on follow-up identification as the positive class, as described in Section 3.3.3.

2. **Conversational Question Rewriting:** The Conversational Question Rewriting component aims to reformulate the user's question in a more clear and complete manner, leveraging the context of the ongoing conversation. By doing so, it improves the system's understanding of the user's intent and facilitates a better retrieval of relevant answers. The component takes the current user's question q_k and its conversation history H_k as input then generates the rewritten question q'_k. This reformulation addresses the ambiguity in the user questions by transforming a concise conversational question into a fully-grown, contextualised ad-hoc query. In particular, we define a QR transformation function as $QR(\cdot)$, which takes a text input sequence, as follows:

$$QR(q_k, H_k, \theta) \rightarrow q'_k \tag{4.2}$$

where θ represents the learnable parameters of the model. The model is then fine-tuned to generate the target question q'_k. To evaluate the models in the Conversational Question Rewriting task, we use the test set of the CANARD dataset (Elgohary et al. 2019), as described in Section 3.4.2. For the evaluation of the system performance in Conversational Question Rewriting, which is a generation task, we adopt the ROUGE recall calculated for unigrams (ROUGE-1 recall) and the BLEU metrics, following (Elgohary et al. 2019, Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Vakulenko, Longpre, Tu & Anantha 2021), as explained in Section 3.4.3.

3. **Clarification Need Classification:** The Clarification Need Classification component determines the necessity of asking clarifying questions. It classifies whether the user's query requires further clarification or if it can be directly addressed without additional information. The component takes the current user's question q_k and its conversation history H_k as input and then determines whether the system should ask a clarifying question

in response to user utterance q_k. The target outputs "1", "2", "3", and "4", collectively denoted by n_k, correspond to the level of importance of asking a clarifying question c_k. The lower the number, the less important it is to ask a clarifying question. In particular, we define a CNC function as $CNC(\cdot)$, as follows:

$$CNC(q_k, H_k, \theta) \rightarrow n_k \tag{4.3}$$

where θ represents the learnable parameters of the model and n_k is the importance of (or the need for) asking a clarifying question. The model is then fine-tuned to determine whether the system should ask a clarifying question in response to a user question q_k. To evaluate the performance of the models on Clarification Need Classification, we use the ClariQ (Aliannejadi et al. 2021) dataset, as described in Section 3.5.2. We use typical classification metrics such as precision, recall, and F1, in line with ClariQ, as discussed in Section 3.5.3.

4. **Asking Clarifying Questions:** This component either generates clarifying questions using a generative model or selects them from a pre-determined pool of questions. By asking clarifying questions to the users it is possible to better understand their intents and to refine the system's interpretation of the users' information needs. By doing so, the user can also provide additional information and feedback to the system in order to improve the search results. The component takes the current user's question q_k and its conversation history H_k as input then predicts (generates or selects) a clarifying question c_k that clarifies the current question q_k. The system then receives feedback, q_{k_2}, from the user for the clarifying question c_k. In particular, we define a transformation function as $Asking(\cdot)$, which takes an input, as follows:

$$Asking(q_k, H_k, \theta) \rightarrow c_k \tag{4.4}$$

where θ represents the learnable parameters of the model. The model is then fine-tuned to generate the target clarifying question c_k. For evaluating our proposed method for generating and selecting clarifying questions, we use the TREC Conversational Assistance Track (CAsT) 2022 (Owoicho et al. 2022), as described in Section 3.6.2. The evaluation focuses on P@1 (using a relevance cutoff at two as positive for binary measures) and assesses performance based on three criteria: relevance, novelty, and diversity, as introduced in Section 3.6.3.

5. **Retriever:** The Retriever is responsible for retrieving a set of relevant passages or documents from a large corpus based on the user's query. This component aims to capture the most relevant information that could potentially contain the answer to the user's question.

The component takes the query q which is the current user's question q_k and its conversation history H_k, or the rewritten question q'_k, as input. It then returns a ranked list of n passages $P^+_{ret} = [p_1, p_2, ..., p_n]$ from a text collection C, where the passages are ranked in order of their likelihood of containing the answer to the question q_k. In particular, we define a retrieval function as $Retriever(\cdot)$, as follows:

$$Retriever(q, C) \rightarrow P^+_{ret} \tag{4.5}$$

To evaluate passage retrieval performance, we use the test sets of the CAsT 2022 (Owoicho et al. 2022) and OR-QuAC (Qu et al. 2020) datasets, as described in Section 3.7.2. We use the NDCG, MAP, MRR, and recall evaluation metrics, as introduced in Section 2.7.

6. **Reranker:** The Reranker component takes the output of the Retriever component and further ranks the retrieved passages based on their relevance to the user's question. The Reranker aims to enhances the precision of the retrieval process by identifying the most relevant passages. This component takes the output of the Retriever, a ranked list of n passage $P^+_{ret} = [p_1, p_2, ..., p_n]$, and the query q as input. It then refines and reorders these passages to produce a more accurate and relevant ranked list $P^+_{reranked} = [p'_1, p'_2, ..., p'_n]$. In particular, we define a reranking function $Reranker(\cdot)$, which takes an input, as follows:

$$Reranker(q, P^+_{ret}) \rightarrow P^+_{reranked} \tag{4.6}$$

where q can be the current user's question q_k and its conversation history H_k, or the rewritten question q'_k. To evaluate the passage reranking performance, we use the test sets of the CAsT 2022 (Owoicho et al. 2022) and OR-QuAC (Qu et al. 2020) datasets, as described in Section 3.7.2. We adopt the NDCG, MAP, MRR, and recall evaluation metrics, as discussed in Section 2.7.

7. **Reader:** The Reader component extracts the final answer from the top-ranked passage identified by the Reranker. It uses various natural language processing techniques to locate the precise answer within the retrieved passage and present it to the user as the system's response. This component takes the output of the Reranker, a ranked list of passages $P^+_{reranked} = [p'_1, p'_2, ..., p'_n]$, and the query q as input. It then extracts the precise and relevant answer a from the top-ranked passage and presents them to the user as the system's response. In particular, we define an answer extraction function $Reader(\cdot)$, as follows:

$$Reader(q, P^+_{reranked}, \theta) \rightarrow a_k \tag{4.7}$$

where θ represents the learnable parameters of the model, a_k denotes the response answer to the user, and q can be the current user's question q_k and its conversation history H_k,

Table 4.2: Summary of the different components used in our ORConvQA framework.

Components	Functions	Inputs	Outputs
Follow-up Question Identification	$FID(\cdot)$	q_k, H_k	$valid/invalid$
Conversational Question Rewriting	$QR(\cdot)$	q_k, H_k	q'_k
Clarification Need Classification	$CNC(\cdot)$	q_k, H_k	n_k
Asking Clarifying Questions	$Asking(\cdot)$	q_k, H_k	c_k
Retriever	$Retriever(\cdot)$	q	P^+_{ret}
Reranker	$Reranker(\cdot)$	q, P^+_{ret}	$P^+_{reranked}$
Reader	$Reader(\cdot)$	$q, P^+_{reranked}$	a_k

or the rewritten question q'_k. To conduct the evaluation of the Reader component, we use the OR-QuAC (Qu et al. 2020) and OR-CoQA (Qu et al. 2021) datasets, which are extractive Question Answering (QA) datasets, as described in Section 3.2.2. We use the two evaluation metrics, namely the word-level F1, and the human equivalence score (HEQ), as explained in Section 3.2.3.

Table 4.2 summarises the inputs and the corresponding outputs for each component in our ORConvQA framework.

4.3 Our Proposed ORConvQA Methods

Overall, there are five different proposed methods in our ORConvQA framework that can be built from the components explained above. Inspired by PyTerrier (Macdonald & Tonellotto 2020), we use the notation $\overset{x}{\gg}$ to denote passing the output of one process, with type x, as input to another, where $x = R$ indicates that the type is a ranking of documents and $x = Q$ indicates that the type is a query/question/utterance. In the following, we describe our proposed methods.

1. Following (Choi et al. 2018, Reddy et al. 2019), our first proposed ORConvQA method, $ORConvQA_{1:dynamicMTL}$, simplifies the ORConvQA setting by hard-coding the relevant passages to the user questions. First, follow-up question identification, $FID(\cdot)$, predicts whether a question q_k by the user is *valid*, in the sense that it is likely to be related to the ongoing conversation with the same user using the text from the question q_k and the conversation history H_k. In contrast, an *invalid* question is one that does not logically follow the previous question. Based on this classification, previous questions or answers are selected and incorporated into a refined conversation history, H'_k. For example, if the question is predicted as valid, there remains $H'_k = H_k$. If invalid, it is reduced to just the first question and answer [1], i.e., $H'_k = [q_1, a_1]$. Then, the reader, $Reader(\cdot)$, extracts the

[1] Following (Elgohary et al. 2019), the first question in the conversation in the CANARD dataset is self-contained, meaning it doesn't rely on any previous context or information from the conversation.

answer from the given passage. Therefore, the full $ORConvQA_{1:dynamicMTL}$ method can be written as follows:

$$\begin{aligned} \text{ORConvQA}_{1:dynamicMTL} &= FID(q_k, Hk, \theta) \\ &\overset{H'_k}{\gg} Reader(q_k, \theta) \end{aligned} \tag{4.8}$$

In Chapter 5 of this thesis, we investigate the use of multi-task learning to simultaneously train the FID and the reader components in the $ORConvQA_{1:dynamicMTL}$ method, so as to share their learner structure. We further detail our method in Section 4.4.1.

2. Recall from Sections 3.3 and 3.4 that prior works addressed the problem of ambiguity in the ORConvQA task by introducing the Follow-up Question Identification (Bertomeu et al. 2006, Kirschner & Bernardi 2007, 2009, Kundu et al. 2020) and Conversational Question Rewriting (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Mele et al. 2021, Ren et al. 2018, Vakulenko, Longpre, Tu & Anantha 2021, Vakulenko, Voskarides, Tu & Longpre 2021, Voskarides et al. 2020, Yu et al. 2020) approaches. In our $ORConvQA_{2:FID+QR}$ method, we adapt the $ORConvQA_{1:dynamicMTL}$ method by incorporating the FID and QR functions in order to address ambiguity in conversational questions as mentioned in Section 1.2. In addition, we replace the Reader component with the Retriever component, aiming to improve the overall performance of the system in handling conversational questions.

   First, $FID(\cdot)$ predicts whether a question q_k by the user is *valid*, in the sense that it is likely to be related to the ongoing conversation with the same user using the text from the question q_k and the conversation history H_k. In contrast, an *invalid* question is one that does not logically follow the previous question. Based on this classification, previous questions or answers are selected and incorporated into a refined conversation history, H'_k. For example, if the questions are predicted to be valid, there remains $H'_k = H_k$. If invalid, it is reduced to just the first question and answer, i.e., $H'_k = [q_1, a_1]$. Following this, the $QR(\cdot)$ function takes q_k and H'_k as inputs to reformulate the question q_k into a rewritten question q'_k. Then, the Retriever uses q'_k as the input to retrieve the top N relevant passages from the text collection C. Finally, the Reranker re-scores and re-orders these N passages. Therefore, the full $ORConvQA_{2:FID+QR}$ method can be written as follows:

$$\begin{aligned} \text{ORConvQA}_{2:FID+QR} &= FID(q_k, Hk, \theta) \\ &\overset{H'_k}{\gg} QR(q_k, \theta) \\ &\overset{q'_k}{\gg} Retriever(I, n) \\ &\overset{P^+_{Ret}}{\gg} Reranker(q'_k, \theta) \end{aligned} \tag{4.9}$$

In Chapter 6 of this thesis, we investigate the use of multi-task learning to simultaneously train the FID and QR components in the $ORConvQA_{2:FID+QR}$ method, using a single text generation model. Section 4.4.2 discusses our method in more detail.

3. Recall from Sections 3.5 and 3.6 that to address the ambiguity of conversational questions, previous works (Aliannejadi et al. 2020*a*, 2021, Owoicho et al. 2022) in Open-Retrieval Conversational Question Answering (ORConvQA) have proposed the use of asking clarifying questions. The $ORConvQA_{3:CNC+Asking}$ method consists of six components: conversational question rewriting (*QR*), clarification need prediction (*CNC*), asking clarifying questions (*Asking*), Retriever, Reranker, and Reader. First, the conversational query rewriting component, $QR(\cdot)$, reformulates the current question q_k and its context (conversation history) H_k into a standalone, omission-free rewritten question q'_k, which can be used in the later stages in a decontextualised manner. The clarification need prediction component, $CNC(\cdot)$, estimates the importance of (or the need for) asking a clarifying question, for the rewritten question q'_k, which we denote by n_k. Based on n_k, the system will decide whether a clarifying question should be obtained. If a clarification is not required, the system retrieves passages using the rewritten utterance q'_k through the Retriever component $Retriever(\cdot)$. If a clarification is needed, the system employs the asking clarifying questions component, $Asking(\cdot)$, to produce the most appropriate clarifying question c_k. For each clarifying question c_k, the user feedback f_k will be provided. Subsequently, *QR* reformulates the question q_k using its context (H_k, c_k, and f_k) into a new rewritten question $q''k$. Then, the Retriever uses $q''k$ as the input to retrieve the top N relevant passages from the text collection C. Finally, the Reranker re-scores and re-orders these N passages. Therefore, the full $ORConvQA_{3:CNC+Asking}$ method can be written as follows:

$$\begin{aligned} \text{ORConvQA}_{3:CNC+Asking} = {} & QR(q_k, Hk, \theta) \\ & \overset{q'_k}{\gg} CNC(\theta) \\ & \overset{n_k}{\gg} Asking(q_k, Hk, \theta) \\ & \overset{c_k, f_k}{\gg} QR(q_k, Hk, \theta) \\ & \overset{q''_k}{\gg} Retriever(I, N) \\ & \overset{P^+_{Ret}}{\gg} Reranker(q, N, \theta) \end{aligned} \tag{4.10}$$

In this thesis, Chapter 7 investigates the use of multi-task learning to simultaneously train the clarification need prediction and asking clarifying questions components in the $ORConvQA_{3:CNC+Asking}$ method, using a single text generation model. We further detail this method in Section 4.4.3.

4. Following (Qu et al. 2021, 2020), we adopt a four-stage architecture, including a conversation question rewriting, a retriever, a reranker, and a reader to extract the answers in a method, called $ORConvQA_{4:Reranker+Reader}$. First, the conversational query rewriting component, $QR(\cdot)$, reformulates the current question q_k and its context (conversation history) H_k into a standalone, omission-free rewritten question q'_k, which can be used in the later stages in a decontextualised manner. Then, the Retriever, $Retriever(\cdot)$, using the rewritten query, retrieves the top N relevant passages from the text collection C. To produce the final answer, the Reranker, $Reranker(\cdot)$ and Reader, $Reader(\cdot)$, subsequently re-score and identify the answer within these top-N passages. Therefore, the full $ORConvQA_{4:Reranker+Reader}$ process can be written as follows:

$$\begin{aligned} \text{ORConvQA}_{4:Reranker+Reader} = & QR(q_k, Hk, \theta) \\ & \overset{q'_k}{\gg} Retriever(C, N) \\ & \overset{P^+_{Ret}}{\gg} Reranker(q, N, \theta) \\ & \overset{P^+_{Reranked}}{\gg} Reader(q, \theta) \end{aligned} \tag{4.11}$$

   In Chapter 8 of this thesis, we investigate the use of multi-task learning to train simultaneously the reranker and the reader in the $ORConvQA_{4:Reranker+Reader}$ method by sharing a single text generation model. We discuss our proposed method in more detail in Section 4.4.4.

5. In the $ORConvQA_{5:MTL3Tasks}$ method, we address the problem of ambiguity in conversational questions by incorporating a conversational question rewriting component. The conversational query rewriting component, $QR(\cdot)$, first reformulates the current question q_k and its context (conversation history) H_k into a standalone, omission-free rewritten question q'_k. This rewritten question q'_k can then be used by the Retriever, $Retriever(\cdot)$, to retrieve the top N relevant passages from the text collection C. The reader, $Reader(\cdot)$, then extracts the answer within these top-N passages. Therefore, the full $ORConvQA_{5:MTL3Tasks}$ process can be written as:

$$\begin{aligned} \text{ORConvQA}_{5:MTL3Tasks} = & QR(q_k, Hk, \theta) \\ & \overset{q'_k}{\gg} Retriever(q, C, N) \\ & \overset{P^+_{Ret}}{\gg} Reader(q, \theta) \end{aligned} \tag{4.12}$$

   In Chapter 8 of this thesis, we investigate the use of multi-task learning to train the QR, retriever, and reader components in the $ORConvQA_{5:MTL3Tasks}$ method by sharing their learner structure. We further detail this method in Section 4.4.5.

In the following section, we present how multi-task learning (MTL) is employed to combine

Table 4.3: Task combination overview for Open-Retrieval Conversational Question Answering.

Methods	Tasks							Chapter
	Reader	FID	QR	CNC	Asking	Retriever	Reranker	
$ORConvQA_{1:dynamicMTL}$	x	x						5
$ORConvQA_{2:FID+QR}$		x	x					6
$ORConvQA_{3:CNC+Asking}$				x	x			7
$ORConvQA_{4:Reranker+Reader}$	x						x	8
$ORConvQA_{5:MTL3Tasks}$	x		x			x		8

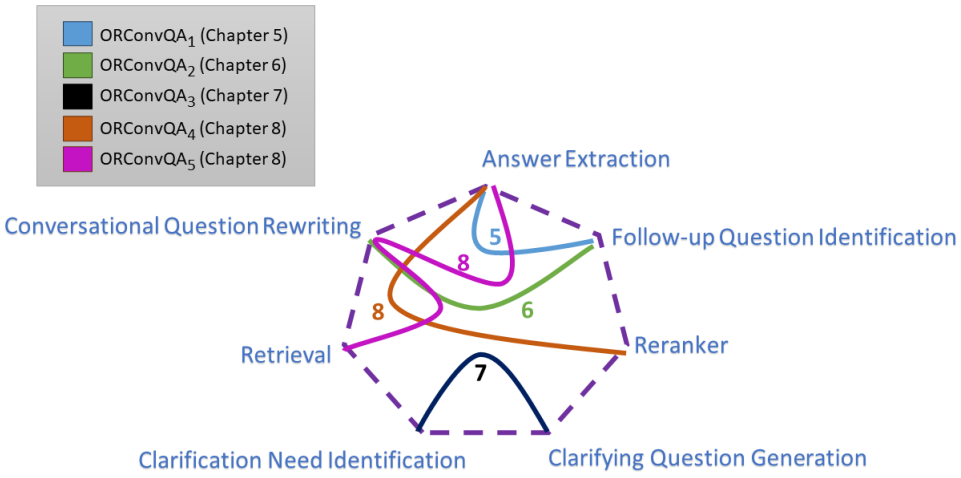


Figure 4.2: Task combination overview for Open-Retrieval Conversational Question Answering. The numbers indicate the corresponding chapters for each combination task.

multiple related tasks within each of the five proposed ORConvQA method to address the gaps identified in Section 3.10.

4.4 Task Combination for MTL

Recall from Section 1.3 that our objective is to address the task of Open-Retrieval Conversational Question Answering (ORConvQA) by leveraging Multi-Task Learning (MTL). MTL is a learning paradigm where multiple tasks are learned jointly to improve the overall performance, as discussed in Sections 2.8 and 3.9. As previously introduced in Section 4.2, we propose five different ORConvQA methods to address the ORConvQA task. This section explains how multiple related tasks in each method are jointly learned using MTL to address the five gaps identified in Section 3.10. An overview of the task combination strategy is depicted in Table 4.3 and Figure 4.2, with numbers indicating the corresponding chapters where each combination task is discussed in detail. In the following, we elaborate on our five proposed methods, initially outlined in Section 4.2, which aim to bridge the five identified gaps in ORConvQA by leveraging MTL.

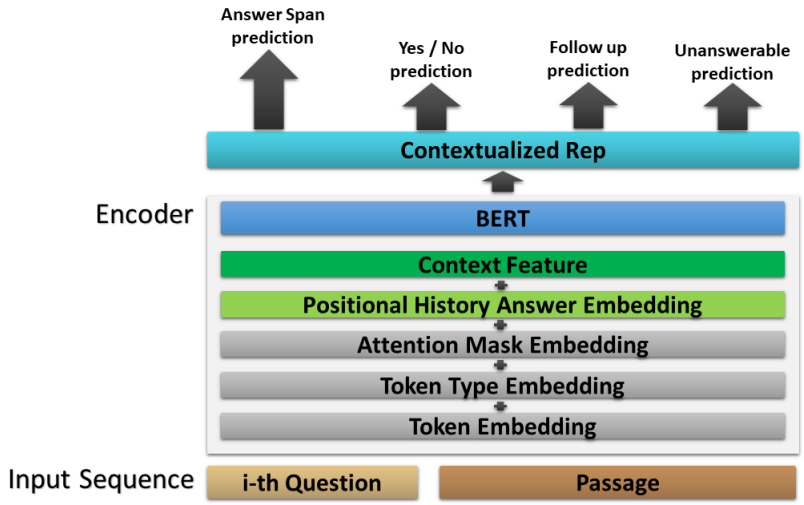


Figure 4.3: The model architecture of the MTL model for answer extraction and follow-up question identification tasks.

4.4.1 Answer Extraction and Follow-up Question Identification

In this section, we aim to address Gap 1 as identified in Section 3.10. Gap 1 states that existing works (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Yeh & Chen 2019) on MTL for ConvQA have successfully adopted static MTL methods. However, there is still room for improvement since during the learning phase' the weights for the all tasks are unchanged and therefore they are not adjusted relative to the importance of the different tasks. Instead, in Chapter 5, we take advantage of a dynamic method in MTL for ConvQA. The goal of our proposed model is to improve the effectiveness of the ConvQA task. As far we know, no prior work has addressed the use of dynamic MTL methods for the ConvQA task. In our proposed MTL approach, we employ the Abridged Linear (Belharbi et al. 2016) for the main Answer Extraction task and the Loss-Balanced Task Weighting (Liu, Liang & Gitter 2019) for the auxiliary tasks, such as Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction. Our approach prioritises the main task after step t during training by setting the task's weight to one while also automatically fine-tuning the tasks' weights by balancing the loss ratio of the auxiliary tasks. In our model, we employ BERT (see Section 2.3.2), which is still a widely used and popular pre-trained model. In the following, we describe in detail our model.

To tackle the Conversational Question Answer (see Section 3.2) and the Follow-up Question Identification (see Section 3.3) tasks, we present our resulting ConvQA model by adopting a Multi-Task Learning approach. Figure 4.3 illustrates the architecture of our model, which consists of three components: an encoder, an answer span predictor and the auxiliary tasks predictor. For the encoder, we deploy a BERT model that encodes the question q_{k+1}, the passage p, and the conversation history H_k as a sequence of m words $C = \{c1, c2, ..., cm\}$ into contextualised token-level ($\hat{T}_k$) and sequence-level ($\hat{S}_k$) representations i.e., $MTL(Reader, FID, \theta) = [\hat{T}_k, \hat{S}_k]$, where $MTL(\cdot)$ is BERT's encoder transformation function. These encodings are customised to

the task by integrating conversation history features. In particular, the $ORConvQA_{1:dynamicMTL}$ method can then be defined using Equation 4.8 as follows:

$$\begin{aligned} ORConvQA_{1:dynamicMTL} = MTL(FID, Reader, \theta) \\ \overset{\hat{T}_k}{\gg} Predict_{answer\_span}(\theta) \\ \overset{\hat{S}_k}{\gg} Predict_{follow-up}(\theta) \end{aligned} \tag{4.13}$$

where $Predict_{answer\_span}(\cdot)$ serves as the classification head (linear layer neural network) for the answer span prediction, while $Predict_{follow-up}(\cdot)$ fulfils the same role for identifying the follow-up questions.

In Chapter 5, to address Gap 1, we perform three consecutive studies to examine the effectiveness of the use of the dynamic Multi-Task Learning for the $ORConvQA_{1:dynamicMTL}$ method. In particular, our investigation aims to answer the following research questions:

RQ 5.1 What is the most effective and efficient Multi-Task Learning method?

RQ 5.2 Does applying the proposed MTL ConvQA model using each of the auxiliary tasks result in effectiveness improvements over learning using only the main task?

RQ 5.3 Does our proposed MTL model lead to not only improving the performance of the main task but also to an improvement in the performances on the auxiliary tasks?

4.4.2 Question Rewriting and Follow-up Question Identification

In this section, we aim to address Gap 2 as identified in Section 3.10. Gap 2 states that while there are existing works for follow-up question identification (Bertomeu et al. 2006, Kirschner & Bernardi 2007, 2009, Kundu et al. 2020) and conversational question rewriting (Lin, Yang, Nogueira, Tsai, Wang & Lin 2020*a*, Mele et al. 2021, Ren et al. 2018, Vakulenko, Longpre, Tu & Anantha 2021, Vakulenko, Voskarides, Tu & Longpre 2021, Voskarides et al. 2020, Yu et al. 2020), none of these works has inherently combined both tasks to more effectively identify ambiguous questions. In Chapter 6, we investigate the combination of both the follow-up question identification task and the conversational question rewriting task for improving the effectiveness of both tasks. Our intuition is that by combining follow-up question identification and conversational question rewriting, the system's response accuracy and relevance can be enhanced. Indeed. we argue that by identifying connections between the user's questions, addressing ambiguities, and leveraging the conversation's context, the system can refine its understanding of the user intent and can provide more precise and relevant responses.

To tackle the Follow-up Question Identification (see Section 3.3) and Conversational Question Rewriting (see Section 3.4) tasks, we propose classification and question rewriting models that leverage historical questions to identify whether a candidate question q_c is a follow-up to the previous question, and to reformulate the current question q_m. Our proposed method uses models

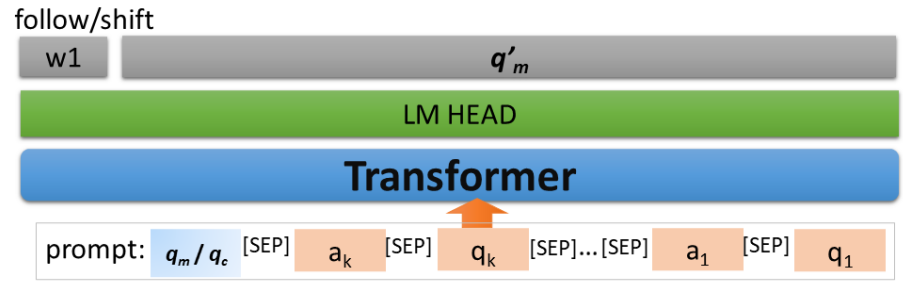


Figure 4.4: A generative model prediction by generating the first token for a classification task and the follow-up tokens for a questing rewriting task.

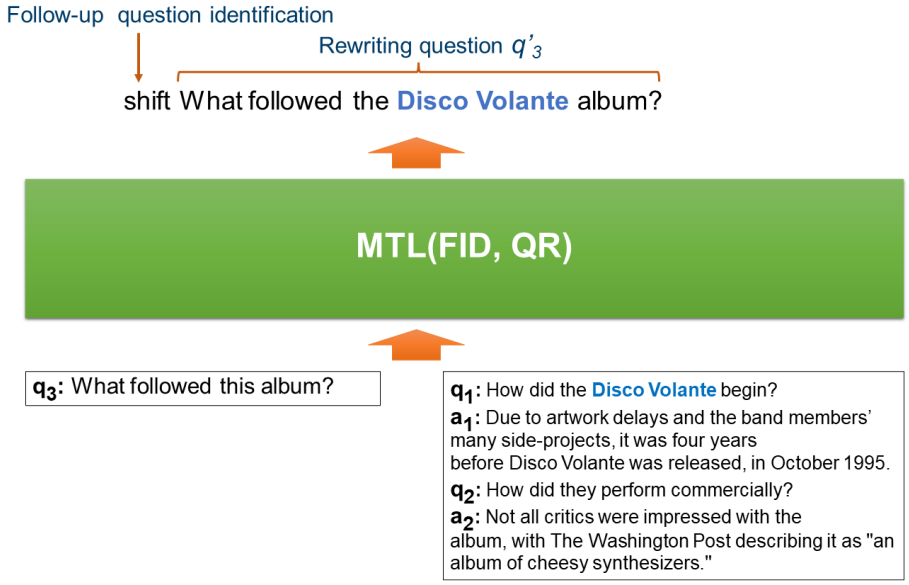


Figure 4.5: Example of output from $MTL(FID, QR)$.

including BART (see Section 2.3.1) and T5 (see Section 2.3.1), which are large pre-trained language models designed for text generation. In particular, text generation approaches can be trained to generate a meaningful textual response based on some input text. Moreover, like BERT, the pre-trained BART and T5 models can be fine-tuned to perform a variety of downstream tasks.

To adopt an MTL approach to a text generation model for jointly learning from both the classification and question rewriting tasks, we deploy generative MTL models to capture the relation between the questions q_c/q_m and the contextual information in the conversation history, including the historical question(s) $\{q_1, q_2, \ldots q_k\}$, and the historical answer(s) $\{a_1, a_2, \ldots a_k\}$, as shown in Figure 4.4. In particular, let $MTL(\cdot)$ denotes a joint learning function as follows:

$$MTL(FID, QR, \theta) \rightarrow w_1, w_2, ..., w_n \tag{4.14}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to generate the

target tokens of length n as shown in Equation (4.14). The token w_1 is either “follow" or “shift"[2] depending on whether the candidate question is a valid follow-up to the previous question or not, while the follow-up tokens $w_2, ..., w_n$ are the output sequence for the target query q'_k. Figure 4.5 shows the example of the input and output of $MTL(FID, QR)$.

In particular, the $ORConvQA_{2:FID+QR}$ method can then be defined using Equation (4.9) as follows:

$$\begin{aligned} ORConvQA_{2:FID+QR} &= MTL(FID, QR, \theta) \\ &\overset{q'_k}{\gg} Retriever(I, n) \\ &\overset{P^+_{ret}}{\gg} Reranker(q'_k) \end{aligned} \tag{4.15}$$

In Chapter 6, to address Gap 2, we conduct two consecutive studies. These investigations aim to examine the effectiveness of combining the tasks of follow-up question identification and conversational question rewriting, with the goal of enhancing the effectiveness of both tasks. Our proposed models employ Multi-Task Learning to simultaneously learn conversational question rewriting and classification sharing a single text generation model. This enables the models to identify whether a question is a follow-up to the previous question and, accordingly, reformulate a question using the dialogue context. In particular, our studies aim to answer the following two research questions:

RQ 6.1 Does the MTL approach of question rewriting and classification using text generation models outperform the single-task learning (STL) of text generation models and existing baselines for the follow-up question identification task?

RQ 6.2 Does the MTL approach of question rewriting and classification using text generation models outperform the single-task learning (STL) of text generation models in the context of the conversational question rewriting and passage retrieval tasks?

4.4.3 Clarification Need Identification and Clarifying Question Generation

In this section, we address the issue highlighted in Gap 3 as identified in Section 3.10. Specifically, Gap 3 states that no previous work has effectively integrated both clarification need classification and the task of asking clarifying questions to more effectively address the step of asking clarifying questions in a mixed-initiative conversational search system. To tackle the tasks Clarification Need Classification (see Section 3.5) and Asking Clarifying Questions (see Section 3.6), we introduce a $ORConvQA_{3:CNC+Asking}$ method consists of four main components, which address the following functionalities:

1. conversational query rewriting, namely T5QR;

[2] We choose “follow" and “shift" as target tokens because T5 tokenises sequences using the SentencePiece approach (Kudo & Richardson 2018), which splits the word “invalid" into two subwords.

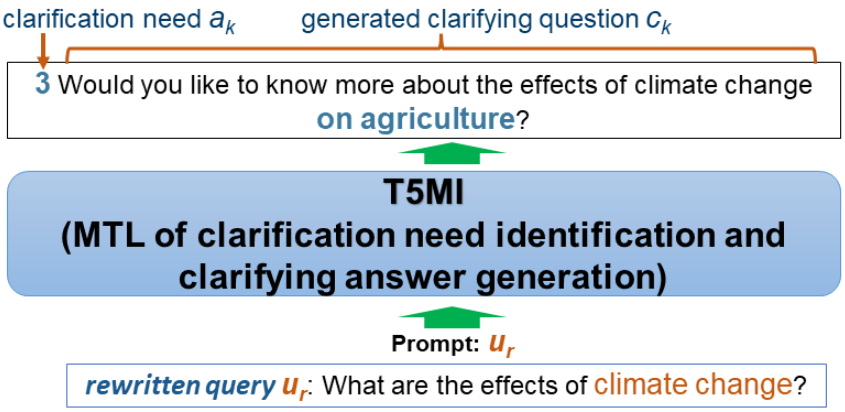


Figure 4.6: MTL of clarification need classification and clarifying question generation.

2. the MTL of the clarification need classification and the generation of clarifying questions, namely T5MI;

3. selecting the clarifying question, namely GTR; and

4. ranking the candidate clarifying questions, namely T5Ranking.

In particular, we describe our proposed T5MI model, which uses a text generation model with Multi-Task Learning for both the clarification need prediction and the generation of the clarifying questions. If a clarification is not required, the system retrieves passages using the rewritten utterance u_r using the retrieval model. If a clarification is needed, the system employs both the generation and selection approaches for clarifying questions – specifically, it generates clarifying questions using the clarifying question generation model and selects from a pre-determined question pool using a clarifying question selection model.

To adopt a Multi-Task Learning (MTL) approach to a text generation model for jointly learning from both the clarification need prediction task and the generation of clarifying question task, the MTL model makes predictions as follows: the first output token is used to estimate the clarification need, while the other output tokens define the output of the clarifying question generation task. To fine-tune the T5 model for a downstream task, we employ the technique of Prompt-based Learning (see Section 2.8.1), which modifies the model by providing a task-specific prompt in addition to the input (Liu et al. 2021). As depicted in Figure 4.6, we deploy a T5 model to analyse the rewritten utterance u_r of the current user's utterance u_k. In particular, we define a jointly learning function as $MTL_{T5MI}(\cdot)$ as follows:

$$MTL_{T5MI}(CNC, Asking, \theta) \rightarrow w_1, w_2, ..., w_n \tag{4.16}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to generate n target tokens, as shown in Equation (4.16). Token w_1 can be "1", "2", "3" or "4" [3] depending on

[3] We follow the assessment procedure of the ClariQ dataset used in this work.

whether the rewritten utterance u_r needs to ask the clarifying question or not, while the follow-up tokens $w_2, ..., w_n$ are the output sequence for the clarifying question c_k.

In particular, the $ORConvQA_{3:CNC+Asking}$ method can then be defined using Equation (4.10) as follows:

$$\begin{aligned} ORConvQA_{3:CNC+Asking} = & \; T5QR(q_k, Hk, \theta) \\ & \overset{q'_k}{\gg} MTL_{T5MI}(CNC, Asking, \theta) \\ & \overset{q'_k}{\gg} GTR(\theta) \\ & \overset{c_k}{\gg} T5Ranking(q'_k, \theta) \\ & \overset{c_k, f_k}{\gg} T5QR(q_k, Hk, \theta) \\ & \overset{q''_k}{\gg} Retriever(I, N) \\ & \overset{P^+_{Ret}}{\gg} Reranker(q''_{k,N}, \theta) \end{aligned} \tag{4.17}$$

In Chapter 7, to address Gap 3, we perform four consecutive studies to investigate the use of multi-task learning to simultaneously determine when a user's query necessitates a clarifying question and generate a set of clarifying questions based on the user's query and conversation history. In particular, our investigation aims to answer the following research questions:

RQ 7.1 Does leveraging the MTL of classification and clarifying question generation on the text generation model improve the effectiveness of the clarification need classification over the existing single-task learning (STL) baselines?

RQ 7.2 How does our proposed hybrid method for generating and selecting clarifying questions compare to other existing baselines?

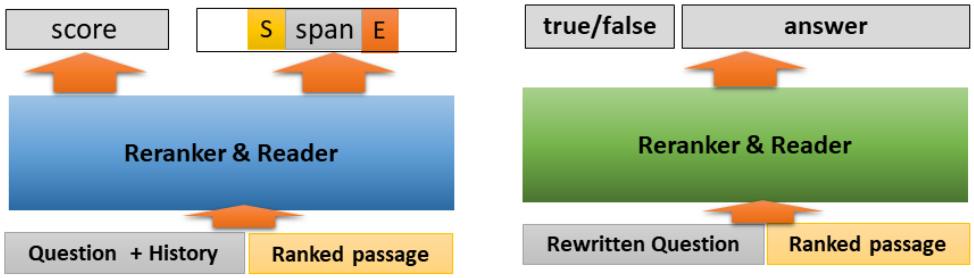
RQ 7.3 How to effectively rewrite the current utterance u_k by using the clarifying question c_k and its feedback for passage retrieval?

RQ 7.4 How effective is our hybrid passage retrieval method compared to existing baselines?

4.4.4 Passage Reranking and Answer Extraction

In this section, we address the issue highlighted in Gap 4 as identified in Section 3.10. Specifically, Gap 4 states that no prior work has combined passage reranking and answer extraction to share a single text generation model that directly extracts the answers for the users instead of predicting the start/end positions in a retrieved passage.

To address the ORConvQA task, prior works (see Section 3.7.4) have adopted a three-stage architecture, including a retriever, a reranker, and a reader to extract the answers. We also adopt this three-stage architecture in our proposed method. By doing so, to tackle the Conversational Question Answering/Answer Extraction (see Section 3.2) and Passage Reranking (see Section 3.8) tasks, we propose an $ORConvQA_{4:Reranker+Reader}$ MTL method, which includes ConvDR (see Section 3.7.4) as a retriever and a newly proposed MTL model called monoQA as reranker



(a) reranker and *extractive* reader (b) reranker and *generative* reader

Figure 4.7: Overview of (a) reranker and extractive reader and (b) reranker and generative reader.

and reader. Our monoQA model uses a text generation model with multi-task learning for both the reranker and reader. Our model, which is based on the T5 text generation model (see Section 2.3.1), is fine-tuned simultaneously for both reranking (in order to improve the precision of the top retrieved passages) and extracting the answer. Unlike previous work (Figure 4.7(a)), monoQA makes predictions by generating the first token for the passage reranking task, followed by the other tokens for the answer extraction task, as illustrated in Figure 4.7(b).

To adopt an MTL approach to a text generation model for jointly learning from both passage reranking and answer extraction, the MTL model makes predictions by generating the first token for the passage reranking task and the follow-up tokens for the answer extraction task. In particular, when fine-tuning the T5 model for a downstream task, we use Prompt-based Learning (see Section 2.8.1), which is a method to modify the model by using a task-specific prompt together with the input (Liu et al. 2021). We deploy a T5 model to capture the relation between the rewritten question q_r of the current question q_c and the passage p. In particular, we define a $MTL_{monoQA}(\cdot)$ joint learning function as follows:

$$MTL_{monoQA}(Reranker, Reader, \theta) \rightarrow w_1, w_2, ..., w_n \tag{4.18}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to generate n target tokens, as shown in Equation (4.18), The token w_1 is "true" or "false"[4] depending on whether the passage is relevant or not to question q_r, while the follow-up tokens $w_2, ..., w_n$ are the output sequence for the answer of the question q_r.

In particular, the $ORConvQA_{4:Reranker+Reader}$ method can then be defined using Equation (4.11) as follows:

$$\begin{aligned} ORConvQA_{4:Reranker+Reader} = & \; QR(q_k, H_k) \\ & \overset{P^+_{ret}}{\gg} Retriever(q, C, N) \\ & \overset{P^+_{ret}}{\gg} MTL_{monoQA}(Reranker, Reader, \theta) \end{aligned} \tag{4.19}$$

[4] We choose "true" and "false" as target tokens following monoT5 (Nogueira, Jiang, Pradeep & Lin 2020*b*).

In Chapter 8, to address Gap 4, we perform three consecutive studies to investigate the use of Multi-Task Learning (MTL) to improve performance on the ORConvQA task by sharing the reranker and reader's learned structure. We propose monoQA, which uses a text generation model with multi-task learning for both the reranker and reader. Our model, which is based on the T5 (see Section 2.3.1) text generation model, is fine-tuned simultaneously for both reranking (in order to improve the precision of the top retrieved passages) and extracting the answer. Moreover, we consider the use of different models to initialisation monoQA during training, since we propose to combine monoT5 (see Section 2.4.1) and UnifiedQA (see Section 3.2.4) to share a single text generation model. Moreover, both monoT5 and UnifiedQA are fine-tuned based on the `t5-base` model. Therefore, we investigate which of monoT5, UnifiedQA, and `t5-base`, are suitable for initialising monoQA.

In particular, our investigation aims to answer the following five research questions:

RQ 8.1 How to select the best training model checkpoint in validation steps, i.e. the best validation loss, the best word-level F1, or the best relevance accuracy?

RQ 8.2 Which of the prompts lead to the best performance of our monoQA model?

RQ 8.3 Which model to use for initialising monoQA, namely which of: monoT5, UnifiedQA, and `t5-base`, lead to the best performance of monoQA on the ORConvQA task?

RQ 8.4 How does monoQA compare to other existing baselines?

RQ 8.5 How does our proposed $ORConvQA_{4:Reranker+Reader}$ MTL method, which is a three-stage pipeline (retriever, reranker, and reader), compare to the two-stage pipeline (retriever and reader) baselines?

4.4.5 Question Rewriting, Passage Retriever, and Answer Extraction

In this section, we address the issue highlighted in Gap 5 as identified in Section 3.10. Specifically, Gap 5 states that no prior work has combined the three tasks: Conversational Question Rewriting (see Section 3.4), Passage Retrieval (see Section 3.7), and Answer Extraction (see Section 3.2), in a uniform model.

To address these three tasks, we propose an $ORConvQA_{5:MTL3Tasks}$ MTL method using a uniform model that integrates three components: conversational question rewriting, the retriever, and the reader. To achieve this, we introduce the $ORConvQA_{5:MTL3Tasks}$ method. In particular, let $MTL(\cdot)$ denotes a joint learning function of the conversational question rewriting ($QR(\cdot)$), passage retrieval ($Retriever(\cdot)$), and answer extraction ($Reader(\cdot)$). Therefore, the $ORConvQA_{5:MTL3Tasks}$ method can then be defined using Equation (4.12) as follows:

$$ORConvQA_{5:MTL3Tasks} = MTL(QR, Retriever, Reader, \theta) \tag{4.20}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to learn from teacher-generated embeddings on oracle reformulated questions (as mentioned in Section 3.7.4),

retrieve the relevant passages, and extract the answers from retrieved passages.

In Chapter 8, to address Gap 5, we perform a consecutive study to investigate the use of Multi-Task Learning (MTL) to improve the performance on the ORConvQA task by sharing the conversational question rewriting, retriever, and reader's learned structure. Our proposed $ORConvQA_{5:MTL3Tasks}$ method incorporates the reader into the bi-encoder ConvDR (see Section 3.7.4) model, which combines the conversational question rewriting with retriever components. The model is fine-tuned simultaneously for conversational question rewriting, passage retrieval, and extracting the answer. In particular, our investigation aims to answer the following research questions:

RQ 8.6 How does our proposed $ORConvQA_{5:MTL3Tasks}$ MTL method, which is an MTL of conversational question rewriting, retriever, and reader, compare to other existing baselines?

4.5 Conclusions

This chapter introduced our proposed ORConvQA framework, consisting of five different methods ($ORConvQA_{1-5}$), which aim to bridge the gaps in existing ORConvQA research and improve the performance of ORConvQA system. The five ORConvQA methods are:

- $ORConvQA_{1:dynamicMTL}$: This method aims to address Gap 1 by leveraging the dynamic Multi-Task Learning to jointly learn Answer Extraction and Follow-up Question Identification in Chapter 5.
- $ORConvQA_{2:FID+QR}$: This method aims to address Gap 2 by combining the follow-up question identification and conversational question rewriting tasks to improve the effectiveness of both tasks in Chapter 6.
- $ORConvQA_{3:CNC+Asking}$: This method aims to address Gap 3 by proposing a T5MI model, which uses a text generation model with Multi-Task Learning for both the clarification need prediction and the generation of the clarifying questions in Chapter 7.
- $ORConvQA_{4:Reranker+Reader}$: This method aims to address Gap 4 by proposing a monoQA model, which uses a text generation model with multi-task learning for both the reranker and reader in Chapter 8.
- $ORConvQA_{5:MTL3Tasks}$: This method aims to address Gap 5 by leveraging the dynamic Multi-Task Learning to jointly learn Conversational Question Rewriting, Passage Retrieval, and Answer Extraction in Chapter 8.

All five ORConvQA methods leverage MTL to jointly learn multiple related tasks and enhance an ORConvQA system. MTL helps to improve the overall performance of the systems, as each task can help to inform the other tasks. The next chapter, Chapter 5, discusses the studies we will conduct to address the gaps we have identified and presented.

Chapter 5

MTL of Answer Extraction and its Auxiliary Tasks

5.1 Introduction

In Section 3.2, we provided an overview of the Conversational Question Answering (ConvQA) task, which consists in answering a question from a given passage in the form of a dialogue. Recall that in the ConvQA task, in order to predict an answer, the system needs to extract text spans from a given passage and understand the question based on the given conversational history. Recently, the advancement in neural language modelling such as BERT (Devlin et al. 2019) (as introduced in Section 2.3.2), and the introduction of two large-scale datasets, namely CoQA (Reddy et al. 2019) and QuAC (Choi et al. 2018) have further boosted research in the ConvQA task, as already discussed in Section 3.2.2. In particular, QuAC introduces a main task, namely Answer Extraction, which consists in answering a question by extracting text spans from a given passage as well as some *auxiliary* tasks, namely Follow-up Question Identification (FID), Yes/No prediction, and Unanswerable prediction (see Section 3.2).

In Section 2.8, we introduced Multi-Task Learning (MTL), which is a way to learn multiple different but related tasks simultaneously. MTL has emerged as a popular solution to combine various sub-tasks involved in ConvQA, such as Answer Extraction, Follow-up Question Identification (FID), Yes/No prediction, and Unanswerable prediction, all within a uniform model (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019). MTL can also be used to leverage the auxiliary tasks to improve the performance of a system on the main task. For example, for the QuAC dataset, Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer (2019), and Yeh & Chen (2019) adopted an MTL approach that jointly learns the *auxiliary* tasks and the main task by sharing the encoder thereby leading to an improvement in the used ConvQA model.

In this chapter, we start to investigate the use of Multi-Task Learning (MTL) for enhancing the performance of Conversational Question Answering (ConvQA) systems. Building upon the discussion of Multi-Task Learning (MTL) in Sections 2.8 and 3.9, and the definition of the

ConvQA task in Section 3.2, we explore how the integration of various sub-tasks within ConvQA, such as Answer Extraction, Follow-up Question Identification (FID), Yes/No prediction, and Unanswerable prediction, can be enhanced using MTL. In particular, we focus on leveraging the advancements in neural language models, like BERT, and the insights gained from large-scale datasets like QuAC to address the task of ConvQA leveraging MTL. This chapter presents our approach to employing MTL in ConvQA, highlighting how it can improve the system performance by effectively learning both the main answer extraction task and its auxiliary tasks simultaneously.

Recall from Section 2.8.2.1 that the MTL methods can be categorised into static or dynamic methods. In the static MTL methods, each of the task's weights used to combine the loss functions of the various used tasks during training are unchanged throughout the learning phase. As a result, they might overemphasise less important tasks at different stages of training. In contrast, in the dynamic MTL methods, each of the task's weights is adjusted automatically to balance the loss rate (Liu, Liang & Gitter 2019) or to balance the weights across tasks (Kendall et al. 2018). However, the implementation of a dynamic MTL method while more complicated has a lower training efficiency than static methods. Therefore, in this chapter, we investigate the efficiency and effectiveness of static and dynamic MTL methods in the ConvQA scenario.

As discussed in Section 3.2.4, existing (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019, Yeh & Chen 2019) Conversational Question Answering (ConvQA) models that leverage Multi-Task Learning (MTL) use the static method with unchanged tasks' weights during the training epochs. For instance, the recently proposed History Attention Mechanism (HAM) model (see Section 3.2.4) attempted to apply Multi-Task Learning in order to improve the effectiveness of conversational QA. However, the tasks' weights in the model were unchanged during the training state and emphasised the main task. FlowDelta (Yeh & Chen 2019) is a ConvQA model that also employs a static MTL method, which sets all tasks' weights equal to one. In the static MTL methods used in HAM and FlowDelta, all of the tasks' weights have not been adjusted throughout the learning phase. As a result of this limitation (identified in Section 3.10 as Gap 1), training resources could be diverted to unnecessary tasks with a possible negative impact on the performance of the learned models.

Section 1.3 presented our thesis statement, hypothesising that a dynamic Multi-Task Learning approach can improve the effectiveness of ConvQA by simultaneously training the main Answer Extraction task and its auxiliary tasks, such as Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction. This chapter aims to test this hypothesis by incorporating these tasks into a unified model, thereby enhancing the ConvQA system. We introduced our proposed $ORConvQA_{1:dynamicMTL}$ method in Section 4.4.1.

Therefore, in this chapter, we address the issue discussed earlier regarding static task weighting, as referred to in Gap 1. We propose a novel dynamic method, called Hybrid Task Weighting to improve the effectiveness of MTL for ConvQA. This method is designed to address the issues raised in Gap 1 by enabling the dynamic adjustment of task weights based on modelling the

differences between these weights, while still prioritising on the main Answer Extraction task.

The contributions of this chapter are summarised as follows:

(1) We leverage dynamic Multi-Task Learning with BERT[1] to effectively address the task of learning the main Answer Extraction task with its auxiliary tasks including Yes/No prediction, Follow-up Question Identification, and Unanswerable prediction;

(2) To further enhance the performance of Multi-Task Learning, we introduce a hybrid strategy, which automatically fine-tunes the multiple tasks' weights along the learning steps. Our method uses Abridged Linear for the main Answer Extraction task and Loss-Balanced Task Weighting for its auxiliary tasks;

(3) The proposed hybrid method yields the best performance improvements over the baselines on the QuAC dataset.

The rest of the chapter is structured as follows: Section 5.2 states the task problem, along with our proposed model to address the task. In Section 5.3, we revisit the concepts of Multi-Task Learning (MTL) discussed in Sections 2.8 and 3.9. This section provides a recap of static MTL, dynamic MTL, and introduces our proposed hybrid MTL methods as they are applied to our proposed model. We present our experimental setup in Section 5.4 and show the results of the experiments in Section 5.5. Finally, we provide concluding remarks in Section 5.6.

5.2 The BERT ConvQA Model

In this section, we describe our proposed Conversational Question Answering (ConvQA) model, which is based on the BERT architecture. This model is employed based on the $ORConvQA_{1:dynamicMTL}$ method outlined in Section 4.4.1. We explain how this model is used to perform the main Answer Extraction task as well as auxiliary tasks, namely Follow-up Question identification, Yes/No prediction, and Unanswerable prediction. In Section 5.2.1, we first expand on the definition of the Conversational Question Answering (ConvQA) and Follow-up Question Identification tasks, as presented in Sections 3.2.1 and 3.3.1. Thereafter, an overview of the proposed ConvQA model is provided in Section 5.2.2. Section 5.2.3 describes how additional features are integrated with the BERT encoder. Then we explain how predictions are made for the main Answer Extraction task as well as the auxiliary tasks in Sections 5.2.4 and 5.2.5, respectively.

5.2.1 Task Definition

We address the tasks of Conversational Question Answering (ConvQA) and Follow-up Question Identification (FID), as described in Sections 3.2.1 and 3.3.1, respectively. Table 5.1

[1] Preliminary experiments found BERT to be more effective than ALBERT or RoBERTa.

Table 5.1: Notations used in Chapter 5.

Notation	Definition
q_k	A current question
q'_k	A rewritten question of the current question q_k
a_k	An answer to current question q_k
H_k	A conversation history i.e. $H_k = [\langle q, a \rangle]$
P^+	A given passage
W	A sequence of m words i.e. $W = \{w_1, w_2, ..., w_m\}$
$\hat{T}_k$	A contextualised token-level representations
$\hat{S}_k$	A contextualised sequence-level representations
$FID(\cdot)$	A Follow-up Question Identification function
$Reader(\cdot)$	A Conversational Question Answering function

presents the notations used in this chapter. These notations include a subset of the notations previously defined in Table 4.1 of Chapter 4, as well as specific notations for this chapter. The ConvQA ($Reader(\cdot)$ function) and FID ($FID(\cdot)$ function) tasks were formalised in Equations (4.7) and (4.1) as follows:

$$\begin{aligned} Reader(q_k, H_k, P^+) &\rightarrow a_k \\ FID(q_k, H_k) &\rightarrow \text{(valid/invalid)} \end{aligned} \tag{5.1}$$

where $Reader(\cdot)$ is a function that extracts the answer a_k from a given passage P^+ using the user's current question q_k and its conversation history $H_k = [\langle q, a \rangle]$ (a list of $k-1$ questions and ground truth answer pairs), while $FID(\cdot)$ is a function that predicts whether or not the candidate follow-up question q_k is a valid follow-up question. Note that the tasks of ConvQA and FID can be formulated as classification tasks. For ConvQA, the task aims to predict the answer a_k by predicting the answer span indices i, j within passage P^+. By doing this, the ConvQA model classifies which span (or which pair of start i and end j indices) is the correct answer. On the other hand, the FID task aims to predict whether the current question q_k is classified as *valid* or *invalid* based on its coherence and relevance within the ongoing conversation context. In the following section, we describe our proposed model to address these tasks.

5.2.2 Model Overview

To tackle the tasks described in the above Section 5.2.1, we present our $ORConvQA_{1:dynamicMTL}$ model (as introduced in Section 4.4.1) by adopting a Multi-Task Learning approach. Figure 5.1 illustrates the architecture of our model, which consists of three components: an encoder, an answer span predictor and the auxiliary tasks predictor. For the encoder, we deploy a BERT model that encodes the question q_{k+1}, the passage p, and the conversation history H_k as a sequence of m words $W = \{w_1, w_2, ..., w_m\}$ into contextualised token-level ($\hat{T}_k$) and sequence-level ($\hat{S}_k$) representations i.e., $MTL(Reader, FID, \theta) = [\hat{T}_k, \hat{S}_k]$, where $MTL(\cdot)$ is BERT's encoder transformation function. These encodings are customised to the task by integrating conversation history features

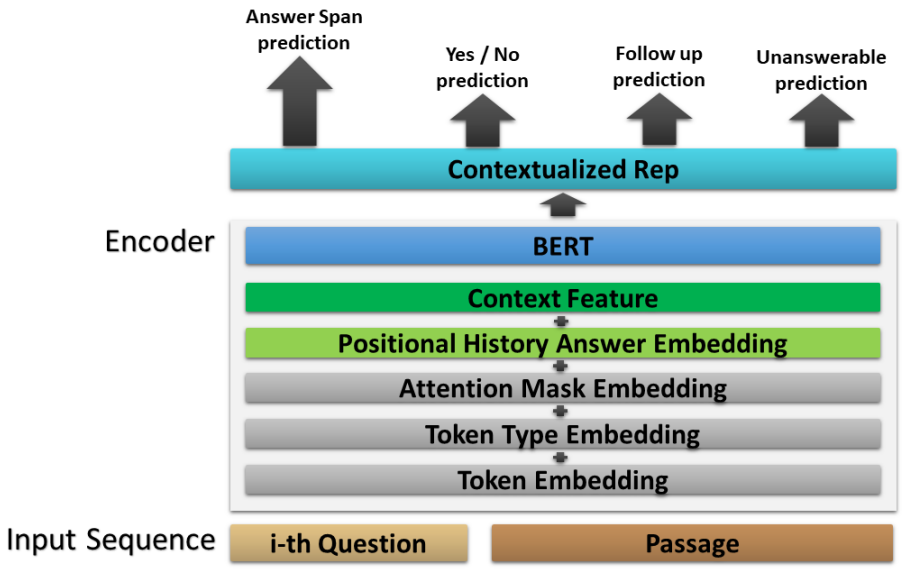


Figure 5.1: The model architecture.

(Section 5.2.3). In particular, as defined in Equation (4.8), the $ORConvQA_{1:dynamicMTL}$ model can then be defined as follows:

$$\begin{aligned} ORConvQA_{1:dynamicMTL} &= MTL(FID, Reader, \theta) \\ &\gg^{\hat{T}_k} Predict_{answer\_span}(\theta) \\ &\gg^{\hat{S}_k} Predict_{follow-up}(\theta) \end{aligned} \tag{5.2}$$

Finally, the representations $\hat{T}_k$ serve as the input for the predictors' modules ($Predict_{answer\_span}$) , which will be detailed later in Section 5.2.4. Whereas the representations $\hat{S}_k$ serve as the input for the predictors' modules ($Predict_{follow-up}$), which will be detailed later in Section 5.2.5.

5.2.3 BERT Encoder Features

In our $ORConvQA_{1:dynamicMTL}$ model (see Equation (5.2)), we modify the BERT input to encapsulate two features – the Positional History Answer Embedding (PosHAE) and the Context Features:

PosHAE: We use this modification feature introduced by Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer (2019) to capture the conversation history into BERT. As exemplified by the example in Table 5.2, the questions in the QuAC dataset often refer to entities in the previous answer(s). Consequently, PosHAE (see Section 3.2.4) has been introduced to embed the relative position of the terms that occur in previous answers within the conversational history H_k. PosHAE helps BERT to track and understand references within the conversation history, improving its accuracy in answering current questions (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019).

Table 5.2: An example dialog from the ConvQA dataset.

p	In 1934 he batted .344 with 18 home runs, 104 RBI, 102 runs scored and 192 hits in 138 games. After a disappointing final season with the White Sox which saw Simmons bat just .267 with 16 home runs and 79 RBI in 128 games (first time in his 11-year career he did not reach .300+ & 100 RBI) he rebounded by hitting .327 with 13 home runs, 112 RBI and 96 runs scored in 1936 for the Detroit Tigers. In 1937 he struggled again, this time with the Washington Senators, batting just .279 with 8 home runs and 84 RBIs in 103 games. He rebounded with a stellar season in 1938, batting .302 with 21 home runs and 95 RBI in just 125 games for Washington. His 21 home runs that year gave Simmons the distinction of being the first player to hit 20 home runs in a year for the Senators. CANNOTANSWER
q_1	Where was he playing in 1933?
a_1	CANNOTANSWER
q_2	What did he do between 1933 and 1938?
a_2	In 1934 he batted .344 with 18 home runs, 104 RBI, 102 runs scored and 192 hits in 138 games.
q_3	Did he lead the league in hitting?
a_3	After a disappointing final season with the White Sox

Context Feature: Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer (2019) introduced a feature vector that encodes the turn number of history answer named Positional History Answer Embedding (PosHAE) which adapted from Qu, Yang, Qiu, Croft, Zhang & Iyyer (2019) and Choi et al. (2018), Yeh & Chen (2019) mark history answers in the passage as context feature. We integrate contextual knowledge of the previous answer within the passage into BERT by following Yeh & Chen (2019) who applied BiDAF++ (Choi et al. 2018). Indeed, BiDAF++ is designed to learn token embeddings that denote whether a token in the passage p is part of a recent answer (see Section 3.2.4).

In addition, we add these two features to tailor BERT specifically for the task of ConvQA.

5.2.4 Answer Extraction

Given the token-level representations $\hat{T}_k$ produced by BERT, we compute the probability of each token being the start token or the end token in order to predict the answer span. In particular, to map a token representation $\hat{T}_k$ to a logit, two sets of parameters are learned for the start vector and the end vector, respectively. After that, the softmax function is applied to obtain probabilities across all token representations $\hat{T}_k$. From this, we obtain p_m^S, and p_m^E, which are the probabilities of token m being the start token or end token, as follows:

$$p_m^S = \text{softmax}(\hat{T}_k(m)) \tag{5.3}$$

$$p_m^E = \text{softmax}(\hat{T}_k(m)) \tag{5.4}$$

5.2.5 Auxiliary Task Prediction

All auxiliary tasks in our dataset, including Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction, are formulated as binary or multi-label classification

tasks. To address each auxiliary task, we take the sequence-level representations $\hat{S}_k$ that are obtained from the $[CLS]$ token (which is the first token of the sequence, produced by BERT). These representations are then used in Equation (5.2), where they serve as the basis for calculating the probability distributions or classifications necessary for each auxiliary task. We apply a softmax function on $\hat{S}_k$ to compute the posterior probabilities across the true and false labels for the multi-label tasks; for the binary tasks, we use a sigmoid function. After that, we compute cross-entropy loss for the multi-label tasks and the binary cross-entropy loss functions for the binary tasks. Next, we describe the MTL approaches to combine the loss functions from the auxiliary tasks with the loss calculated on the main task.

5.3 Multi-Task Learning for ConvQA

In this section, we explain how Multi-Task Learning (MTL) approaches are used to integrate the loss functions of auxiliary tasks with that of the main task. As previously described in Section 2.8.2.1, existing loss weighting approaches in Multi-Task Learning (MTL) can be categorised as either *static* or *dynamic*, depending on whether the importance assigned to the loss of each task remains fixed or varies during the learning process. For completeness, we reiterate the details of *static* and *dynamic*, in Sections 5.3.1 and 5.3.2, respectively. In addition, we provide the details of our proposed hybrid task weighting method in Section 5.3.3.

5.3.1 Static MTL

Static MTL methods are the most frequently used MTL approaches for ConvQA. They apply a fixed weighting of the different loss functions of the auxiliary tasks throughout the training process. This strategy is simple but yet expensive to fine-tune. Instead, many previous studies just report the use of uniform weights for tasks, such as setting all of them to 1.0 (Yeh & Chen 2019), or setting their sum to 1 (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019). The total loss function of this method is defined as follows:

$$\mathcal{L}_{total} = \mu \mathcal{L}_{ans} + \lambda \sum_{a \in A} \mathcal{L}_a \tag{5.5}$$

where A is the set of auxiliary tasks, μ is the weight for the main task and λ is the weight for A.

5.3.2 Dynamic MTL

Applying a static weighting to the auxiliary tasks can unnecessarily apply learning resources to the auxiliary tasks, instead of the main task. Indeed, this can lead to an overfitting to the wrong task and hence to underfitting on the main task (Chen et al. 2018). On the other hand, in the dynamic MTL approaches, the loss weighting of the tasks is instead continually adjusted

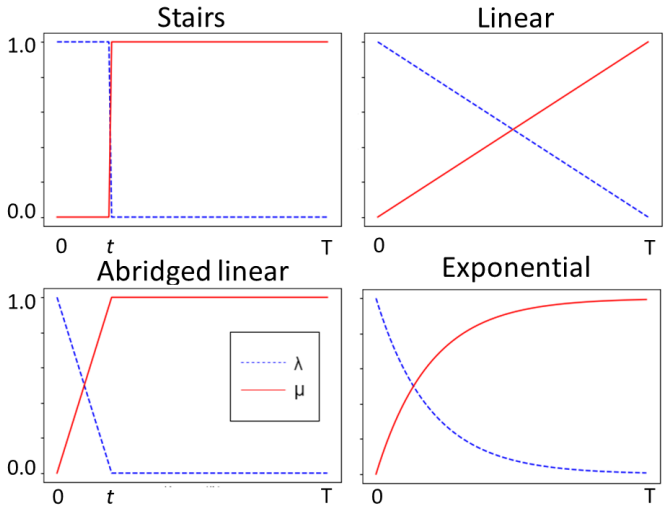


Figure 5.2: Dynamic Evolving Weighting approaches.

during learning. Examples of dynamic approaches are Evolving Weighting (Belharbi et al. 2016), Loss-Balanced Task Weighting (Liu, Liang & Gitter 2019), and Uncertainty Weighting (Kendall et al. 2018), which were previously introduced in Sections 2.8 and 3.9, discussed further below.

Evolving Weighting: Belharbi et al. (2016) proposed to evolve the loss weighting during the training steps according to a *schedule*. A training step is defined as the number of batches of the training data, such that the total number of steps is the number of batches multiplied by the number of training epochs. Four different schedules were proposed. Figure 5.2 gives an overview of how the four schedules vary the weights of the main and auxiliary tasks – μ and λ, respectively – across the training steps. These four schedules are described below:

Stairs schedule: The initial emphasis is on the auxiliary task, with $\mu = 0$ and $\lambda = 1$. At a determined training step t, $\mu = 1$ and $\lambda = 0$.

Linear schedule: The weight of the auxiliary task decreases linearly at each training step, such that the auxiliary weight $\lambda = 1$ tends to 0; in contrast, the weight of the main task increases linearly, i.e. $\mu = (1 - \lambda)$. In particular, given that the total number of steps T is known in advance, $\lambda_t = \frac{t}{\mathrm{T}}$.

Abridged Linear schedule: In a linear schedule, μ rises over the full training schedule to step T. This may not place sufficient emphasis on the main task during training. Instead, in the Abridged Linear schedule the weight on the auxiliary task λ falls linearly to 0 by a threshold step t_τ. After t_τ, all emphasis is on the main task (i.e. $\mu = 1$).

Exponential schedule: The weights evolve exponentially to the step number, i.e. $\mu = exp(\frac{-t}{\sigma})$, where t is the current number of training steps, and σ is the slope parameter, determined as a fraction of the total training steps T, as shown in Figure 5.2. We tune σ using a grid search within the range of 0.1 to 0.5 of T, identifying $\sigma = 0.3T$ as the optimal setting, which effectively balances the learning dynamics throughout the training, optimising model performance.

Loss-Balanced Task Weighting (LBTW) (Liu, Liang & Gitter 2019): This MTL method aims to reduce *negative transfer* by using the task-specific loss to balance the different auxiliary tasks. Negative transfer is when the performance of the task is decreased by Multi-Task Learning compared to the Single-Task Learning (STL). This method employs the *loss ratio* between the current loss and the initial loss of each task to adjust the task's weight. The task with the loss ratio closest to 1 needs to contribute more to the total loss. By increasing the weight of the task with loss ratio that is closest to 1, this method attempts to *balance* the tasks' importances.

Uncertainty Weighting (Kendall et al. 2018): This method is the most often used Multi-Task Learning approach, which is a weighting strategy that consists in analysing the uncertainty of each task. In this method, each of the task's weights is adjusted by deriving a multi-task loss function when maximising the Gaussian likelihood (Ruder 2017).

In the following, we describe our proposed hybrid approach.

Algorithm 1: Hybrid Task Weighting

Given S tasks and parameter α.
Initialise neural network weights W.
for each epoch i **do**
  Store the first batch loss as $\ell_{(0,i)} \in \mathbb{R}^S$
  for each batch of data B **do**
    Get the loss on each task $\ell_B \in \mathbb{R}^S$
    if step $t \leq t_\tau$
      Set the main task weight $\mu = (\frac{t}{T})$
    else
      Set the main task weight $\mu = 1$
    for each auxiliary task s **do**
      Set the auxiliary task weight $\lambda = (\frac{\ell_{(B,s)}}{\ell_{(0,i,s)}})^\alpha$
      Update weighted loss $\ell_{(B,s)} = \ell_{(B,s)} \times \lambda$
    Update weighted loss $\ell_{(B,m)} = \ell_{(B,m)} \times \mu$
    Set the total loss $\ell_{total} = \ell_{(B,m)} + \sum_{i=1}^{s} \ell_i$

5.3.3 Hybrid Task Weighting

Among the existing dynamic MTL methods, Uncertainty Weighting (Kendall et al. 2018), and Loss-Balanced Task Weighting (Liu, Liang & Gitter 2019) both weight all tasks without prioritising on the main task, such that resources are unnecessarily allocated to other tasks, thereby leading to a possible underfitting on the main task (Guo et al. 2018). For this reason, we propose a Hybrid Task Weighting approach, which applies an Abridged Linear schedule for weighting the main Answer Extraction task and LBTW Liu, Liang & Gitter (2019) for weighting the auxiliary tasks, such as Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction.

In particular, for the Abridged Linear schedule, we take a step threshold $t_\tau = \mathrm{T}/10$, i.e. 10% of all steps, which is the same as the warm-up ratio we use (see Section 5.4.5 for further details). To apply LBTW for the auxiliary tasks, a hyperparameter α is used to balance the influence of the task-specific weights, i.e. α=0.5 [2] (Liu, Liang & Gitter 2019). For each batch, the weight of each task is calculated by using the loss ratio between the loss at step t and the loss at t=0, thereby balancing the loss rates of the auxiliary tasks. Algorithm 1 provides further details about the implementation of our hybrid approach. This algorithm details our 'Hybrid Task Weighting' method. It initialises network weights and processes each batch by calculating task-specific losses. The main task weight is adjusted linearly up to a certain step threshold and remains constant thereafter. Auxiliary task weights are dynamically calculated using Loss-Balanced Task Weighting, factoring in the initial and current losses, modulated by a hyperparameter α. The algorithm then combines these weighted losses for the total loss calculation, effectively balancing the contributions of the main and auxiliary tasks throughout the training process.

5.4 Experimental Setup

In this section, we start by outlining our research questions in Section 5.4.1. We then describe the used dataset, QuAC, and its auxiliary tasks in Section 5.4.2. We present the list of our baselines in Section 5.4.3. We discuss the used evaluation metrics in Section 5.4.4, and the applied hyper-parameter settings in Section 5.4.5.

5.4.1 Research Questions

In this chapter, we address three key research questions. Firstly, one of our central contributions is the comparison of existing Multi-Task Learning (MTL) strategies, when used in the same Conversational Question Answering (ConvQA) model both in terms of effectiveness and efficiency. By doing this, we investigate whether there is an actual difference between the static and dynamic loss weighting methods, in guiding the learning process in a ConvQA scenario. Moreover, to the best of our knowledge, there has been no previous study that investigated dynamic loss weighting for the ConvQA task on the QuAC dataset. Hence, our first research question is:

RQ 5.1 What is the most effective and efficient Multi-Task Learning method, including our proposed Hybrid Task Weighting method, for ConvQA?
Secondly, we investigate the effectiveness of the combination of the auxiliary tasks, such as Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction, to improve

[2] Following Liu, Liang & Gitter (2019), in our preliminary experiments comparing $\alpha = 0.1$ and $\alpha = 0.5$ for the Loss-Balanced Task Weighting (LBTW), $\alpha = 0.5$ demonstrated better performance, suggesting a more optimal balance of the task-specific weights at this higher value.

Table 5.3: Statistics of the QuAC datasets

	Train	Dev.	Test	Overall
questions	83,568	7,354	7,353	98,407
dialogs	11,567	1,000	1,002	13,594
unique sections	6,843	1,000	1,002	8,854
questions / dialog	7.2	7.4	7.3	7.2
% yes/no	26.4	22.1	23.4	25.8
% unanswerable	20.2	20.2	20.1	20.2

the performance of the main Answer Extraction task, namely we posit the following research question:

RQ 5.2 Does applying the proposed MTL ConvQA model with our Hybrid Task Weighting method using each of the auxiliary tasks results in effectiveness improvements over learning using only the main Answer Extraction task?
Finally, we examine how the effectiveness of using MTL in the learning process impacts the performance of the auxiliary tasks, as follows:

RQ 5.3 Does our proposed MTL model lead to not only improving the performance of the main Answer Extraction task but also to an improvement in the performance on the auxiliary tasks, such as Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction,?

5.4.2 Dataset

To conduct our evaluation of the MTL methods when integrated into the BERT ConvQA model, we choose QuAC (Choi et al. 2018), a large-scale dataset for ConvQA over passages extracted from Wikipedia articles, as previously described in Section 3.2.2. Unlike other Machine Reading datasets such as SQuAD (Rajpurkar et al. 2018, 2016), this dataset is considered to be a multi-turn dataset where the questions and answers simulate conversations. The main reason for choosing this dataset for our experiments is that it provides not only an Answer Extraction as the main task but it also provides other auxiliary tasks namely, the affirmation (Yes/No prediction) and continuation (Follow up prediction) classification tasks. Moreover, we also observe that if an answer in QuAC is tagged as CANNOTANSWER, then this means that the corresponding question cannot be answered. Hence, using these kinds of answers, we define another Unanswerable prediction task as an additional auxiliary task to use in our MTL method. For further information about the QuAC dataset, we also provide a summary of its statistics in Table 5.3. We describe below each of the used auxiliary tasks:

Yes/No prediction: This task consists of three possible labels: yes, no, neither where yes or no are represented as the sought answer to this question type; otherwise it will be 'neither'. Choi et al. (2018) observed that there were 25.8% of yes/no questions in the QuAC dataset.

Follow up prediction: This classification task consists in predicting the continuation of a given question, and has three possible labels: follow up, maybe follow up, not follow up.

Unanswerable prediction: This task has two possible labels: yes/no allocated by inspecting the answer text associated to each question in the dataset. If the answer text is CANNOTANSWER, the label is yes otherwise it is no. 20.2% of all questions in the QuAC dataset are unanswerable.

5.4.3 Baselines

We use as baselines all MTL methods described in Section 5.3. Specifically, our baselines consist of the Static MTL methods from Section 5.3.1, namely *sum to 1* and *equal to 1*, and the dynamic MTL methods from Section 5.3.2, namely Evolving Weighting (Stair, Linear, Abridged Linear, and Exponential), Loss-Balanced Task Weighting, and Uncertainty Weighting as baselines. In addition, we also include Single-Task Learning as baselines to gauge the effectiveness of Multi-Task Learning as well as our proposed Hybrid Task Weighting method.

5.4.4 Evaluation Metrics

Since we are using the QuAC dataset, we naturally adopt the two evaluation metrics in the corresponding challenge, which consist of the word-level F1, and the human equivalence score (HEQ), both of which were previously introduced and detailed in Chapter 3, Section 3.2.3. The word-level F1, commonly used in Machine Comprehension and in the ConvQA tasks Choi et al. (2018), Rajpurkar et al. (2018, 2016), evaluates the overlap between the system's prediction and the ground truth answer span. Meanwhile, the HEQ metric is used to evaluate the percentage of examples for which the deployed model's F1 is equivalent to or higher than the human F1. This metric is composed of HEQ-Q, computed on the question level, and HEQ-D, computed at the dialogue level. The QuAC challenge defines the human performance to have an HEQ-Q and HEQ-D of 100%. Finally, we use McNemar's test to measure the statistical significance between the prediction performances.

5.4.5 Hyper-parameter Settings

We implement all models using the Pytorch version of BERT from HuggingFace (Wolf et al. 2020), namely using the `bert-base-uncased`[3] model as our encoder. Following Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer (2019), the model configuration is as follows: the max sequence length is set to 12, the stride in the sliding window is set to 128, the max question length is set to 64, the max answer length set to 35, the number of training epochs is set to 5 and the batch size is set to 12. To train our BERT ConvQA model, we use the BertAdam weight decay optimiser, with

[3] https://huggingface.co/transformers/pretrained_models.html

Table 5.4: Effectiveness of various task-weighting methods for Conversational Question Answering (ConvQA). † denotes a result statistically different from that of our proposed Hybrid Task Weighting model (McNemar's test, $p < 0.05$); ‡ denotes a significant improvement over the STL baseline. The highest value for each measure (row) is highlighted.

	Single-task learning	Static task weighting		Evolving Weighting				Loss-Balanced Task Weighting	Uncertainty Weighting	Hybrid Task Weighting
		Sum to 1	Eq.1	Linear	Exponential	Abridged Linear	Stair			
F1	69.08 †	69.56 †	69.40 †	70.97 †‡	71.16 †‡	71.37 †‡	69.73 †	69.41 †‡	69.20 †	**72.28** ‡
HEQ-Q	65.51	66.06	65.65	67.49	67.88	67.78	66.24	65.71	65.43	**68.71**
HEQ-D	11.6	11.10	10.7	12.8	**13.1**	12.10	11.80	11.6	11.3	13.00

Table 5.5: Efficiency of different MTL methods. The highest value for each training and evaluating phase is highlighted.

Model		#iters/sec	
		Training	Evaluating
Single-task learning		2.23	3.95
Static task weighting	Sum to 1.	2.13	4.05
	All eq. 1	2.25	**4.12**
Evolving Weighting	Linear	**2.31**	3.99
	Exponential	2.26	3.90
	Abridged Linear	2.20	3.84
	Stair	2.15	4.07
Loss-Balanced Task Weighting		2.05	3.91
Uncertainty Weighting		1.5	3.94
Our propose $ORConvQA_{1:dynamicMTL}$ Hybrid MTL method		2.04	4.04

an initial learning rate of 5e-5 while the learning rate warming up portion is 10%. For all our experiments, we use a single Nvidia TITAN RTX GPU.

5.5 Results Analysis

We first report our evaluation results for various MTL methods using our ConvQA model in Section 5.5.1. Our findings for the usefulness of the auxiliary tasks in MTL are detailed in Section 5.5.2. In Section 5.5.3, we examine if our proposed $ORConvQA_{1:dynamicMTL}$ Hybrid MTL method (see Section 5.3.3) enhances the performance of the system on the auxiliary tasks.

5.5.1 RQ 5.1: Effectiveness and Efficiency of the MTL Methods

We first investigate the performance of the baselines in comparison to our proposed $ORConvQA_1$ hybrid MTL method for Multi-Task Learning on the validation set[4] of the QuAC dataset. All MTL methods are trained on the provided QuAC training set by using all the auxiliary tasks, namely the Yes/No prediction, the Follow up prediction and the Unanswerable prediction classification

[4] The QuAC's test set is only accessible by submitting to its leaderboard. Hence, we provide results using the provided validation set.

tasks. In this section, we focus on the performance of the system on the main task (i.e. the Answer Extraction task).

First, we examine the effectiveness of the MTL methods, including our proposed $ORConvQA_1$ hybrid MTL method and those baselines listed in Section 5.3. Table 5.4 shows the single-task learning baseline (denoted STL) in the first column and the MTL methods in the following columns. Within Table 5.4, the best result in each row is highlighted in bold. From this table, we observe that the F1 performance of all the MTL methods is better than the STL baseline. In fact, our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method method achieves the best F1 and HEQ-Q performances, at 72.28 and 68.71, respectively. The best reported HEQ-D score is achieved by the Exponential Evolving Weighting method at 13.1 followed by our $ORConvQA_{1:dynamicMTL}$ hybrid MTL method at 13.0. Indeed, our proposed method is more effective than the Abridged Linear and the Loss-Balanced Task Weighting dynamic methods, showing that while it emphasises the main task (c.f. Abridged Linear), it also balances the auxiliary tasks through the use of the LBTW method. Moreover, all of the dynamic task weighting methods significantly outperform the STL model, except for the Stair and Uncertainty Weighting methods (McNemar's test, $p < 0.05$).

Next, we investigate the efficiency of the tested MTL methods by comparing the average number of iterations per second needed during training and evaluation. Table 5.5 depicts the efficiency of the MTL methods for the BERT model. In this table, the higher the number, the higher the efficiency, while the best result is highlighted in bold. We observe that the Linear Evolving Weighting method yields the best efficiency in comparison to all other methods – at 2.31 iterations per second during learning – while the static task weighting method (equal to 1) exhibits the best evaluation efficiency. In addition, our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method shows efficiency as expected in both the training and evaluating phases. Although it doesn't lead in the training phase, it shows competitive efficiency during evaluation, suggesting a well-balanced approach to managing the complexities of multi-task learning.

Overall the efficiency of most models during evaluation is fairly similar, at around 3.8 to 4.1 iterations per second. We argue that this is because during the evaluation phase, all models have the same structure, and only differ in terms of weights. On the other hand, during learning, the Evolving Weighting method is slightly faster than the other baseline methods including our own proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method due to the simple manner in which it calculates the task weight. Moreover, training the ConvQA model using the Uncertainty Weighting method exhibits more training time than other methods. Indeed, this approach has the most complex implementation.

In response to RQ 5.1, we find that our BERT ConvQA model learned through Multi-Task Learning by using a hybrid approach has the best effectiveness, yielding statistically significant improvement over the baselines. Moreover, we observe that there is little difference between the efficiency of our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method, and that of the static task weighting methods, or the Single-Task Learning in both the training and evaluation phases

Table 5.6: Comparison of different combinations of auxiliary tasks. † denotes a statistically significant improvement over STL with $p < 0.05$ using the McNemar's test. The highest value for each measure is highlighted.

	F1	HEQ-Q	HEQ-D
STL	69.08	65.51	11.6
Yes/No	68.60	64.98	11.6
Follow-up	69.76 †	66.74	12.0
Unanswerable	72.27 †	68.87	13.5
Yes/No + Follow up	**72.66** †	68.87	11.7
Yes/No + Unanswerable	72.48 †	**69.02**	13.2
Follow-up + Unanswerable	71.91 †	68.52	**14.2**
All	72.28 †	68.71	13.0

even though our approach has a more complex implementation.

5.5.2 RQ 5.2: Combination of Auxiliary Tasks vs. Single-Task Learning

Next, we conduct experiments to determine the best combination of auxiliary tasks, which helps to improve the performance of the main Answer Extraction task. In these experiments, all models are learned by using our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method as the Multi-Task Learning strategy for the BERT ConvQA model. We vary the choice of auxiliary tasks from those detailed in Section 5.4.2, namely Yes/No prediction, Follow-up question identification and Unanswerable prediction. Single-Task Learning (STL) acts as a baseline for these experiments.

Table 5.6 presents the effectiveness of the different combinations of auxiliary tasks (each row is a different combination). We observe that the highest scores for the F1, HEQ-Q and HEQ-D measures are not obtained from the same combination. In particular, applying Multi-Task Learning using the Yes/No and Follow up tasks achieves the best F1 performance compared to the other combinations. However, when using the HEQ-Q metric, it is apparent that the combination of the Yes/No prediction and Unanswerable prediction is the best. Furthermore, the combination of the Follow up prediction and Unanswerable prediction yields the best model in terms of the HEQ-D metric. From these results, we further analyse why the models that include Unanswerable prediction as one of the auxiliary tasks, have higher HEQ scores in comparison to models that use either the Yes/No prediction or the Follow up prediction as the auxiliary tasks.

We found that a key issue is the number of correct predictions for the unanswerable questions. The more correct answers achieved by the MTL models on this type of questions, the more likely the performance will be higher in terms of HEQ. From Table 5.6, we also observe that the model that fused all the auxiliary tasks is not the best choice for MTL, and its performance on all metrics is similar to the model that used only the Unanswerable prediction auxiliary task.

In answer to RQ 5.2, we conclude that most of the combination models are better than just learning the main task, except the model that solely used the Yes/No prediction as an auxiliary

Table 5.7: Evaluation results of the auxiliary tasks on different MTL methods. † denotes statistically significant differences between the STL model and the indicated model (McNemar's test, $p < 0.05$). The highest value for each auxiliary task is highlighted.

Model		Accuracy of auxiliary tasks		
		Yes/No	Follow up	Unanswerable
STL		**71.20**	**58.07**	72.34
Static MTL	Sum to 1	68.08 †	57.43	76.38
	All eq. 1	69.10 †	57.61	73.08
Evolving	Linear	68.10 †	57.49	76.92
	Exponential	65.51 †	56.84	79.81
	Abridged Linear	50.80 †	52.48	**80.15**
	Stair	49.32 †	46.89 †	79.41
Loss-Balance Task Weighting		67.05 †	57.31	76.31
Uncertainty		68.95 †	57.75	72.68
Our propose $ORConvQA_{1:dynamicMTL}$ hybrid method		68.63 †	57.61	76.45

task. This raises the question as to why the model that combines all the auxiliary tasks does not outperform the models that include Unanswerable as an auxiliary task. We conjecture that negative transfer (see Section 5.3.2) might be a possible reason explaining the drop in the performance of MTL. We leave the investigation of this issue to future work.

5.5.3 RQ 5.3: Our $ORConvQA_{1:dynamicMTL}$ hybrid MTL Method Performances on The Auxiliary Tasks?

In this section, we examine the performance of the auxiliary tasks after training in a multi-task setting. To conduct this experiment, we also compare all baselines mentioned in Section 5.4.3 to determine whether the MTL strategies help to improve the accuracy of the classifiers on the auxiliary tasks. Table 5.7 shows, for each auxiliary task, the Accuracy values[5] obtained using single-task learning (STL) – learned for each task separately – and the MTL methods. In this table, the best results for each task are highlighted in bold. We observe that, for the Yes/No prediction tasks, all of the models trained by the MTL methods exhibit significantly degraded performance than when training using STL; for the Follow up and Unanswerable tasks, all of the MTL models excluding the Stair Evolving Weighting method are statistically indistinguishable. This is intuitive as the STL model is trained specifically for that task. However, for the Unanswerable task, Abridged Linear achieves the best result.

Moreover, the lowest accuracy in the Yes/No prediction and Follow up prediction is associated to the Stair Evolving Weighting method. We postulate that this is because learning with the Stair Evolving Weighting method after step t, the $\mathcal{L}_{total}$ in Equation (5.5) is derived only from $\mathcal{L}_{ans}$, and hence it does not learn to address the auxiliary tasks after that step t. We also observe that

[5] We follow the authors of the QuAC (Choi et al. 2018) dataset in applying Accuracy for measuring the performance on the auxiliary classification tasks.

the accuracy of all tasks achieved by the model learned from Loss-Balanced Task Weighting (LBTW) and our proposed method are similar, likely because we leverage the LBTW method to learn the auxiliary tasks in our proposed model.

Overall, in answer to RQ 5.3, we conclude that our method cannot enhance the performance of the auxiliary tasks. We conjecture that a possible reason for this result is that there might be an occurrence of a negative transfer (see Section 5.3.2) during learning, resulting in a drop in the performance of the Multi-Task Learning on the auxiliary tasks. We leave investigating the contrasts between the impact of MTL on the three auxiliary tasks to future work.

5.6 Conclusions

In this chapter, we proposed a $ORConvQA_{1:dynamicMTL}$ method (as described in Section 4.4.1) for Conversational Question Answering (ConvQA), which learns to extract the correct answer, by applying Multi-Task Learning (MTL). Our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method makes use of Evolving Weighting by Abridged Linear for learning the main task, while the auxiliary tasks are addressed using Loss-Balanced Task Weighting. Our proposed $ORConvQA_{1:dynamicMTL}$ method directly addresses Gap 1 (as identified in Section 3.10), where we stated that the current ConvQA approaches using MTL lacks dynamic adjustment of task importance during learning. In particular, our investigation aims to answer three research questions as follows:

First, to answer RQ 5.1, "What is the most effective and efficient Multi-Task Learning method for ConvQA?", we investigated the performance of the baselines in comparison to our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method focusing on both effectiveness and efficiency. The results in Table 5.4 showed that our $ORConvQA_{1:dynamicMTL}$ model learned through Multi-Task Learning by using our proposed hybrid method has the best effectiveness, yielding statistically significant improvements over the baselines. Whereas, the results in Table 5.5, the efficiency of our proposed method is comparable to that of the static task weighting methods. In both the training and evaluating phases, our proposed $ORConvQA_{1:dynamicMTL}$ hybrid MTL method shows similar efficiency to other methods, even though it is more complex. This shows that our approach improves ConvQA tasks effectively without losing efficiency.

In order to answer RQ 5.2, "Does applying the MTL ConvQA model with our Hybrid Task Weighting method using each of the auxiliary tasks result in effectiveness improvements over learning using only the main Answer Extraction task?", our experiments aimed to identify the most effective combination of auxiliary tasks. As shown in Table 5.6, we found that models combining multiple auxiliary tasks outperformed the model learning only the main Answer Extraction task, except the model that solely used the Yes/No prediction as an auxiliary task. This raises the question as to why the model that combines all the auxiliary tasks does not outperform the models that includes Unanswerable as an auxiliary task. We conjecture that negative transfer,

as mentioned in Section 5.3.2, might be a possible reason explaining the drop in the performance of MTL. We leave the investigation of this issue to future work.

Finally, to response RQ 5.3, in Section 5.5.3, we examined the performance of the auxiliary tasks after training in a multi-task setting. To conduct this experiment, we also compare all baselines mentioned in Section 5.4.3 to determine whether the MTL strategies help to improve the accuracy of the classifiers on the auxiliary tasks. In particular, we concluded that our method cannot enhance the performance of the auxiliary tasks. We conjecture that a possible reason for this result is that there might an occurrence of a negative transfer (see Section 5.3.2) during learning resulting in a drop in the performance of the Multi-Task Learning on the auxiliary tasks. We leave investigating the contrasts between the impact of MTL on the three auxiliary tasks to future work.

Hence, in conclusion to the hypothesis presented in Section 1.3, namely that a dynamic Multi-Task Learning approach that simultaneously trains the main task of answer extraction, along with auxiliary tasks such as Follow-up Question Identification, Yes/No prediction, and Unanswerable prediction, can enhance the system's effectiveness for Conversational Question Answering. Gap 1 highlighted the lack of dynamic adjustment of task importance during learning in existing ConvQA approaches using MTL. Our findings in this chapter concluded that our $ORConvQA_{1:dynamicMTL}$ method learned through Multi-Task Learning by using our proposed Hybrid method effectively addresses Gap 1. Our $ORConvQA_{1:dynamicMTL}$ hybrid MTL method not only incorporates dynamic task weighting adjustment but also yields statistically significant improvements in the effectiveness of the main Answer Extraction task over baseline methods, including Single-Task Learning, static MTL, and dynamic MTL.

In Chapter 6, we conduct an investigation to combine the tasks of the follow-up question identification and the conversational question rewriting to more effectively identify ambiguous questions.

Chapter 6

Multi-Task Learning of Question Rewriting and Follow-up Question Identification

6.1 Introduction

In Chapter 5, one of our hypotheses in Section 1.3, showing that a dynamic Multi-Task Learning (MTL) approach can indeed enhance Conversational Question Answering (ConvQA). In this chapter, we investigate whether by combining Follow-up Question Identification and Conversational Question Rewriting, the system's response accuracy and relevance can be enhanced. This chapter aims to test this hypothesis from Section1.3 by incorporating these two tasks into a single text generation model. Indeed, we argue that by identifying connections between the user's questions, addressing ambiguities, and leveraging the conversation's context, the system can refine its understanding of the user's intent and can provide more precise and relevant responses. As a result, in Section 4.4.2, we introduced our proposed $ORConvQA_{2:FID+QR}$ method, which leverages the Multi-Task Learning (MTL) to simultaneously address the challenges of understanding the context of a conversation, while also reformulating the user's questions for more effective information retrieval.

In Section 1.2, we discussed the challenges of answering user questions in ORConvQA, which mimics how humans seek information in the form of a dialogue. In order to effectively answer the user's questions, an ORConvQA system needs to resolve any ambiguities arising from the posed questions based on the conversation history with the user (Kundu et al. 2020). To address the ambiguity of the conversational questions, we have explored many approaches, such as Follow-up Question Identification and Conversational Question Rewriting, as previously described in Sections 3.3.4 and 3.4.4. We also identified Gap 2 in Section 3.10, which stated the lack of effective integration between these two tasks. In this chapter, we investigate the combination of both the Follow-up Question Identification task and the Conversational Question Rewriting task to improve the effectiveness of both tasks.

As introduced in Section 2.8, Multi-Task Learning (MTL), which is a method of learning several different but related tasks at the same time, has become a popular approach for tackling multiple tasks in a uniform model (Qu, Yang, Qiu, Zhang, Chen, Croft & Iyyer 2019). MTL can also be used to increase a system's performance on the text generation task by leveraging classification tasks. For example, Ide & Kawahara (2021) adopted an MTL approach using a text generation BART model (see Section 2.3.1), which jointly learns a classification task and a text generation task by sharing the learner, and showed an improvement in an emotion-aware dialogue response generation model. We also adopted the MTL approach using the BART model (Ide & Kawahara 2021) in our proposed models. On the other hand, a text generation T5 model (see Section 2.3.1) has been shown to be usable for multi-task learning both the text generation and classification tasks such as passage ranking/re-ranking and answer generation/extraction (Jiang et al. 2022*b*, Lee et al. 2022*b*). Due to the good effectiveness of text generation approaches on various tasks, in this chapter, we argue that text generation models – BART and T5 – can be adapted for the Follow-up Question Identification and Conversational Question Rewriting tasks.

Therefore, in this chapter, we propose a $ORConvQA_{2:FID+QR}$ Multi-Task Learning (MTL) method that uses a text generation model for both question rewriting and classification. Our models, based on BART and T5, are trained to rewrite conversational questions and identify follow-up questions simultaneously.

Our contributions are as follows:

(1) We leverage Multi-Task Learning with a text generation model to effectively address the tasks of Follow-up Question Identification and Conversational Question Rewriting;

(2) Using the recent LIF dataset (c.f. Section 3.3.2), we compare our models to two recent baselines from the literature, and show that our Multi-Task Learning BART model yields the best F1 and Macro-F1 performance improvements over the strongest baseline, the three-way attentive pooling model, with statistically significant improvements ranging from 3.5% to 10.5% on all LIF test sets;

(3) Our proposed Multi-Task Learning T5 model significantly outperforms the single-task learning of question rewriting models for passage retrieval on the OR-QuAC test set (Section 3.7.2).

The rest of the chapter is structured as follows: Section 6.2 states the definitions of the Follow-up Question Identification and Conversational Question Rewriting tasks, along with our proposed $ORConvQA_{2:FID+QR}$ method (c.f. Section 4.4.2) to address these tasks simultaneously. We present our experimental setup in Section 6.3 and show the results of the experiments in Section 6.4. Finally, we provide concluding remarks in Section 6.5.

Table 6.1: Notations used in Chapter 6.

Notation	Definition
q_k	A current question
q'_k	A rewritten question of the current question q_k
a_k	An answer to current question q_k
H_k	A conversation history i.e. $H_k = [\langle q, a \rangle]$
P^+_{ret}	A retrieved passage
C	A passage collection
W	A sequence of m words i.e. $W = \{w_1, w_2, ..., w_m\}$
$\hat{T}_k$	A contextualised token-level representations
$FID(\cdot)$	A Follow-up Question Identification function
$QR(\cdot)$	A Conversational Question Rewriting function
$Retriever$	A passage retriever function
$Reranker$	A passage reranker function

6.2 A MTL Model for Classification and Question Rewriting

In this section, we describe our proposed Open-Retrieval Conversational Question Answering (ORConvQA) models, which are based on the BART and T5 architectures. These models are instantiations of the $ORConvQA_{2:FID+QR}$ method outlined in Section 4.4.2. We explain how these models are used to perform the Follow-up Question Identification (Section 3.3) task as well as the Conversational Question Rewriting (Section 3.4) task. In Section 6.2.1, we first recall the definitions of the Follow-up Question Identification (Section 3.3.1) and Conversational Question Rewriting (Section 3.4.1) tasks. An overview of the proposed text generation model follows in Section 6.2.2.

6.2.1 Task Definitions

In this chapter, we aim to tackle the tasks of Follow-up Question Identification (FID) (see Sections 3.3.1) and Conversational Question Rewriting (QR) (see Section 3.4.1). Table 6.1 presents the notations used in this chapter, which include a subset of the symbols defined in Chapter 4, Table 4.1. The FID and QR tasks were formalised in Equations (4.1) and (4.2) of Chapter 4 as follows:

$$\begin{aligned} FID(q_k, H_k) &\rightarrow (valid/invalid) \\ QR(q_k, H_k) &\rightarrow q'_k \end{aligned} \tag{6.1}$$

where $FID(\cdot)$ is a function that predicts whether or not the candidate follow-up question q_k is a valid follow-up question, while $QR(\cdot)$ is a function that reformulates the user's question q_k in a more clear and precise manner. The question rewriting function leverages the context H_k from the ongoing conversation to generate a rewritten question q'_k. Note that the FID task can be formulated as a classification task while QR is a text generation task. In the following section, we describe our proposed models to address these tasks.

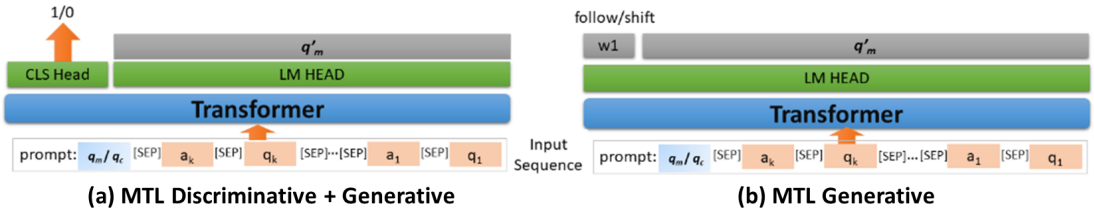


Figure 6.1: A schematic comparison for MTL models for (a) a discriminative+generative model prediction by applying a CLS head to create a score for a classification task and an LM head to generate the tokens for a Question Rewriting task (see Section 3.4.4) and (b) a generative model prediction by generating the first token for a classification task and the follow-up tokens for a questing rewriting task.

6.2.2 Models Overview

To tackle the tasks described in Section 6.2.1, we propose classification and question rewriting models that leverage historical questions to identify whether a candidate question q_k is a follow-up to the previous question and to reformulate the current question q_k. Our proposed $ORConvQA_{2:FID+QR}$ method in Chapter 4 can use models, such as BART (Section 2.3.1) and T5 (Section 2.3.1), which are large pre-trained language models designed for text generation, as described in Section 2.3.1. In particular, the text generation approaches can be trained to generate a meaningful textual response based on some input text. Moreover, like BERT, the pre-trained BART and T5 models can be fine-tuned to perform a variety of downstream tasks.

In addition, the manner in which a text generation model is used in the classification tasks can differ, as they can be fine-tuned as discriminative or generative models. In a discriminative setup, the model is adapted for binary classification by adding a fully-connected layer with two output neurons (corresponding to each class) upon a special [CLS] token in BERT, or the last token in BART. In contrast, a generative setup reframes natural language processing (NLP) tasks as text generation tasks - for instance, classification is performed by examining what text is generated and the corresponding likelihood.

To adopt an MTL approach to a text generation model for jointly learning from both the classification and question rewriting tasks, the MTL model can be used in either a discriminative+generative or in a generative setup as shown in Figure 6.1. A discriminative+generative MTL model makes predictions by applying a CLS head to create a score for a classification task and an LM head to generate the tokens for a question rewriting task (Section 3.4.4). On the other hand, a generative MTL model makes predictions by generating the first token for a classification task and the follow-up tokens for a query rewriting task. Furthermore, text generation models such as BART can be used as either discriminative+generative or generative MTL models. On the other hand, the T5 model can only be used as a generative MTL model (Raffel et al. 2020).

In particular, when fine-tuning the T5 model for a downstream task, a *prefix text* is required – for example "translate English to German:" might be used for a translation task. Indeed, as mentioned in Section 2.3.1, the text generation models have been shown to achieve state-of-the-art

performances in classification (Lewis et al. 2019, Raffel et al. 2020), as well as in document re-ranking – by ranking based on the likelihood of generating a particular token (dos Santos et al. 2020, Nogueira, Jiang & Lin 2020) – (outperforming the BERT models) and even in arithmetic tasks (Nogueira et al. 2021). Hence, for the Multi-Task Learning of both the Follow-up Question Identification (FID) and Conversational Question Rewriting (QR) tasks, we choose the MTL generative version of the BART and T5 models. However, for comparison purposes, we also deploy the discriminative+generative versions of the BART models in our experiments.

More precisely, we deploy generative models to capture the relation between the question q_k and the contextual information in the conversation history, including the historical question(s) $\{q_1, q_2, \ldots q_k\}$, and the historical answer(s) $\{a_1, a_2, \ldots a_k\}$, as shown in Figure 6.1(b). In particular, as defined in Equation (4.14) of Chapter 4, let $MTL(\cdot)$ denotes a joint learning function of the tasks defined in Equation (6.1) as follows:

$$MTL(FID, QR, \theta) \rightarrow w_1, w_2, ..., w_m \tag{6.2}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to generate the target tokens length m as shown in Equation (6.2), where the token w_1 is either “follow" or “shift"[1] depending on whether the candidate question is a valid follow-up to the previous question or not, while the follow-up tokens $w_2, ..., w_m$ are the output sequence for the target query q'_k.

Recall from Section 4.4.2 that we introduced the $ORConvQA_{2:FID+QR}$ method, which can be defined using Equation 6.2 as follows:

$$\begin{aligned} ORConvQA_{2:FID+QR} = MTL(FID, QR, \theta) \\ \overset{q'_k}{\gg} Retriever(C) \\ \overset{P^+_{ret}}{\gg} Reranker(q'_k) \end{aligned} \tag{6.3}$$

At inference time, following Nogueira, Jiang, Pradeep & Lin (2020*b*), we apply a softmax only on the logits of the "follow" and "shift" tokens of the first generated token w_1, to determine the probability of follow-up question as follows:

$$score^{rr} = \text{softmax}(w_1) \tag{6.4}$$

Next, we evaluate our MTL $ORConvQA_{2:FID+QR}$ method and its resulting system in comparison to several existing baselines as detailed in the next section.

[1] We choose “follow" and “shift" as target tokens because T5 tokenises sequences using the SentencePiece approach (Kudo & Richardson 2018), which splits the word “invalid" into two subwords.

Table 6.2: Statistics of the used datasets

	LIF + CANARD			LIF		OR-QuAC
	Train	Dev	Test-I	Test-II	Test-III	Test
#questions	62,839	4914	5,992	5,247	2,685	5571
#valid follow-up	22,056	1,559	1,940	1,940	1,940	-
#invalid follow-up	40,783	3,355	4,052	3,307	745	-

6.3 Experimental Setup

We define our research questions in Section 6.3.1, describe the used datasets in Section 6.3.2 and present our baselines in Section 6.3.3

6.3.1 Research Questions

We address two key research questions. Firstly, we aim to compare our Multi-Task Learning (MTL) method for the text generation models (BART and T5) with the existing strongest follow-up question prediction baseline models according to (Kundu et al. 2020), namely the three-way attentive pooling model and BERT. Indeed, a key contribution of our work is the comparison of the effectiveness of MTL on text generation approaches in the follow-up question prediction task. To the best of our knowledge, our work is the first study to investigate applying MTL for text generation models on the Follow-up Question Identification task. Hence, our first research question is:

RQ 6.1: Does the MTL approach of question rewriting and classification using text generation models outperform the single-task learning (STL) of text generation models and existing baselines for the follow-up question identification task?

Second, another key contribution of our work is the comparison of the effectiveness of MTL on text generation approaches in the Conversational Question Rewriting task. By doing this, we investigate the effectiveness of the rewritten query in improving the conversational question rewriting and passage retrieval systems' performance.

RQ 6.2: Does the MTL approach of question rewriting and classification using text generation models outperform the single-task learning (STL) of text generation models in the context of the conversational question rewriting and passage retrieval tasks?

6.3.2 Datasets

We experiment using the LIF dataset (Kundu et al. 2020) and the CANARD dataset (Elgohary et al. 2019), which were previously described in Sections 3.3.2 and 3.4.2, respectively. For training the models to address both tasks, in the training and development sets, we integrate LIF and CANARD by selecting only the candidate questions from the LIF dataset that exist in the

CANARD dataset. To evaluate the models in the follow-up question identification task, we use the three test sets of the LIF dataset, namely Test-I, Test-II, and Test-III. In all three test sets, the valid follow-up questions (label = 1) are constructed from the "should ask follow-up question" instances in the QuAC dataset (see Section 3.2.2). For the invalid follow-up questions (label = 0), Test-I combines the invalid instances from Test-II & Test-III. For Test-II, questions with a high similarity to the current passage are sampled from other conversations. On the other hand, for Test-III, the invalid follow-up questions are sampled from the non-follow-up questions of the same conversation in QuAC. For the question rewriting task, we use the test sets of the OR-QuAC dataset (see Section 3.7.2). We also aggregate a passage collection from the OR-QuAC dataset, which is an aggregation of three existing datasets consisting of QuAC (Section 3.2.2), CANARD (Section 3.4.2), and English Wikipedia, to evaluate the passage retrieval performance of the conversational question rewriting task. This allows us to assess our models' performance across both the Conversational Question Answering and Passage Retrieval tasks. For further information about the used datasets, we provide a summary of their statistics in Table 6.2.

6.3.3 Baselines and Implementation Details

Follow-up question identification task: We only include neural models as our baselines since the existing rule-based and statistical machine learning models have been shown to be much less effective in the Follow-up Question Identification task in a previous study (Kundu et al. 2020). Indeed, as baselines, we choose the strongest baseline in the previous study (Kundu et al. 2020) (see Section 3.3.4), namely *BERT* (Section 2.3.2), as well as the existing state-of-the-art (SOTA) *three-way attentive pooling* model (Section 3.3.4) from the same study. For the three-way attentive pooling model, we reproduce the model and its evaluation results provided by (Kundu et al. 2020). We additionally compare our MTL of the generative BART and T5 models with the *Single-Task Learning (STL) of T5 and BART*. The STL models are only learned to predict whether a given question is a follow-up question. We additionally compare our generative BART model with the *discriminative+generative* version of BART, as described in Section 6.2.2.

Conversational question rewriting task: We compare our query rewriting methods with the following models: *Raw:* The user's original current question; *Manual:* The questions are written by humans from the CANARD dataset. We also compare our proposed $ORConvQA_{2:FID+QR}$ MTL method with the *Single-Task Learning (STL) of BART and T5*, which are learned to only generate the rewritten question q'_m.

Hyperparameter settings: We implement the BERT, GPT-2, BART, and T5 models using the following PyTorch models from HuggingFace (Wolf et al. 2020), namely `bert-base`, `facebook/bart-base`, and `ramsrigouthamg/t5_paraphraser`:[2] These models are

[2] Initial experiments found this T5 model to be more effective than the original `t5-base` across a number of tasks.

Table 6.3: Results for Follow-up Question Identification. † denotes a performance significantly worse than the MTL BART (McNemar's test, $p < 0.05$); ‡ denotes a performance significantly worse than the MTL T5 (McNemar's test, $p < 0.05$). 3-way AP denotes the Three-Way Attentive Pooling. (dis), (gen), and (dis+gen) denote the discriminative, generative, and discriminative+generative models, respectively (see Section 6.2). The highest value for each measure is highlighted.

Models		Test-I				Test-II				Test-III			
		P	R	F1	Macro-F1	P	R	F1	Macro-F1	P	R	F1	Macro-F1
STL	BERT	70.7	79.5	74.9†‡	80.8†‡	85.6	79.5	82.5†	86.4†	80.2	79.5	79.9†	64.2†‡
	3-way AP	**71.6**	70.0	71.6†‡	79.2†‡	74.4	76.8	75.6†‡	80.5†‡	79.7	70.0	74.6†‡	60.4†‡
	BART(dis)	69.7	79.4	74.2†‡	80.3†‡	85.7	79.4	82.5†‡	86.4†	78.9	79.4	79.1†	62.1†‡
	BART(gen)	71.0	79.7	75.1†‡	81.0†	87.4	79.7	83.4†	87.1†	79.1	79.7	79.4†	62.5†‡
	T5	69.9	83.0	75.9†	81.4†	85.3	83.0	84.2†	87.5†	79.5	83.0	81.2†	64.0†
MTL	BART (dis+gen)	71.5	84.6	77.5	82.6	**86.3**	84.6	85.4	88.5	80.6	84.6	82.5	66.4
ORConvQA$_{2:FID+QR}$	BART (gen)	70.3	**87.3**	**77.9**	**82.7**	84.9	**87.3**	**86.1**	**88.9**	80.4	**87.3**	**83.7**	66.9
	T5	71.3	83.5	76.9†	82.2	85.4	81.6	83.5†	87.1†	**84.6**	77.6	80.9†	**68.9**

configured as follows:[3] the maximum sequence length is set to 512, the number of training epochs is set to 5, the batch size is set to 24, and we use Adam optimiser with a learning rate of 0.00005. For text generation, we use a beam search with a beam width of 5.

Evaluation metrics: Since the Follow-up Question Identification task is a binary classification task, we evaluate performances using classical classification metrics, namely Precision, Recall, F1 and Macro-F1. These evaluation metrics have been previously introduced and detailed in Section 3.3.3. Indeed, following Kundu et al. (2020), reporting Macro-F1 enables accuracy on topic shift detection to be measured, while F1 focuses solely on follow-up identification as the positive class. We use McNemar's test to measure statistically significant differences between the models' classification performances. For the evaluation of the Conversational Question Rewriting performance, we adopt the ROUGE recall calculated for unigrams (ROUGE-1 recall) and the BLEU metrics, as previously described in Section 3.4.3. As for the passage retrieval evaluation, we use Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (NDCG) and Recall@1000 as metrics, which were explained in Section 2.7. For each query, the top 1000 documents are considered. We use the paired t-test for testing significant differences.

Passage retrieval pipeline: We use the PyTerrier (Macdonald & Tonellotto 2020) platform (see Section 2.5) for indexing and retrieving passages. For passage ranking, we incorporate BM25 and DPH with the monoT5 re-ranker (see Section 2.4.1).

6.4 Results Analysis

We now address RQ 6.1 and RQ 6.2, and conclude with a qualitative analysis.

6.4.1 RQ 6.1: Effectiveness on Follow-up Question Identification Task

We investigate the performance of the baselines mentioned in Section 6.3.3 in comparison to our proposed *ORConvQA*$_{2:FID+QR}$ MTL method for follow-up question classification on all three test sets of the LIF dataset. Table 6.3 shows the results for each evaluated model across each of the three test sets.

Comparison of Our Proposed *ORConvQA*$_{2:FID+QR}$ MTL Method Using BART and T5 Models with Baselines for Follow-up Question Identification: From Table 6.3, on the three test sets of the LIF dataset, we observe that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, achieves the highest Recall, F1, and Macro-F1, except our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, on Test-III for Macro-F1. The best precision scores on all three test sets are obtained by the three-way attentive pooling, discriminative+generative MTL BART, and our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, respectively. Within the table, on all three test sets, our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, significantly outperforms the baselines, BERT, the three-way attentive pooling model, STL BART (both discriminative and generative), and STL T5 in terms of F1 and Macro F1, except our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, in Test-III for Macro-F1. These results suggest that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, exhibits a strong overall performance and exceeds other models in terms of Recall, F1 score, and Macro-F1 score across most test sets. This highlights its ability to accurately predict true positive instances (*valid* follow-up question) while maintaining a good balance between precision and recall.

Comparison of Our Proposed *ORConvQA*$_{2:FID+QR}$ MTL Method, using BART and T5 Models, with STL Baselines: We observe that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, significantly outperforms the STL BART (both discriminative and generative) models in terms of F1 and Macro-F1 on all three test sets. This indicates that the MTL approach, which jointly trains the model on the follow-up question identification and conversational question rewriting tasks, provides a notable advantage over the STL approach for the BART classifier model. However, our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, does not outperform the STL T5 model in terms of F1 and Macro-F1, but the two models are not significantly different.

Comparison Between Our Generative MTL Models: We observe that our proposed *ORConvQA*$_2$ MTL method, using BART model, significantly outperforms our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, in terms of F1 on all three test sets, and also significantly outperforms our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, in terms of Macro-F1 on Test-II. Comparing the discriminative+generative and generative MTL BART models, we find

[3] Settings follow `https://github.com/gmihaila/ml_things/`

Table 6.4: Comparison between the MTL models and the query rewriting baselines. † denotes a performance that is significantly worse than the MTL BART model (paired t-test, $p < 0.05$); ‡ denotes a performance that is significantly worse than the MTL T5 model (paired t-test, $p < 0.05$). 3-way AP denotes the Three-Way Attentive Pooling. (gen) and (dis+gen) denote the generative, and discriminative+generative models, respectively (see Section 6.2). The highest value for each measure is highlighted.

Models		Question Rewriting		First Stage (BM25)				Re-ranker (monoT5)		
		ROUGE-1	BLEU	MAP	MRR	R@1000	NDCG	MAP	MRR	NDCG
Raw		62.82†‡	36.01†‡	0.0410†‡	0.0424†‡	0.2335†‡	0.0733†‡	0.0786†‡	0.0809†‡	0.1059†‡
STL	BART	72.91†‡	48.15†	0.1517†‡	0.1617†‡	0.5576†‡	0.2257†‡	0.2438†‡	0.2580†‡	0.3046†‡
	T5	73.22‡	45.86†	0.1720‡	0.1843‡	0.6055‡	0.2524‡	0.2783‡	0.2957‡	0.3430‡
MTL	BART(dis+gen)	73.12‡	48.00†	0.1571‡	0.1685‡	0.5790‡	0.2348‡	0.2562‡	0.2708‡	0.3189‡
ORConvQA$_{2:FID+QR}$	BART(gen)	73.56‡	**48.79**	0.1646‡	0.1760‡	0.6006‡	0.2440‡	0.2661 ‡	0.2823‡	0.3309‡
	T5	**74.12**	45.86†	**0.2008**	**0.2150**	**0.6373**	**0.2829**	**0.3106**	**0.3302**	**0.3764**
Manual		100.00	100.00	0.2486	0.2682	0.8012	0.3540	0.3811	0.4066	3295.0

that there is little difference between the effectiveness of the two versions of the MLT BART models on both F1 and Macro-F1 scores on all three test sets.

Therefore, in response to RQ 6.1, we find that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, jointly learned through both the Follow-up Question Identification and Conversational Question Rewriting tasks has the best overall effectiveness, yielding statistically significant improvements in terms of F1 and Macro-F1 over the baselines, on each of the three test sets of the LIF dataset.

6.4.2 RQ 6.2: Effectiveness on Conversational Question Rewriting Task

Next, we examine the effectiveness of the Conversational Question Rewriting models including our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using the T5 and BART models, and those listed in Section 6.3.3 (STL BART, STL T5, and discriminative+generative MTL BART) on the test set of the OR-QuAC dataset. Table 6.4 presents the effectiveness of various question reformulation models for conversational question rewriting, evaluated based on the ROUGE-1 recall and BLEU scores (see Section 3.4.3). Furthermore, the models' effectiveness for passage retrieval is evaluated when applied with the BM25 [4] retrieval model (see Section 2.2). The effectiveness of the monoT5 re-ranker, applied to the same set of 1000 retrieved passages, is listed in the corresponding row alongside the first stage (BM25) results. The effectiveness of the manually rewritten questions can be seen as an upper bound for the question rewriting methods.

Comparison of Our Proposed *ORConvQA*$_{2:FID+QR}$ MTL Method, using BART and T5 Models, with Baselines for Conversational Question Rewriting: In Table 6.4, we observe that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, achieves the highest ROUGE-1 score by significantly outperforming all STL baselines, demonstrating its superior performance in capturing the recall of the rewritten questions at the unigram level (individual words). On the other hand, our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, achieves the

[4] We also conducted experiments with the DPH retrieval model, which yielded similar trends.

highest BLEU score by significantly outperforming all STL baselines, indicating its effectiveness in measuring the similarity between the generated texts and the reference texts (the manually rewritten questions). Moreover, we can observe that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, outperforms the STL BART model in terms of both the ROUGE and BLEU scores. This indicates that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, achieves a better performance in the conversational question rewriting task compared to the STL BART model. Comparing our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, to the STL T5 model, our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, achieves a higher ROUGE-1 score, indicating a better performance in conversational question rewriting. However, both models have the same BLEU score. Therefore, the results show that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using both BART and T5 models, is more effective in conversational question rewriting than single-task learning.

Comparison of Our Proposed *ORConvQA*$_{2:FID+QR}$ MTL Method, Using BART and T5 Models, with Baselines for Passage Retrieval: From Table 6.4, we observe that our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, achieves the highest MAP, MRR, and Recall@1000, and does significantly improve over our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, and all the STL models in both first stage retrieval and re-ranking. Comparing the MTL and STL models, we observe that both of our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 and BART models, significantly outperform their corresponding STL models. Contrasting the performances of the MTL discriminative+generative BART model with our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, we find that there is little difference between the effectiveness of the two versions of the MLT BART model.

Comparison of our Proposed *ORConvQA*$_{2:FID+QR}$ MTL Method, using BART and T5 Models, with the Baselines for Both Conversational Question Rewriting and Passage Retrieval Tasks: our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using T5 model, demonstrates its effectiveness in both tasks. It not only captures the recall of the rewritten questions at the unigram level but also enhances passage ranking, resulting in our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using the T5 model, outperforming both our proposed *ORConvQA*$_{2:FID+QR}$ MTL method, using BART model, and all of the STL models, yielding statistically significant improvements on both tasks. To illustrate these findings, we provide a further qualitative analysis in Section 6.4.3.

In answer to RQ 6.2, we conclude that our proposed *ORConvQA*$_{2:FID+QR}$ method, which applies the Multi-Task Learning of question rewriting and classification to the T5 model, improves performance in conversational question rewriting and passage retrieval, yielding statistically significant improvements over the MTL discriminative+generative BART model and all the STL models.

Q1: How did the Disco Volante begin?
A1: Due to artwork delays and the band members' many side-projects, it was four years
before Disco Volante was released, in October 1995.
Q2: How did they perform commercially?
A2: Not all critics were impressed with the
album, with The Washington Post describing it as "an **album of cheesy synthesizers**."
Q3: What followed this album?
Manual (Q'3): What followed the **Disco Volante** album?
MTL T5: shift What followed the Disco
Volante album?
STL T5: What followed the album of **Cheesy Synthsynergics**?

Figure 6.2: Comparison of question rewriting models.

6.4.3 Qualitative Analysis

In this section, we conduct a qualitative analysis to further support for our findings concerning the performance of our MTL T5 model in comparison to the STL T5 model, as discussed in Section 6.4.2. The purpose of this qualitative analysis is to further validate our results and to shed additional lights on the advantages of the MTL approach in the Follow-up Question Identification and Conversational Question Rewriting tasks. First, we present an example of dialogue that exemplifies the distinct advantages derived from the use of the MTL approach in our model, specifically in Conversational Question Rewriting. Next, we proceed to compare the differences in NDCG scores between our proposed MTL model and the STL model for passage retrieval.

Conversational Question Rewriting: To illustrate the advantages of the MTL approach in the Follow-up Question Identification and Conversational Question Rewriting tasks, we provide an example dialogue selected based on the highest ROUGE-1 score achieved by our MTL T5 model. This example dialogue, shown in Figure 6.2, showcases the benefits of employing the MTL strategy.

This example consists of a conversation history with two turns (Q1, A1, Q2, A2), the current question (Q3), the manually rewritten question (Q'3), the rewritten question of our MTL T5 model, and the rewritten question of the STL T5 model. The MTL T5 model successfully predicts the word "shift" indicating an invalid follow-up question, as Q3 deviates from the previous conversation's topic of album performance and instead inquires about subsequent events. The MTL T5 model exhibits a superior performance in predicting invalid follow-up questions, demonstrated by the model achieving the best Macro-F1 score on the Test-III LIF dataset (as described in Section 6.4.1). Notably, this test set comprises sampled invalid follow-up questions from the *same conversation*, making this achievement particularly noteworthy. This prediction helps the model to differentiate and choose the accurate entity "Disco Volante" instead of the

misleading prediction "Cheesy Synthsynergics" made by the STL T5 model. This demonstrates the ability of the MTL T5 model to better leverage and interpret the context of the conversation, leading to more accurate predictions and an improved performance.

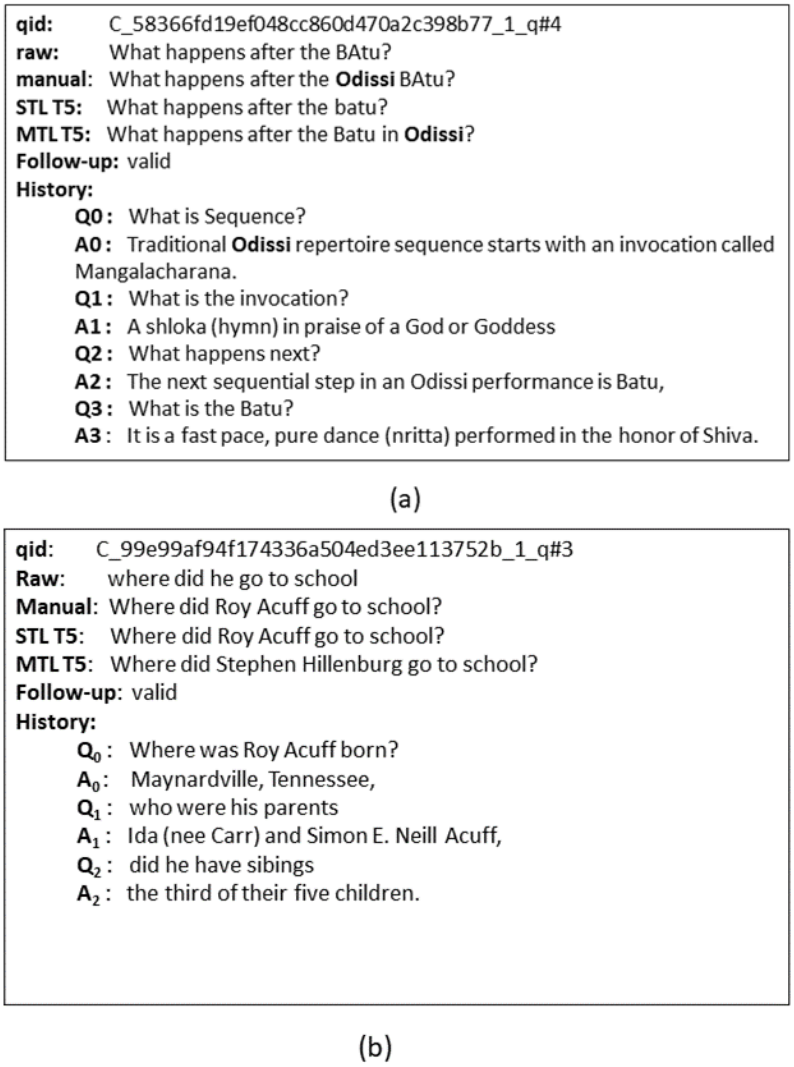


Figure 6.3: Examples of dialogue differences in NDCG for queries in the OR-QuAC query set. (a) a higher NDCG for MTL T5 wrt STL T5 (b) a higher NDCG for STL T5 wrt MTL T5.

Passage Retrieval: We also compare the differences in terms of NDCG scores when using a BM25 ranking model with our proposed MTL model in comparison to using it with the STL model in passage retrieval (MTL T5 vs. STL T5). Figure 6.3 shows two examples of dialogues selected based on the difference in NDCG scores between MTL T5 and STL T5 on the OR-QuAC query set. Overall, MTL T5 outperforms STL T5 for 781 questions, while the opposite was true for 497 questions. Following Macdonald et al. (2021), we only consider differences larger than 0.15 absolute NGCG when inspecting the effect of the MTL approach. Hence, these numbers demonstrate that our proposed MTL T5 model exhibits a superior performance over the STL T5 model in passage retrieval. To further illustrate this point, we closely examine the

predictions made by the MTL T5 model in Figure 6.3 (a). It is clear that the MTL T5 model successfully identifies a candidate question as a valid follow-up to the previous question, thereby demonstrating its capability to potentially aid in the correct resolution of the omitted entity (Odissi). On the other hand, in Figure 6.3 (b), the candidate question "where did he go to school" would not have logically followed the previous question "did he have siblings". However, the MTL T5 model predicted this candidate question as a valid follow-up question; hence this could lead the model to incorrectly resolve the entity (Roy Acuff). As a result, we can observe that the effectiveness of the Follow-up Question Identification task does influence the Conversational Question Rewriting task performance.

6.5 Conclusions

In this chapter, to effectively address the ambiguities in conversational questions, we have proposed a method for Open-Retrieval Conversational Question Answering (ORConvQA), which learns to predict the follow-up question and rewrites the conversational question simultaneously. Our proposed $ORConvQA_{2:FID+QR}$ method makes use of text generation models including BART and T5 by generating the first token for a classification task and the follow-up tokens for a questing rewriting task. Our proposed $ORConvQA_{2:FID+QR}$ method directly addressed the issue highlighted in Gap 2 in Section 3.10, where we stated that there is no effective integration between Follow-up Question Identification and Conversational Question Rewriting. In particular, our investigation in this chapter aims to answer two research questions.

First, to answer RQ 6.1, "Does the MTL approach of question rewriting and classification using text generation models outperform the single-task learning (STL) of text generation models and existing baselines for the Follow-up Question Identification task?", we investigated the performance of the baselines in comparison to our proposed $ORConvQA_{2:FID+QR}$ method for follow-up question classification. As shown in Table 6.3, our proposed $ORConvQA_{2:FID+QR}$ method using BART outperformed the baselines in terms of F1 and Macro-F1 across all three test sets of the LIF dataset, indicating its effectiveness in Follow-up Question Identification.

In order to answer RQ 6.2, "Does the MTL approach of question rewriting and classification using text generation models outperform the single-task learning (STL) of text generation models in the context of the conversational question rewriting and passage retrieval tasks?", we examined the effectiveness of our proposed $ORConvQA_{2:FID+QR}$ method in comparison to the baselines on the test set of the OR-QuAC dataset. The results, presented in Table 6.4, demonstrated that our proposed $ORConvQA_{2:FID+QR}$ method using T5 achieved the highest ROUGE-1 score, surpassing all STL baselines in capturing the unigram-level recall of rewritten questions. In addition, for passage retrieval, our proposed $ORConvQA_{2:FID+QR}$ method using T5 achieved the highest MAP, MRR, and Recall@1000, significantly improving over the MTL BART model and all STL models in both first stage retrieval and re-ranking (see Table 6.4).

Lastly, Section 6.4.3 prevoided a qualitative analysis supporting our findings. This qualitative analysis aimed to further validate our results and elucidate the advantages of the MTL approach in Follow-up Question Identification and Conversational Question Rewriting. For conversation question rewriting, Figure 6.2 showed the ability of the MTL T5 model to better leverage and interpret the context of the conversation, leading to more accurate predictions and an improved performance. For passage retrieval, Figure 6.3 showed that the effectiveness of the Follow-up Question Identification task does influence the Conversational Question Rewriting task performance.

Hence, we can now validate our hypothesis presented in Section 1.3, namely that the use of Multi-Task Learning of Follow-up Question Identification and Conversational Question Rewriting sharing a single text generation model can indeed enhance the system's effectiveness in both tasks. Indeed, we concluded that our proposed $ORConvQA_{2:FID+QR}$ method using BART had the best effectiveness, yielding statistically significant improvements over the baselines. Moreover, our proposed $ORConvQA_{2:FID+QR}$ method using T5 performed the best both in terms of Conversational Question Rewriting, first-stage retrieval, and re-ranking. Gap 2 highlighted the lack of an effective integration between the Follow-up Question Identification and Conversational Question Rewriting. Our findings in this chapter show that our $ORConvQA_{2:FID+QR}$ method learned through Multi-Task Learning by sharing a single text generation model, such as BART and T5, effectively addresses Gap 2.

In the next chapter, to further effectively address the ambiguities in conversational questions, we conduct an experiment to demonstrate the effectiveness of our hybrid of generation and selection for asking clarifying questions. Our method leverages the Multi-Task Learning of clarification need classification and the generation of clarifying questions. This allows us to simultaneously determine when the user's query necessitates a clarifying question and to generate a set of clarifying questions based on the user's query and conversation history.

Chapter 7

Multi-Task Learning of Clarification Need Identification and Clarifying Question Generation

7.1 Introduction

Recall from Section 1.2 that to effectively answer the user's questions, an Open-Retrieval Conversational Question Answering (ORConvQA) system needs to resolve any ambiguities arising from the posed questions based on the conversation history with the user (Kundu et al. 2020). In Chapter 6, we proposed the $ORConvQA_{2:FID+QR}$ method, which leveraged Multi-Task Learning (MTL) to simultaneously learn the tasks of Follow-up Question Identification (FID) and Conversational Question Rewriting (QR) that have been described in Sections 3.3 and 3.4, respectively. Our proposed $ORConvQA_{2:FID+QR}$ method made use of text generation models including the BART and T5 models (see Section 2.3.1) by generating the first token for a classification task and the follow-up tokens for a questing rewriting task. The experiments in Sections 6.4.1 and 6.4.2, along with the analysis presented in Section 6.4.3 showed that leveraging MTL in the $ORConvQA_{2:FID+QR}$ method improved performance for both the FID and QR tasks.

In this chapter, we further explore how to address the ambiguities in conversational questions. As described in Clarification Need Classification (Sections 3.5.4) and Asking Clarifying Questions (see Section 3.6.4), existing works aim to improve the search experience for users by allowing for more natural interactions between the users and search systems. A mixed-initiative conversational search system (Keyvan & Huang 2022, Zamani et al. 2022) combines both the machine and the human initiative in order to improve the search experience for the user. It achieves this by asking clarifying questions to the users in order to better understand their intents and to refine the system's interpretation of the users' information needs, as previously described in Section 3.6.

To address the task of Asking Clarifying Questions in conversational search, as previously discussed in Section 3.6.4, existing works can be broadly classified into two categories: *generation*

and *selection* approaches. In the generation-based, for example, Owoicho et al. (2022) fine-tuned the T5 (see Section 2.3.1) and GPT-3 (see Section 2.3.3) text generation models to generate the clarifying questions. We consider these models (T5 and GPT-3) as generative baselines for comparison with our system. On the other hand, selection-based approaches, Owoicho et al. (2022) adopted retrieval models such as BM25 (see Section 2.2) and miniLM (Wang, Wei, Dong, Bao, Yang & Zhou 2020) to select a candidate set of questions from the question pool. In this work, we also consider other recent effective retrieval models such as ANCE (see Section 2.4.2), monoT5 (see Section 2.4.1), ColBERT (see Section 2.4.3), and TCT-ColBERT (see Section 2.4.2) as clarifying question selection baselines.

However, as discussed in Section 3.6.4, both generation and selection approaches for asking clarifying questions have their advantages and disadvantages. We argue that by combining both the generation and selection approaches in a uniform framework, we can produce a better set of questions and ensure that the selected question is relevant to the user's query. However, we are not aware of any previous work that effectively combines both the generation and selection approaches for the task of asking clarifying questions, in order to leverage the best of both worlds. In this chapter, we introduce a novel hybrid asking clarifying question method, which uniformly combines the generation and selection processes.

Based on the literature reviewed in Sections 2.8 and 3.9, Multi-Task Learning (MTL) is described as a method where multiple different but related tasks are learned simultaneously. In particular, as introduced by Aliannejadi et al. (2021), the task of asking clarifying questions (see Section 3.6.2) in conversational search can be broken down into two subtasks: determining when to ask clarifying questions (Clarification Need Classification) and how to generate them (Asking Clarifying Questions). In this chapter, we aim to address these two subtasks simultaneously thereby providing a comprehensive approach to the task of asking clarifying questions in conversational search. To the best of our knowledge, as identified in Gap 3, no prior work has seamlessly and uniformly combined both subtasks to more effectively address the step of asking clarifying questions in a mixed-initiative conversational search system. In this chapter, as stated in Section 1.3, we hypothesise that MTL can be used to jointly learn the Clarification Need Classification and the generation of the clarifying question subtasks in order to simultaneously determine when the user's query necessitates a clarifying question and to generate a set of clarifying questions accordingly. Therefore, to address the limitation in Gap 3, we propose a newly MTL model called *T5MI*, which is included in our proposed *ORConvQA*3 method, as introduced in Section 4.4.3.

To summarise, in this chapter, we introduce the $ORConvQA_{3:CNC+Askng}$ method, which includes our proposed MTL model called *T5MI*, a selecting asking clarifying question called *GTR*, and our ranking clarifying questions called *T5Ranking*, to address the issue presented in Gap 3. Our proposed $ORConvQA_{3:CNC+Askng}$ method is a novel approach for generating and selecting clarifying questions in a mixed-initiative conversational search. In generating clarifying

questions, our *T5MI* model leverages the Multi-Task Learning of clarification need classification and the generation of clarifying questions to simultaneously determine when the user's query necessitates a clarifying question and to generate a set of clarifying questions based on the user's query and conversation history. For selecting the clarifying questions, we use the state-of-the-art Generalisable T5-based dense Retriever (*GTR*) (see Section 2.4.2) for retrieving clarifying questions from the question pool. To rank the candidate clarifying questions obtained from both the generation and selection approaches, we score the questions using a text generation model for point-wise question classification called *T5Ranking*. By leveraging both the generation and selection models, our $ORConvQA_{3:CNC+Askng}$ method is able to generate a better set of questions and to ensure that the selected question is relevant to the user's query.

Our contributions in this chapter can be summarised as follows:

(1) We leverage Multi-Task Learning (MTL) with a single text generation T5 model by jointly learning a classifier for clarification need and the effective generation of clarifying questions called *T5MI*;

(2) We introduce a $ORConvQA_{3:CNC+Askng}$ MTL method including hybrid of generating and selecting clarifying questions. Our $ORConvQA_{3:CNC+Askng}$ method generates clarifying questions using MTL *T5MI* model and selects clarifying questions from a pool of pre-determined question using *GTR* model to effectively address the task of asking clarifying questions;

(3) We evaluate the performance of our hybrid asking clarifying question method (T5MI + GTR + T5Ranking), which is included in our proposed $ORConvQA_{3:CNC+Askng}$ method, on a recent dataset of mixed-initiative conversational search and show the effectiveness of our proposed hybrid method (T5MI + GTR + T5Ranking), which significantly outperforms existing strong baselines with improvements at P@1 by up to 20% on relevance criteria and 30% on novelty criteria.

The rest of the chapter is structured as follows: Section 7.2 recalls the definitions of the Clarification Need Classification (CNC) and Asking Clarifying Questions tasks, along with our proposed $ORConvQA_{3:CNC+Askng}$ method (see Section 4.4.3). Our proposed $ORConvQA_{3:CNC+Askng}$ method makes use of our T5MI model: an MTL of the Clarification Need Classification (CNC) and Asking Clarifying Questions, GTR: a selecting clarifying questions model, and T5Ranking: a ranking the candidate clarifying question model, to address these tasks simultaneously. We present our experimental setup in Section 7.3 and show the results of the experiments in Section 7.4. Finally, we provide concluding remarks in Section 7.5.

7.2 Generating and Selecting Clarifying Questions

In this section, we describe our mixed-initiative system that combines both the machine and the human initiative in order to improve the search experience for the user. Our system makes use of the $ORConvQA_{3:CNC+Askng}$ method outlined in Section 4.4.3 and consists of four

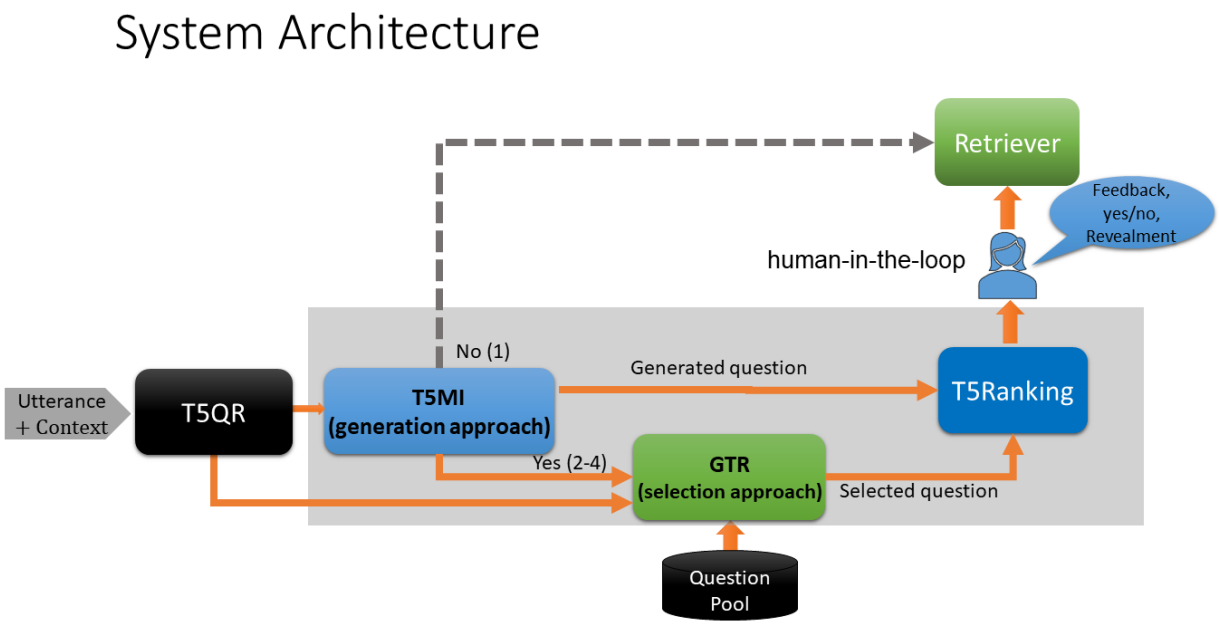


Figure 7.1: The overall framework of our Mixed-Initiative Conversational Search system.

main models. We first explain how these models are used to perform the Clarification Need Classification and Asking Clarifying Questions task in Section 7.2.1. An overview of our proposed $ORConvQA_{3:CNC+Askng}$ method follows in Section 7.2.2. Then, we explain how to fine-tune the involved models in Section 7.2.3.

7.2.1 Task Definitions

In this chapter, we aim to tackle the tasks of clarification need classification (CNC) (see Section 3.5.1) and asking clarifying questions (Asking) (see Section 3.6.1). Table 7.1 presents the notations, which include a subset of the symbols defined in Chapter 4, Table 4.1. The CNC and Asking tasks have been formalised in Equations (4.3) and (4.4) of Chapter 4 as follows:

$$\begin{aligned} CNC(q_k, H_k) &\rightarrow n_k \\ Asking(q_k, H_k) &\rightarrow c_k \end{aligned} \tag{7.1}$$

where $CNC(\cdot)$ is a function that determines the necessity of asking clarifying questions, n_k is the importance of (or the need for) asking a clarifying question, while $Asking(\cdot)$ is a function that generates clarifying questions using a generative model or selects them from a pre-determined pool. The clarification need classification task aims to classify whether the user's query requires further clarification or if it can be directly addressed without additional information. The asking clarifying questions task aims to either generate clarifying questions using a generative model or to select them from a pre-determined pool of questions. Note that the task *CNC* can be formulated as a classification task while *Asking* is a text generation task. In the following section, we describe our proposed $ORConvQA_{3:CNC+Askng}$ method to address these tasks.

Table 7.1: Notations used in Chapter 7.

Notation	Definition
u_k	A current utterance
r_k	An answer to current utterance u_k
H_k	A conversation history i.e. $H_k = [\langle u, r \rangle]$
c_k	A clarifying question
C_k	A list of c_k i.e. $C_k = [c_{k_1}, c_{k_2}, ...c_{k_n}]$
c'_k	A top-ranked clarifying question
u_{k_2}	A user feedback
u_r	A rewritten utterance of u_k and H_k
u'_r	A rewritten question of c'_k, u_k, u_{k_2}, and H_k
a_k	An importance level of asking a clarifying question
W	A sequence of m words i.e. $W = \{w_1, w_2, ..., w_m\}$
"$[sep]$"	A delimiter token
$w_1, ..., w_n$	An output sequence for the target query
E_u	an embedding of u_r
E_c	an embedding of c
$score(u_r, c)$	a similarity score between u_r and c
$QR(\cdot)$	A conversational question rewriting function
$CNC(\cdot)$	A clarification need classification function
$ASKING(\cdot)$	A asking clarifying questions function
$Retriever$	A passage retriever function
$Reranker$	A passage reranker function

7.2.2 Models Overview

To tackle the tasks described in Section 7.2.1, typically, a system for asking clarifying questions (Keyvan & Huang 2022, Zamani et al. 2022) consists of three components:

1. clarification need classification;
2. generating or selecting clarifying questions; and
3. ranking the candidate clarifying questions.

However, to address the tasks in Section 7.2.1, the system needs to correctly interpret a question in the context of an ongoing conversation. Hence, we introduce a conversational query rewriting component, which reformulates the current utterance u_k and conversation history H_k, into a clearer, standalone utterance u_r. In addition, we adopt a hybrid method that seamlessly combines the generation and selection of clarifying questions. Furthermore, we leverage Multi-Task Learning (MTL) for jointly learning the clarification need classification and the generation of clarifying questions processes by sharing a single text generation model. As a consequence, as presented in Figure 7.1, our proposed $ORConvQA_{3:CNC+Askng}$ method consists of four main components, which address the following functionalities:

Table 7.2: The input-output of each component of our overall hybrid method.

Model	Input	Output
T5QR	$H_k; u_k$	u_r / u'_r
T5MI	u_r	$a_k,\ c_k$
GTR	u_r	c_k
T5Ranking	$u_r; c_k$	"true" or "false"

1. conversational query rewriting, namely T5QR;
2. the MTL of the clarification need classification and the generation of clarifying questions, namely T5MI;
3. selecting the clarifying question, namely GTR; and
4. ranking the candidate clarifying questions, namely T5Ranking.

In the following, we describe our proposed $ORConvQA_{3:CNC+Askng}$ method and its models & components. These components include the hybrid method, denoted as $Hybrid(\cdot)$, which is a combination of components (2) T5MI and (3) GTR above. This hybrid method is used for generating and selecting clarifying questions. Table 7.2 shows the input-output correlation of each component in our proposed system. Focusing on the leftmost component in Figure 7.1, T5QR reformulates the current user's utterance u_k and its context (conversation history) into a standalone, omission-free [1] rewritten utterance u_r, which can be used in the later stages in a decontextualised manner. The next component, our proposed T5MI model, employs Multi-Task Learning to simultaneously learn the Clarification Need Classification and the generation of the clarifying questions subtasks by sharing a single text generation model. T5MI estimates the importance of (or the need for) asking a clarifying question for the rewritten utterance u_r, which we denote as a_k. Based on a_k, the system will decide whether a clarifying question should be obtained. If a clarification is not required, the system retrieves passages using the rewritten utterance u_r and the retrieval model. If a clarification is needed, the system employs both the generation and selection approaches for clarifying questions – specifically, it generates clarifying questions using T5MI and selects from a pre-determined pool of questions using GTR. In order to produce the most appropriate clarifying question, the system scores the candidate clarifying questions obtained from both the generation and selection approaches, using T5Ranking. For each clarifying question, the user's feedback will be provided, as illustrated in the top-right of Figure 7.1. Next, we use each clarifying question c_k and its feedback as the context for rewriting the current user utterance u_k using the conversational query rewriting T5QR model.

In particular, as define in Equation (4.17) in Section 4.4.3, the $ORConvQA_{3:CNC+Askng}$ method can then be defined as follows:

[1] Omission: A noun group after a preposition can be omitted if it has already been mentioned in previous queries (Yu et al. 2020).

$$
\begin{aligned}
ORConvQA_3 = T5QR(u_k, H_k, \theta) \\
&\overset{u_r}{\gg} Hybrid\ (T5MI\,(CNC, Asking, \theta), \text{GTR}) \\
&\overset{C_k}{\gg} T5Ranking(u_r, \theta) \\
&\overset{c'_k}{\gg} T5QR(u_k, u_{k\_2}, H_k, \theta) \\
&\overset{u'_r}{\gg} Retriever(I) \\
&\overset{P^+_{Ret}}{\gg} Reranker(u'_k, \theta)
\end{aligned}
\tag{7.2}
$$

We now describe each component of our overall proposed $ORConvQA_{3:CNC+Askng}$ method in detail.

1. **T5QR: Conversational Query Rewriting:** The left part of Figure 7.1 presents the conversational query rewriting component. Following Dalton et al. (2021) and Owoicho et al. (2022), we deploy a generative model to capture the relationship between the current user utterance u_k and the contextual information in the conversation history H_k, including a list of $k-1$ utterances and the response text pairs, i.e. $H_k = [\langle u_i, r_i \rangle]_{i=1}^{k-1}$. In particular, we define a T5QR transformation function as $T5QR(\cdot)$, which takes a text input sequence, as follows:

$$
T5QR(\, u_1 \text{ “[SEP]” } r_1 \text{ “[SEP]” } u_2 \text{ “[SEP]” } ... \text{ “[SEP]”} u_{k-1} \text{“[SEP]” } r_{k-1} \text{ “[SEP]” } u_k \big) \tag{7.3}
$$

$$
T5QR(\cdot) = w_1, w_2, ..., w_n \tag{7.4}
$$

where "[SEP]" is a separator token. The model is then fine-tuned to generate the target tokens of length n as shown in Equation (7.4), where $w_1, ..., w_n$ are the output sequence for the target query. Rewriting the user's query first using the T5QR model helps refine and contextualise the user's intentions based on the conversation history before any clarification attempts are made. This process not only improves the clarity and specificity of the user's original question but also prepares it for more effective processing in subsequent system components, such as clarification, by ensuring that the query fully incorporates and reflects the preceding dialogue context.

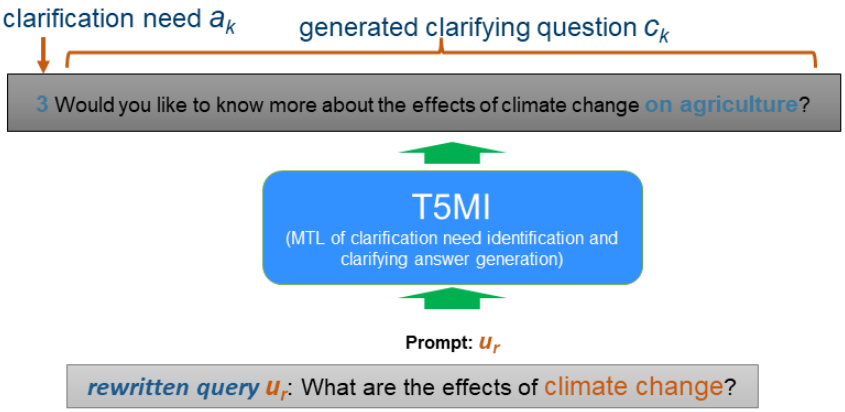


Figure 7.2: T5MI: MTL of clarification need classification and clarifying question generation.

2. **MTL of Clarification Need Classification & Generation of Clarifying Question(T5MI):** As shown in the second part from the left of Figure 7.1, our proposed model uses T5 (see Section 2.3.1), a large pre-trained language model designed for text generation tasks. In particular, text generation approaches, such as BERT (see Section 2.3.2), BART (see Section 2.3.1), and T5 (see Section 2.3.1), can be trained to generate a coherent and relevant response based on some input text. These models can also be fine-tuned to perform a variety of downstream tasks. To adopt a Multi-Task Learning (MTL) approach to a text generation model for jointly learning from both the clarification need identification and the generation of clarifying question, the MTL model makes predictions as follows: the first output token is used to estimate the clarification need, while the other output tokens define the output of the clarifying question generation task. To fine-tune the T5 model for a downstream task, we employ the technique of Prompt-based Learning (see Section 2.8.1). As depicted in Figure 7.2, we deploy a T5 model to analyse the rewritten utterance u_r of the current user's utterance u_k. In particular, as defined in Equation (4.16) of Chapter 4, let $MTL_{T5MI}(\cdot)$ denotes a joint learning function of the tasks defined in Equation (7.1) as follows:

$$MTL_{T5MI}(CNC, Asking, \theta) \rightarrow w_1, w_2, ..., w_m \tag{7.5}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to generate m target tokens, as shown in Equation (7.5). Token w_1 can be "1", "2", "3" or "4" [2] depending on whether the rewritten utterance u_r needs to ask the clarifying question or not, while the follow-up tokens $w_2, ..., w_m$ are the output sequence for the clarifying question c_k.

3. **Clarifying Question Selection:** The bottom part of Figure 7.1 presents the clarifying question selection component. We use a Generalizable T5-based dense Retriever (GTR) (see Section 2.4.2), which consists of a question encoder and a passage encoder. Recall from Chapter 3 that the generalisability of the GTR model allows it to perform well on a

[2] Following the assessment procedure of the ClariQ dataset used in this work.

wide range of tasks and domains, as it uses a large pre-trained model that can be adapted into different specific tasks without being fine-tuned. The GTR model first encodes the rewritten utterance u_r and the pre-determined clarifying question c in the question pool into the embedding space:

$$E_u = GTR(u_r),$$
$$E_c = GTR(c) \tag{7.6}$$

where E_u and E_c are the embeddings of u_r and c, respectively. GTR uses the similarity scoring function *sim*, which is the cosine similarity of E_u and E_c, to calculate the relevance score of a pre-determined clarifying question c for the rewritten utterance u_r:

$$score(u_r, c) = sim(E_u, E_c) \tag{7.7}$$

To retrieve the top K clarifying questions in the embedding space, we deploy GTR with FAISS (Johnson et al. 2021), which is a library for efficient approximate nearest neighbour search.

4. **Clarifying Question Ranking (T5Ranking):** The right part of Figure 7.1 presents our proposed model for this component, which uses T5 (see Section 2.3.1), a large pre-trained language model designed for text generation. We chose the encoder-decoder T5 model for ranking because an existing variant, monoT5 (see Section 2.4.1), has been widely used and shown to be effective in similar tasks. To rank the clarifying questions obtained from both the generation and selection approaches, following (Nogueira, Jiang & Lin 2020), we fine-tune the T5 model for point-wise question classification. In particular, let $T5Ranking(\cdot)$ denotes a generative transformation function that takes an input sequence as follows:

$$T5Ranking(f_{prompt}(u_r, c_k)) \rightarrow w_1 \tag{7.8}$$

where $f_{prompt}()$ is a prompt function (template) to format the (rewritten) utterance u_r, and the clarifying question c_k into an input sequence for T5Ranking. The model is then fine-tuned to generate a target token, as shown in Equation (7.8), where the output token w_1 is either "true" or "false"[3] depending on whether the clarifying question c is relevant or not to the rewritten utterance u_r. At inference time, we follow monoT5 (Nogueira, Jiang, Pradeep & Lin 2020*a*) and calculate a ranking score by applying a softmax function to the logits of the "true" and "false" tokens of the first generated token (w_1) in the sequence as follows:

$$score = softmax(w_1) \tag{7.9}$$

[3] Following monoT5 (Nogueira, Jiang, Pradeep & Lin 2020*b*), we select "true" and "false" as target tokens.

7.2.3 Training

This section presents an overview of the training procedure for the T5MI and T5Ranking models. In particular, Multi-Task Learning is employed to simultaneously predict the clarification needs and the generation of the clarifying questions by sharing a single text generation model.

7.2.3.1 T5MI:

Given a rewritten utterance u_r (see Section 7.2.2), our T5MI model is jointly trained for the clarification need identification and the generation of clarifying question as follows:
Joint training: We fine-tune the T5MI model to generate the tokens for both the clarification need identification and the generation of clarifying question. In particular, our T5MI model leverages the prompt function (Equation (7.3)) to format the rewritten utterance u_r into an input sequence. The T5MI model then generates a contextual representation h, which is used by the decoder to perform attention and to generate the next token. In particular, given a tuple $\langle u_r, c \rangle$, the training objective is to minimise the following loss function L_{gen}:

$$\mathcal{L}_{gen} = \sum_{i=0}^{N} \log P(c_k^i | h, c_k^{:i}) \tag{7.10}$$

where N is the length of the target clarifying question c_k, c_k^i is the i^{th} token in c_k, and c_k^0 is the beginning of sequence token (<s>).

Recently, Prompt-based Learning (see Section 2.8.1), a method that adapts pre-trained language models to downstream tasks by using task-specific prompts, has recently gained traction for tackling various tasks using a single model. To fine-tune the T5MI model for the clarification need classification and the generation of clarifying question, we also adopt prompt-based learning by modifying the model's input using a new template with the prefix "Clarification Question:":

$$\text{“}\textit{Clarification Question} : \{u_r\}\text{”} \tag{7.11}$$

7.2.3.2 T5Ranking:

We approach the ranking of clarifying questions as a relevance prediction problem, where the task is to evaluate the relevance of a candidate clarifying question to a user's utterance by assigning a relevance score. To capture this task, we apply the following input template:

$$\text{“}\textit{Question Relevance} : \{u_r\}\ [sep]\ \{c_k\}\text{”} \tag{7.12}$$

where [sep] is a T5-provided special token that acts as a separator token between the candidate clarifying question c_k and the rewritten utterance u_r.

Table 7.3: Statistics of the used datasets

Dataset	Feature	Train	Dev	Test
ClariQ	topics	187	50	61
	questions	3,766	163	270
	documents	∼2 million		
TREC CAsT 2022	topics	-	-	18
	questions	-	-	205
	documents	∼17 million		

Next, we evaluate our proposed $ORConvQA_{3:CNC+Askng}$ method, as defined in Equation (7.2), and its resulting mixed-initiative system for asking clarifying questions in comparison to several existing baselines as detailed in the next section.

7.3 Experimental Setup

Our experiments address the four following research questions:

RQ 7.1 Does leveraging the MTL of classification and clarifying question generation on the text generation model improve the effectiveness of the clarification need classification over the existing single-task learning (STL) baselines (e.g. the best approaches from the ConvAI3 leaderboard (Aliannejadi et al. 2020*a*))?

RQ 7.2 How does our proposed hybrid method for generating and selecting clarifying questions compare in the effectiveness of asking clarifying questions with other existing baselines, namely: (1) the **miniLM-BERT**, **BM25-baseline** , **T5-***, and **GPT-3-*** systems, as described in Section 3.6.4; (2) existing recent and strong retrievers including ANCE (see Section 2.4.2), ColBERT (see Section 2.4.3), and TCT-ColBERT (see Section 2.4.2) as baselines for selecting the clarifying questions; (3) cross-encoder rerankers, such as monoT5 (see Section 2.4.1).

RQ 7.3 How to effectively rewrite the current utterance u_k by using the clarifying question c_k and user feedback u_{k_2} for passage retrieval?

RQ 7.4 How effective is our proposed $ORConvQA_{3:CNC+Askng}$ method in terms of passage retrieval performance compared to existing baselines, namely: the **miniLM-BERT**, **BM25-baseline**, **T5-***, and **GPT-3-*** systems, as described in Section 3.6.4?

The remainder of this section provides details on the datasets used (see Section 7.3.1), describes our baselines (see Section 7.3.2), and explains the experimental implementation (see Section 7.3.3).

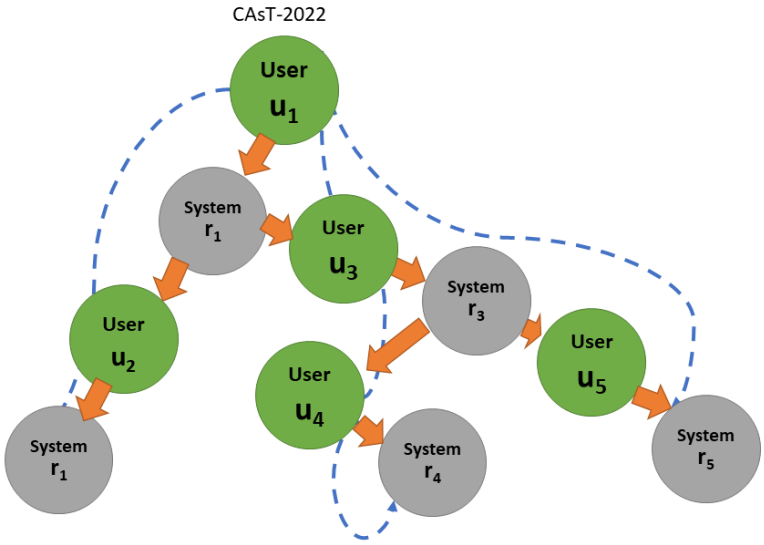


Figure 7.3: Example of a CAsT 2022 dialogue tree with 1 main topic, 3 sub-topics, and 5 user utterances.

7.3.1 Datasets

To conduct the evaluation of our proposed $ORConvQA_{3:CNC+Askng}$ method, including our hybrid method for generating and selecting clarifying questions, we choose the TREC Conversational Assistance Track (CAsT) 2022 and the ClariQ datasets, as previously described in Section 3.6.2. For completeness, we reiterate the details of the datasets, TREC Conversational Assistance Track (CAsT) 2022 and ClariQ, in this chapter, as originally presented in Section 3.6.2. For further information about the used datasets, we also provide a summary of their statistics in Table 7.3.

The TREC CAsT 2022 dataset includes a collection of evaluation topics in the form of search conversations, a mixed-initiative question pool for the mixed-initiative subtask, and three document collections. The three document collections are: (1) The KILT Wikipedia dump from 2019/08/01, consisting of 5 million articles; (2) The MSMARCO V2 document corpus from the 2021 TREC Deep Learning Track, consisting of 11.9 million documents from the Bing search engine; and (3) The TREC Washington Post collection (V4 2020), consisting of 728,626 news articles from 2012 to 2020. The evaluation topics include a dialogue tree that represents all possible conversations between the user and the system. Figure 7.3 illustrates a CAsT 2022 dialogue tree composed of 1 main topic, 3 sub-topics, and 5 user utterances. As depicted in Figure 7.3, user utterance u_1 can appear in all sub-topics, including (1) $u_1 : r_1 : u2 : r2$; (2) $u_1 : r_1 : u_3 : r_3 : u_4 : r_4$; and (3) $u_1 : r_1 : u_3 : r_3 : u_4 : r_5$. However, for the purpose of evaluating performance in the conversational search task, u_1 can only be evaluated in sub-topic (1) (Owoicho et al. 2022). For each user utterance, in the CAsT 2022 mixed-initiative sub-task, the system can pose questions to the user to gain additional context. For the mixed-initiative sub-task, each submitted system would retrieve assessments for the *top-one* clarifying question returned per

turn. These assessments were obtained by TREC CAsT organisers using crowdsourcing to gather judgments of relevance, diversity, and novelty. After submitting the clarifying questions to TREC CAsT 2022, the organisers gathered user feedback for the top-ranked question in each turn. The feedback for the clarifying question serves as context to rewrite the raw utterance. In this work to evaluate the clarification questions not returned among the *top-one* processed questions by the organisers, additional relevance judgements were conducted following the same procedure as Owoicho et al. (2022). We engaged five workers to assess each turn and clarification question pair, given a conversation context as exemplified by Figure 7.4. Furthermore, following CAsT 2022 (see Section 3.7.2), our generated clarification questions are evaluated in terms of: relevance (follows logically from previous conversation), novelty (adds new information), and diversity (number of options provided). The final judgment for each question is based on a majority vote from the workers (mode), or an average of all judgments if there is no clear majority. Following the same used guidelines in CAsT 2022 (Owoicho et al. 2022), to eliminate low quality judgments, we identified and inspected instances of low-quality questions within each topic. Using these inspections, we filtered out judgements made by workers who deviated from our standards, such as labeling a clearly non-relevant question as moderately relevant (i.e. a score of 3 instead of 0). For example, we anticipate that the question "Are you looking for a specific web site?" should be judged as non-relevant or partially relevant for the user utterance "How can I demonstrate this?" in the example of Figure 7.4. If a worker rated it as relevant or highly relevant, we would eliminate their assessments for the entire topic.

ClariQ (see Section 3.6.2) is a crowdsourced dataset for clarifying questions in open-domain for single and multi-turn conversations, which is based on ambiguous queries from the TREC Web tracks (Clarke et al. 2011). We use the training and development sets of ClariQ for training our T5MI and T5Ranking models. Indeed, this is necessary as CAsT 2022 only provides an evaluation set for clarifying questions and does not include any means to evaluate the task of classifying clarification needs. Furthermore, the ConAI3 (Aliannejadi et al. 2020*a*) challenge has a leaderboard on clarification need identification using the ClariQ dataset. We use this leaderboard in addressing RQ 7.1. Moreover, as ClariQ addresses both (1) "When to ask clarifying questions during dialogues," and (2) "Which clarifying questions should be asked?", we can employ Multi-Task Learning using the ClariQ dataset tasks, including clarification need identification and asking clarifying questions, to train our MTL T5MI model. However, for evaluating approaches on the asking clarifying questions task, we solely use the CAsT 2022 dataset as the ClariQ dataset does not provide an evaluation for generating clarifying questions.

7.3.2 Baselines

To demonstrate the effectiveness of our proposed $ORConvQA_{3:CNC+Askng}$ method (refer to Equation (7.2)), which includes our hybrid method for generating and selecting clarifying question (T5MI+GTR+T5Ranking), we compare it with the 11 baseline systems listed as (a)-(k)

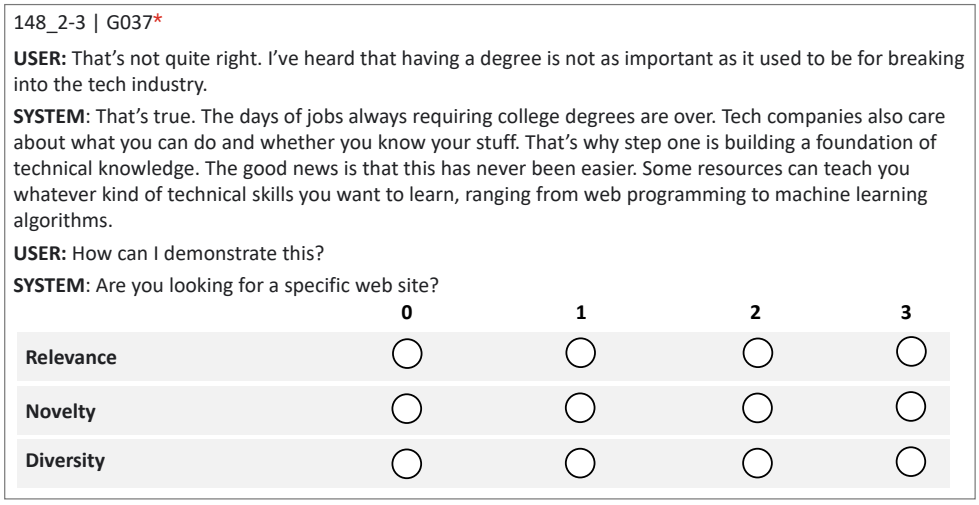


Figure 7.4: Example of evaluating clarifying questions based on levels of relevance, novelty, and diversity (0-3) (Owoicho et al. 2022).

Table 7.4: List of baselines.

	Baseline	Architecture	Trained dataset	Input
	generating clarifying questions			
(a)	T5-raw (Section 3.6.4)	encoder-decoder	ClariQ	u_k
(b)	T5-rewrite (Section 3.6.4)	encoder-decoder	ClariQ	u_r
(c)	GPT-3-raw (Section 3.6.4)	decoder only	CAsT 2021	u_k
(d)	GPT-3-rewrite (Section 3.6.4)	decoder only	CAsT 2021	u_r
(e)	GPT-3-full-context (Section 3.6.4)	decoder only	CAsT 2021	H_k, u_k
	selecting clarifying questions			
(f)	BM25 (Section 2.2)	sparse	-	u_r
(g)	miniLM-BERT (Section 3.6.4)	cross-encoder	ClariQ	u_r
(h)	ANCE (Section 2.4.2)	bi-encoder	MSMARCO	u_r
(i)	monoT5 (Section 2.4.1)	cross-encoder	MSMARCO	u_r
(j)	ColBERT (Section 2.4.3)	bi-encoder	MSMARCO	u_r
(k)	TCT-ColBERT (Section 2.4.2)	bi-encoder	MSMARCO	u_r

in Table 7.4.

- **Generative baselines:** In particular, the baselines (c)-(e) use GPT-3 (see Section 2.3.3) with few-shot prompting from the CAsT 2022 data (see Section 3.7.2). We select these GPT-3 baselines because GPT-3 is a state-of-the-art generative model that can be the basis for creating the strongest generation of clarifying question baselines. In addition, we include Single-Task Learning baselines (a)-(e) (Table 7.4 (top-half)), based on T5 (see Section 2.3.1) and GPT-3 (see Section 2.3.3), for comparison with our Multi-Task Learning T5MI model, jointly learning the clarification need classification and the generation of clarifying question.

- **Selective baselines:** Baseline (f), BM25 (see Section 2.2), ranks questions using BM25 with an automatic rewritten query. Baseline (g), miniLM-BERT (see Section 3.6.4),

employs a two-step approach to question selection. First, a candidate pool of questions is selected using the all-MiniLM-L6-v2 model (Wang, Wei, Dong, Bao, Yang & Zhou 2020). Then, these candidates are re-ranked using a BERT model (see Section 2.3.2), trained on the ClariQ dataset. In addition, for baselines (h)-(k), we use recent existing retrievers, mostly fine-tuned on MSMARCO, without further fine-tuning.

To evaluate the components of our proposed $ORConvQA_{3:CNC+Askng}$ method in generating and selecting questions, we also include two additional variant baseline methods:

(m) "Generation Only", which relies solely on T5MI to generate clarifying questions, and

(n) "Selection Only", which only employs the clarifying question selection component, i.e. GTR (see Section 2.4.2).

7.3.3 Experimental Implementations

Hyperparameter settings: For baselines (a)-(g), we directly use the results provided by the CAsT 2022 organisers as they do not provide the models to allow further reproduction. Furthermore, we implement the dense retriever baselines (h) ANCE [4], (i) monoT5 [5], (j) ColBERT [6], and (k) TCT-ColBERT [7] using their various provided checkpoints including `passage_ance_firstp`, `castorini/monot5-base-msmarco`, `colbertv2.0`, and `castorini/tct_colbert-v2-hnp-msmarco`, respectively. For the selecting clarifying question model (GTR) [8], we use the `sentence-transformers/gtr-t5-large` checkpoint.

We implement the T5MI and T5Ranking models using PyTorch models from HuggingFace (Wolf et al. 2020), namely `castorini/monot5-large-msmarco` (Nogueira, Jiang, Pradeep & Lin 2020*a*). The T5MI and T5Ranking models are configured as follows: the maximum sequence length is set to 512, the number of training epochs is set to 20, the batch size is set to 16, and the learning rate is set to $5e^{-5}$. The T5MI and T5Ranking models are fine-tuned on a NVIDIA RTX A6000. To train T5MI, during the evaluation step, we selected the best training model checkpoint by using the highest F1-measure, which is calculated as the percentage of correctly predicted first tokens generated by the model with the development set of the ClariQ dataset. As previously described in Section 3.5.2, the target tokens "1", "2", "3", and "4" correspond to the level of importance of asking a clarifying question a_k. The lower the number, the less important it is to ask a clarifying question. For text generation, we use a beam search with a beam width of 5.

Evaluation metrics: To evaluate the Clarification Need Classification, we use typical classification metrics such as Precision, Recall, and F1, as previously described in Section 3.5.3. To evaluate the effectiveness of Asking Clarifying Questions, as described in Section 3.6.3), we apply P@1 (using a relevance cutoff at two as positive for binary measures), and assess based on

[4] https://github.com/microsoft/ANCE/

[5] https://huggingface.co/castorini/monot5-base-msmarco

[6] https://github.com/stanford-futuredata/ColBERT/

[7] https://huggingface.co/castorini/tct_colbert-v2-hnp-msmarco

[8] https://huggingface.co/sentence-transformers/gtr-t5-large

Table 7.5: Accuracy of our Multi-Task Learning T5MI model for Clarification Need Classification (ClariQ test set) compared to Single-Task Learning systems on the ConvAI3 leaderboard.

Rank	Creator	Model Name	Precision	Recall	F1
1	TAL ML	Roberta+++	**59.81**	**65.57**	**60.70**
2	T5MI (ours.)	T5MI	56.89	60.66	58.39
3	Cactusjam	Roberta+Stats	59.63	59.02	54.16
4	TAL ML	Roberta++	52.90	55.74	52.53

three criteria: relevance, novelty, and diversity. To evaluate passage retrieval performance, as introduced in Section 2.7, we adopt the NDCCG@3, MAP@1000, MRR@1000, and Recall@1000 metrics used in CAsT 2022. We then use paired t-tests to determine the statistical significance of performance differences between the systems' performances.

Passage retrieval pipeline: We use the PyTerrier platform (see Section 2.5) for indexing and retrieving passages. We implement a hybrid sparse-dense retrieval model, as described in Section 2.6, which combines the sparse retrieval method (DPH with Bo1 (Amati & Van Rijsbergen 2002) query expansion) and the dense TCT-ColBERT (see Section 2.4.2) applied on a FAISS (Johnson et al. 2021) index. The passages returned by both retrieval models are merged and reranked using monoT5 (see Section 2.4.1). We employ a hybrid of sparse and dense retrieval due to its demonstrated high effectiveness on the MSMARCO v2 dataset (which is part of the CAsT 2022 document collection) as shown by Wang, MacAvaney, Macdonald & Ounis (2021*b*).

7.4 Results Analysis

We now address each of RQs 7.1-7.4 (see Section 7.3) in turn.

7.4.1 RQ 7.1: Identifying Clarification Needs

In this section, we focus on the accuracy of the mixed-initiative system for asking clarifying questions on the clarification need classification. Table 7.5 shows the performance of our Multi-Task Learning (MTL) T5MI model compared with Single-Task Learning (STL) systems from the ConvAI3 leaderboard on the test set of the ClariQ dataset. Our MTL T5MI model is trained as described in Section 7.2.3.

Table 7.5 shows that Roberta+++ by TAL ML has the highest Precision, Recall, and F1 scores (59.81, 65.57, 60.70 respectively), followed by our MTL T5MI model. A further investigation of the results reveals that the high performance of Roberta+++ is due to its use of both the user utterance u_k and the user feedback u_{k_2} in its prediction of clarification need a_k (Li et al. 2020). In contrast, T5MI only considers the user utterance u_k. Moreover, in reality, the user feedback can only be obtained by asking clarifying questions. This suggests that the effectiveness of the Roberta+++ model may be limited in real-world applications, as it relies on user feedback that

Table 7.6: Evaluation results on TREC CAsT 2022 compared to the baselines. ⋆, †, and ‡ denote a performance that is significantly different compared to our hybrid method for generating and selecting clarifying questions (T5MI+GTR+T5Ranking), (m) Selection only baseline, and (n) Generation only baseline (paired t-test, $p < 0.05$), respectively; The highest value for each measure is highlighted.

	Approach	**Asking Clarifying Questions**		
		Relevance@1	Novelty@1	Diversity@1
	generation baselines			
(a)	T5 raw	0.232⋆‡	0.166⋆‡	0.185⋆‡
(b)	T5 rewrite	0.320⋆‡	0.229⋆‡	0.210⋆‡
(c)	GPT-3 raw	0.433⋆‡	0.263⋆‡	0.356
(d)	GPT-3 rewrite	0.454⋆	0.346⋆	0.371
(e)	GPT-3 full context	0.119⋆‡	0.073⋆‡	0.082⋆‡
	selection baselines			
(f)	BM25	0.345⋆	0.293⋆	0.307
(g)	miniLM-BERT	0.371⋆	0.317⋆	**0.395**
(h)	ANCE	0.253⋆†	0.166⋆†	0.190⋆†
(i)	monoT5	0.247⋆†	0.181⋆†	0.151⋆†
(j)	ColBERT	0.217⋆†	0.171⋆†	0.137⋆†
(k)	TCT-ColBERT	0.186⋆†	0.132⋆†	0.122⋆†
	ours.			
(m)	Selection only	0.356⋆	0.284⋆	0.222⋆
(n)	Generation only	0.543	0.434	0.336
	Hybrid (T5MI+GTR+T5Ranking)	**0.567**	**0.494**	0.369

isn't always immediately available.

In answer to RQ 7.1, we find that our MTL T5MI model through Multi-Task Learning by sharing the tasks of clarification need identification and the generation of clarifying question is overall effective.

7.4.2 RQ 7.2: Quality of Clarifying Questions

In this section, we investigate the performance of our hybrid method (T5MI+GTR+T5Ranking) for generating and selecting clarifying questions (denoted by Hybrid in Table 7.6), in comparison to the baselines (a)-(n) (described in Section 7.3.2) on the test sets of the CAsT 2022 dataset for asking clarifying questions. Table 7.6 presents the evaluation results, comparing our proposed hybrid method in comparison with various baselines (see Section 7.3.2). The focus of this section is on the left-half of Table 7.6, which presents the evaluation results for asking clarifying questions. We evaluate the performance of various approaches using the P@1 metric (threshold 2), as per the CAST 2022 evaluation methodology, averaging across the utterances of 205 users. We calculate the metric based on three criteria: relevance (Relevance@1), novelty (Novelty@1), and diversity (Diversity@1), as described in Section 7.3.3.

- **Comparison of Our Proposed Hybrid Method with the Baselines (a)-(n):** From the table, we observe that our proposed hybrid method (T5MI+GTR+T5Ranking) achieves the highest performance, significantly outperforming all generation and selection baselines on Relevance@1 and Novelty@1. However, for Diversity@1, the baseline (g) miniLM-BERT is the highest performing; however, there is no significant difference between this baseline and our proposed $ORConvQA_{3:CNC+Askng}$ method. In addition, our proposed hybrid method (T5MI+GTR+T5Ranking) significantly outperforms the baselines (a), (b), (e), and (h)-(m) in terms of Diversity@1 (paired t-test, $p < 0.05$). Comparing our proposed hybrid method (T5MI+GTR+T5Ranking) and its variants (see Section 7.3.2) , the results in Table 7.6 show that our proposed hybrid method (T5MI+GTR+T5Ranking) outperforms both the (m) 'Selection only' and (n) 'Generation only' variant methods on all evaluation metrics criteria, with the improvement being significant compared to the (m) 'Selection only' baseline. We further analyse the results to determine the reason for the lack of a significant difference between our proposed $ORConvQA_{3:CNC+Askng}$ method and the baseline (n) 'Generation only'. By doing this, in Section 7.4.5, we conduct an analysis that qualitatively highlights the differences between the generated and selected clarifying questions within our proposed system.

- **Comparison of Our (n) 'Generation only' Variant with Other Generative Baselines (a)-(e):** We observe that our (n) 'Generation only' baseline (our generation for clarifying question T5MI model) exhibits a better performance than the other generative baselines (a)-(e), significantly outperforming them (indicated by ‡) in terms of Relevance@1, Novelty@1, and Diversity@ (except (d) 'GPT-3 rewrite' on all evaluation criteria and (c) 'GPT-3 raw' on Diversity@1). These results show that the Multi-Task Learning T5MI model outperforms the Single-Task Learning baselines (a)-(e), which are based on the T5 and GPT-3 models. This is exemplified by the Relevance@1 score of our T5MI model, which is 19.6% higher than that of the best performing Single-Task Learning baseline ((d) GPT-3 rewrite), and by the Novelty@1 score, which is 25.4% more than the corresponding score of the same baseline. These comparisons show that the advantage of using the Multi-Task architecture is that it can learn to share information between the two combined tasks, allowing the model to generate more informative and relevant clarifying questions. We also analyse why there is no significant difference between our (n) 'Generation only' (T5MI), GPT-3-Rewrite (wrt. Novelty@1 and Diversity@1), and GPT-3-Raw (Diversity@1). This is likely explained in that models based on GPT-3 generate more diverse and novel text, due to their 175B parameters, compared to our T5MI model, which has 220× fewer parameters (770M). In addition, Table 7.6 shows that the baseline (e) GPT-3-full-context' has the lowest performance on all evaluation criteria. A closer examination of the outputs from the baseline (e) GPT-3-full-context' provides insights into this result. In particular, of the 205 responses generated by this baseline, only 28 are classified as clarifying questions by

Table 7.7: Effectiveness of different input sequences for rewriting the current user utterance u_k using the T5QR model. † denotes a performance that is significantly worse than the input sequence $H_k; c_k; u_{k\_2}; u_k$ (paired t-test, $p < 0.05$); The highest value for each measure is highlighted.

Context	Sequence	NDCG@3	MAP	MRR
Full context	$H_k; c_k; u_{k\_2}; u_k$	**0.359**	**0.209**	**0.551**
	$H_k; u_k; c_k; u_{k\_2}$	0.295†	0.162†	0.463†
Clarifying questions and user feedback	$c_1; r_1; ...; c_k; u_{k\_2}; u_k$	0.323†	0.191†	0.527†
	$c_1; r_1; ...; u_k; c_k; u_{k\_2}$	0.271†	0.146†	0.445†

the organisers, which typically contain a question mark (?). The remaining 177 responses are identified as direct answers to the user's utterances rather than asking the questions for clarification. This result explains the baseline (e)'s lower performance in comparison to other models.

- **Comparison of Our (m) 'Selection only' Variant with Other Existing Dense Retrievers Baselines (h)-(k):** We find that the (m) 'Selection only' baseline (our selection for clarifying question GTR model) significantly outperforms (indicated by ‡) other existing dense retrievers and the state-of-the-art ranking baselines (h)-(k) on all evaluation criteria. In particular, our 'Selection only' method shows an improvement up to 91.4% in Relevance@1, 115.2% in Novelty@1, and 81.9% in Diversity@1 over the baseline (k) TCT-ColBERT. These findings demonstrate the GTR model's benefits and effectiveness compared to existing dense retrievers and state-of-the-art reranking baselines for selecting clarifying questions.

Therefore, in response to RQ 7.2, we find that our hybrid method (T5MI+GTR+T5Ranking) for generating and selecting clarifying questions has the best overall effectiveness, yielding statistically significant improvements in terms of Relevance@1 and Diversity@1 over the strong baselines (a)-(k) on the CAsT 2022 dataset. In addition, the use of Multi-Task Learning is shown to improve the performance of the T5MI model in asking clarifying questions, compared to the performance of single-task learning generative baselines like GPT-3 and T5.

7.4.3 RQ 7.3: Rewriting the Current Utterance

Next, we perform a comparison of the effectiveness of four different input sequences for rewriting the current user utterance u_k using the T5QR model (see Section 7.2). A successful rewriting will result in improved performances on passage ranking. The input sequences examined in this study include the use of all previous user utterances and system responses ($H_k = [\langle u_i, r_i \rangle] i = 1^{k-1}$), or the use of all previous clarifying questions and user responses ($c_{1:k-1}; u_{1\_2:k-1\_2}$) in combination with the relative position of the current user utterance u_k and the clarifying question c_k. Table 7.7 presents the results of our experiments, where the impact of the various input sequences upon passage ranking is evaluated using NDCG@3, MAP, and MRR (see Section 2.7).

The results from the table reveal that the input sequence consisting of all previous user utterances and system responses, followed by the clarifying question, user response, and the current user utterance ($H_k; c_k; u_{k\_2}; u_k$), yields the highest performance for all three measures, with an NDCG@3 of 0.359, MAP of 0.209, and MRR of 0.551. This input sequence significantly outperforms the other input sequences (indicated by †) in terms of NDCG@3, MAP and MRR (paired t-test, $p < 0.05$). In addition, these results indicate that the input sequence using all previous user utterances and system responses outperforms an input sequence that employs all previous clarifying questions and user responses. We postulate that when rewriting the current user utterance u_k, the use of all previous user utterances and system responses in the input sequence of the T5QR model allows the model to benefit from the context and information provided by these previous system responses. Furthermore, our results show that the input sequence that includes the current user utterance u_k at the end performs better than the sequence that appends the user response u_{k_2} at the end. This is likely due to the fact that when the user response u_{k_2} is appended at the end of the input sequence, the T5QR model focuses on it instead of the current utterance u_k, hence it does not provide an accurate rewrite of the current user utterance. For instance, consider the following dialogue: the current utterance u_k is “Interesting. Why is the A50 better?", to which the system responds with a clarifying question c_k of "Would you like to see reviews on the Galaxy A50?", and the user then replies with "yes, I’m very interested about it". The rewritten utterance u_r for the input sequence that appends the user response u_{k_2} at the end is "Yes, I’m very interested about the Galaxy A50." while the current user utterance u_k that would be appended is "Why is the Galaxy A50 better than the Moto G7 and Moto G7 Power?".

In response to RQ 7.3, we conclude that the input sequence $H_k; c_k; u_{k_2}; u_k$ is the most effective for rewriting the current user utterance using the T5QR model. This sequence yields the highest values for NDCG@3, MAP, and MRR and produces statistically significant improvements compared to other input sequences. Hence, to address RQ 7.4, we employ the input sequence $H_k; c_k; u_{k\_2}; u_k$ with the T5QR model for rewriting the current user utterance u_k and compare it against the baselines for passage retrieval.

7.4.4 RQ 7.4: Effectiveness on Conversational Search

We now investigate the passage retrieval performance of our proposed $ORConvQA_{3:CNC+Askng}$ method (as defined in Equation (7.2)) and the baselines (a)-(g) (described in Section 7.3.2) on the CAsT 2022 test set (see Section 3.7.2). Table 7.8 shows the effectiveness of various clarifying question approaches when applied to the T5QR model, using the input sequence $H_k; c_k; u_{k\_2}; u_k$ (as described in Section 5.3). The results are obtained from a passage retrieval pipeline, outlined in Section 4.3. The last row of the table shows the effectiveness of the automatically rewritten questions without using clarifying questions. Unfortunately, we cannot report results for the baselines (h) to (n), since TREC CAsT 2022 has only released the user feedback u_{k_2} (see

Table 7.8: Evaluation results on TREC CAsT 2022 compared to the baselines. ⋆ denotes a performance that is significantly different compared to our proposed *ORConvQA*$_{3:CNC+Askng}$ method (paired t-test, $p < 0.05$); The highest value for each measure is highlighted.

	Approach	**Retrieval**			
		NDCG@3	MAP	MRR	Recall
	generation baselines				
(a)	T5 raw	0.260⋆	0.133⋆	0.379⋆	0.491⋆
(b)	T5 rewrite	0.255⋆	0.134⋆	0.386⋆	0.501⋆
(c)	GPT-3 raw	0.348	0.194	0.517	0.526
(d)	GPT-3 rewrite	0.343	0.199	0.518	0.539
(e)	GPT-3 full context	0.343	0.200	0.530	0.535
	selection baselines				
(f)	BM25	0.325	0.192	0.503	0.525
(g)	miniLM-BERT	0.328	0.178⋆	0.511	0.511⋆
	ours.				
	ORConvQA$_{3:CNC+Askng}$	**0.359**	**0.209**	**0.551**	**0.545**
	without using clarifying questions.				
	User's Utterances	0.355	0.202	0.542	0.541

Section 7.3.1) for the top-ranked question in each turn of the submitted run. Note that the omission of results for models (h-n) does not prevent the RQ2 from being analysed.

Comparison of Our Proposed *ORConvQA*$_{3:CNC+Askng}$ Method with the Baselines (a)-(g) for Passage Retrieval. On analysing Table 7.8, we observe that our *ORConvQA*$_{3:CNC+Askng}$ method outperforms all the baselines (a)-(g) on all passage ranking evaluation measures. In addition, our proposed *ORConvQA*$_{3:CNC+Askng}$ method demonstrates a significant improvement over the T5-based baselines (a) and (b) and also significantly outperforms miniLM-BERT (baseline (g)) in terms of MAP and Recall. Furthermore, our proposed *ORConvQA*$_{3:CNC+Askng}$ method stands out as the only one with an improved performance compared to the automatically rewritten utterances without using clarifying questions (albeit not by a significant margin). These results raise the question as to why our proposed *ORConvQA*$_{3:CNC+Askng}$ method does not outperform the baselines (c)-(f). As explained in Section 7.3.1, Figure 7.3, in the TREC CAsT dataset (see Section 3.7.2), the user utterance evaluation is based on sub-trees, and in particular when a user utterance involves multiple sub-trees, the assessment for passage retrieval is based solely on the first sub-tree.

The evaluation of an approach's effectiveness in asking clarifying questions on the passage retrieval task is consequently limited, since it only considers a subset of the clarifying questions and ignores the corresponding user feedback. Hence, evaluating an approach for asking clarifying questions on passage retrieval does not take into account a significant portion of the clarifying questions and their corresponding user feedback. On closer inspection, we observe that negative user feedback such as "I don't know" or "not related to my search", impacts the T5QR query rewriting model. Indeed, when receiving negative feedback, the T5QR model may fail to

incorporate the clarifying questions and feedback to improve the user's utterance, resulting in similar rewritten queries to those without using clarifying questions. This leads to little differences between baselines (c)-(f) compared to using the automatically rewritten utterances without clarifying questions. Hence, as a further investigation, in Section 7.4.6, we present an analysis of the number of users' negative feedback received for each baseline.

In answer to RQ 7.4, we conclude that our proposed $ORConvQA_{3:CNC+Askng}$ method does help to improve the overall passage retrieval performance of a mixed-initiative conversational search system, yielding statistically significant improvements over the baselines (a) - (b) and (g) on TREC CAsT 2022.

7.4.5 Comparative Analysis

In this section, we first aim to evaluate the ranking performance of our generated clarifying questions compared to those selected by our clarifying question selection component. We also delve into a comparative study on how the generated and selected clarifying questions differ in terms of Diversity, Novelty, and Relevance, as presented in Table 7.6 in Section 7.4.2. Firstly, on inspection of the results, we observe that, out of 176 generated clarifying questions, the T5ranking model is the most effective for 94, thus further highlighting its superior performance.

Secondly, in terms of Diversity, the generated questions outperform the selected ones, with a count of 49 versus 19, indicating their wider range. Novelty is a closer contest, with 41 instances for the generated questions compared to 42 selected by our selection for clarifying question GTR component, implying a near-equal potential for introducing unique aspects. In terms of Relevance, the generated questions take the lead with 63 instances over 25 selected by our component, which could reflect a more direct relationship of the generated queries to the user's information needs.

In conclusion, this section highlights the effectiveness of generated clarifying questions, demonstrating superiority in ranking and diversity. However, novelty remains similar, affirming the value of selecting questions. These findings underline the potential of a hybrid approach for producing clarifying questions.

7.4.6 User Feedback Analysis

Following the results in Section 7.4.4, we perform an analysis of the negative feedback received from the users in the TREC CaST evaluation for our hybrid method (T5MI+GTR+T5Ranking) as well as for each baseline (a)-(f). Indeed, the evaluation Section 7.4.4 is based on effectiveness and does not consider if there is any implied sentiment in the utterance a user makes in response to a clarifying question. Indeed, we argue utterances containing positive feedback can be indicative of the usefulness of the corresponding clarifying question in the conversation. Hence, an effective approach for asking clarifying questions should exhibit more positive sentiment in user's utterances, and less negative sentiment.

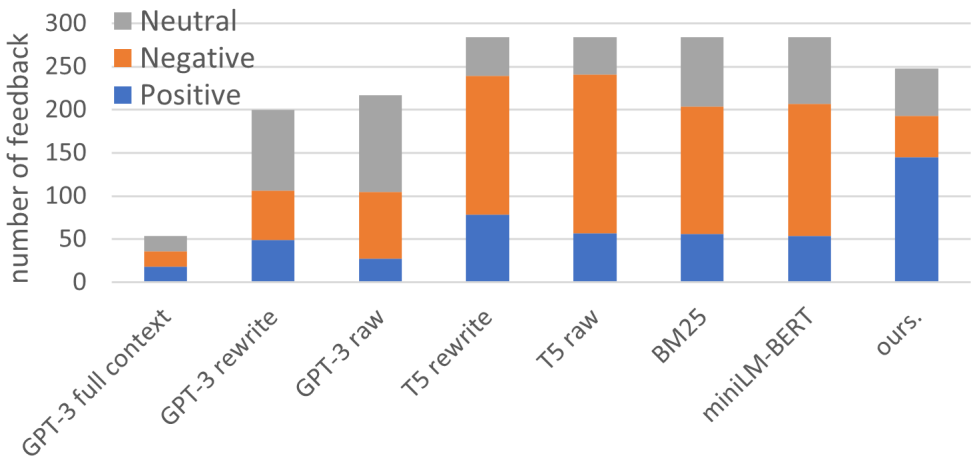


Figure 7.5: Comparison of Sentiment Analysis Results on different asking clarifying questions approaches.

To separate the positive and negative user feedback, we fine-tune the T5-base model using 239 manually annotated examples of user feedback from the CAsT 2021 (see Section 3.7.2) dataset. The model is fine-tuned to classify the user's response to a clarifying question by generating the tokens "positive", "negative", or "neutral". For example, a positive sentiment is predicted for an utterance such as "Yes, I'm very interested in it", while a negative sentiment is predicted for "I can't understand you". On the other hand, an utterance like "I'd like to know how to make lotion at home" is predicted as neutral. Figure 7.5 shows the distribution of feedback ("positive", "negative", and "neutral") across the different compared approaches, as predicted by the T5 classifier. From the figure, it can be seen that the feedback received by our hybrid method (T5MI+GTR+T5Ranking) has the highest number of positive feedback (145), followed by T5 rewrite (78). The lowest number of positive feedback is found for GPT-3 raw (18). Regarding negative feedback, the highest number of negative feedback are received by users for T5 raw (184), followed by T5 rewrite (161) and miniLM-BERT (153). On the other hand, the lowest number of negative feedback was noted for our hybrid method (T5MI+GTR+T5Ranking) (48), confirming its overall good performance compared to other approaches. For the neutral sentiment, the highest number of neutral feedback is observed for GPT-3 raw (112), followed by GPT-3 rewrite (94) and BM25 (80). The lowest number of neutral feedback is noted for T5 raw (43). Overall, these results confirm that our hybrid method (T5MI+GTR+T5Ranking) improves the user's predicted satisfaction with the clarifying questions – compared to other baseline methods – based on the users' sentiment expressed in response to the asked clarifying questions.

7.5 Conclusions

In this chapter, to further effectively address the ambiguities in conversational questions, we proposed an $ORConvQA_{3:CNC+Askng}$ method (as defined in Equation (7.2)), including a hybrid

method (T5MI+GTR+T5Ranking) for generating and selecting clarifying questions, in mixed-initiative conversational search, combining the strengths of clarifying question generation and selection. To do so, we leveraged Multi-Task Learning to simultaneously determine clarification needs and generate clarifying questions called T5MI, and GTR to select questions from the pool. The candidate questions from both approaches are then scored using a text generation model for point-wise question classification called T5Ranking. Our proposed $ORConvQA_{3:CNC+Askng}$ method directly addressed the issue presented in Gap 3, where we stated that prior works have not focused on a comprehensive approach for asking clarifying questions. Instead, they have either solely generated clarifying questions using generative models or selected them from pre-determined pools. In addition, no prior work has effectively leveraged Multi-Task Learning to simultaneously determine when clarifying questions are needed and generate relevant questions based on the user's initial query and conversation history. In particular, our investigation in this chapter aimed to answer four research questions as follows:

We first examined the effectiveness of our T5MI model on clarification need classification (RQ 7.1). Table 7.5 showed that our T5MI model through Multi-Task Learning by sharing the tasks of clarification need classification and the generation of clarifying questions was overall effective. As a result of its demonstrated effectiveness, we have integrated the MTL T5MI model into our proposed $ORConvQA_{3:CNC+Askng}$ method, leveraging its strengths in clarification need classification and generation for clarifying question to enhance the overall performance of our mixed-initiative system to answer RQ 7.2. Next, we showed the effectiveness of our hybrid method (T5MI+GTR+T5Ranking) in asking clarifying questions, comparing it with both existing generation and selection baselines for asking clarifying questions (see Table 7.6). These experimental results showed that our hybrid method (T5MI+GTR+T5Ranking) had the best overall effectiveness, yielding statistically significant improvements over the strong baselines. In addition, the use of joint learning was shown to improve the performance of the T5MI model in asking clarifying questions, compared to the performance of Single-Task Learning generative baselines like GPT-3 and T5 (see Section 3.6.4). In Section 7.4.3, we showed that the input sequence $H_k; c_k; u_{k_2}; u_k$ is the most effective for rewriting the current user utterance using the T5QR model (see Table 7.7). Therefore, we employ the input sequence $H_k; c_k; u_{k\_2}; u_k$ with the T5QR model for rewriting the current user utterance u_k and compare it against the baselines for passage retrieval in response to RQ 7.4. Table 7.8 showed that our proposed $ORConvQA_{3:CNC+Askng}$ method, which includes our hybrid method (T5MI+GTR+T5Ranking) for generating and selecting clarifying questions, improved the overall passage retrieval performance of a mixed-initiative conversational search system, yielding statistically significant improvements over the baselines (a) - (b) and (g). Our findings in Section 7.4.5 showed that our generating clarifying questions is more effective than selecting clarifying questions, especially in terms of ranking and diversity. This underscores the benefits of using a hybrid approach for producing clarifying questions. Moreover, in Section 7.4.6, we also performed an analysis of the negative

feedback received from the user for our proposed $ORConvQA_{3:CNC+Askng}$ method for asking clarifying questions as well as for each baseline (a)-(f) following the results in Section 7.4.4. Figure 7.5 presented that our proposed $ORConvQA_{3:CNC+Askng}$ method improves the user's predicted satisfaction with the clarifying questions – compared to other baseline methods – based on the users' sentiment expressed in response to the asked clarifying questions.

In summary, in this chapter, we further validated the hypothesis of our proposed thesis statement in Section 1.3 that leveraging MTL can be used to generate more effective clarifying questions by jointly learning Clarification Need Classification and clarifying question generation. In this chapter, we concluded that our MTL T5MI model showed marked improvements in clarification need identification compared to the baselines. For asking clarifying questions, our hybrid method (T5MI+GTR+T5Ranking) for generating and selecting clarifying questions, showed the best overall effectiveness in experiments on the TREC CAsT 2022 dataset, significantly outperforming existing strong baselines with improvements at P@1 by up to 20% on the relevance criteria and 30% on the novelty criteria. Furthermore, our proposed $ORConvQA_{3:CNC+Askng}$ method, which includes our hybrid method (T5MI+GTR+T5Ranking), also showed improvements in the overall passage retrieval performance of a mixed-initiative conversational search system.

In the next chapter, we conduct experiments to demonstrate the effectiveness of our monoQA model, which uses a text generation model with Multi-Task Learning for both the reranker and the reader. Our model, which is based on the T5 text generation model (see Section 2.3.1), is fine-tuned simultaneously for both reranking (in order to improve the precision of the top retrieved passages) and extracting the answer.

Chapter 8

monoQA: Multi-Task Learning of Reranking and Answer Extraction

8.1 Introduction

In Section 3.1, we discussed the challenges of effectively answering user questions in Open-Retrieval Conversational Question Answering (ORConvQA), including (1) resolving ambiguities in conversational questions; (2) retrieving and identifying the most relevant passages; and (3) extracting the relevant answer. In Chapters 6 and 7, we focused on improving the performance of the ORConvQA system by addressing the challenge of ambiguities in conversational questions. The analysis of experimental results in these chapters demonstrated that by leveraging Multi-Task Learning (MTL) and sharing the learner structure, our proposed MTL methods, namely *ORConvQA*$_2$ and *ORConvQA*$_3$, can improve performance on the ORConvQA task.

Unlike Chapters 6 and 7, which addressed the challenge of ambiguities in conversational questions, this chapter focuses on addressing the retrieval and identification of the most relevant passages, as well as the extraction of relevant answers. As previously described in Section 3.2, the ConvQA task consists in understanding the question based on a given conversational history, and extracting an answer from a given passage. This task is an extractive type of Question Answer (QA), meaning that the answer takes the form of a span in the provided passage, and can be successfully tackled by employing an extractive or generative *reader*. As previously described in Section 3.7, there has been more focus on retrieval as part of the ConvQA pipeline, known as Open-Retrieval Conversational Question Answering (ORConvQA). In this setting, the ORConvQA system needs to apply the ConvQA model upon passages retrieved from a large collection, given a question, before actually extracting the answer.

To address the ORConvQA task, prior works (see Section 3.7.4) have adopted a three-stage architecture, including a retriever, a reranker, and a reader to extract the answers. First, the retriever retrieves the top K relevant passages from the collection based on a question and its conversation history, as mentioned in Sections 3.7 and 3.8. The reranker and the reader then

respectively rerank and identify an answer in the top K passages. We also adopt this three-stage architecture in our proposed model in this chapter. However, in order to investigate the effectiveness of the reranker, we consider a two-stage pipeline including a retriever and a reader, as a baseline for comparison with our resulting system. For the retriever, as previously discussed in Section 3.7.4, due to the good effectiveness of bi-encoder dense retrievers, in particular, ConvDR (see Section 3.7.4), we adapt this type of retrieval models as our retriever. We also consider other recent existing bi-encoder passage retrievers such as TCT-ColBERT (see Section 2.4.2) and CQE (Lin et al. 2021*a*) as baseline passage retrievers.

Based on the literature reviewed in Sections 2.8 and 3.9, Multi-Task Learning (MTL) is described as a method where multiple different but related tasks are learned simultaneously. For instance, MTL has been employed in order to efficiently answer the questions posed by the users (see Section 3.2.4). In this manner, the network structure is shared between the reranker and the reader. By doing this, existing works (see Sections 3.8.2 and 3.2.4) have also typically approached reranking and *extractive* reading as classification tasks, with two fully-connected layers (one for the reranker and reader, respectively) added to find an answer span for the retrieved passages (start/end positions) as well as to predict the relevance score of the question to the passage as previously presented in Section 3.2.4. In this chapter, we use the Multi-Task Learning of the reranker and the *extractive* reader as our strongest baseline.

On the other hand, monoT5 (see Section 2.4.1), has been shown to outperform BERT-based models in passage reranking (Nogueira, Jiang, Pradeep & Lin 2020*b*). In addition, UnifiedQA (see Section 3.2.4) has been shown to yield impressive performances on many extractive QA datasets. However, to the best of our knowledge, as identified in Gap 4, no prior work has combined monoT5 and UnifiedQA by sharing a single text generation model, in order to directly extract the answers instead of predicting the start/end positions in a retrieved passage. In this chapter, we argue that a joint learning using Multi-Task Learning can enhance the learning efficiency and prediction accuracy of a model for the ORConvQA task, since by sharing the learning model the reranker and reader can simultaneously predict the answer and reranking score. Indeed, a joint learning by sharing a single model trained using MTL reduces memory needs and speeds up inference. Therefore, to address the limitation in Gap 4, we propose a $ORConvQA_{4:Reranker+Reader}$ MTL method. This method incorporates ConvDR as a retriever and a newly proposed MTL model, monoQA, which functions as both a reranker and a reader, as previously described in Section 4.4.4. In addition, we combine the effective monoT5 (to rerank the retrieved passages) and UnifiedQA (to extract the answer from the highest scored passage) models into a strong baseline. Moreover, for comparison with our MTL $ORConvQA_{4:Reranker+Reader}$ method, we investigate a different MTL method by combining the answer extraction with the passage retrieval, rather than with the passage reranker as in monoQA. By doing this, we also combine the conversational question rewriting task into a new $ORConvQA_{5:MTL3Tasks}$ method to address the ambiguities in the conversation question. Therefore, we propose an $ORConvQA_{5:MTL3Tasks}$ MTL method for

learning the conversational question rewriting, passage retrieval and answer extraction tasks simultaneously, as introduced in Section 4.4.4.

As stated in Section 1.3, we hypothesise that MTL can be used to enhance the performance of the ORConvQA task by sharing the learned structure of the reranker and reader in a single text generation model. This chapter aims to test this hypothesis by simultaneously learning the tasks of passage reranking and answer extraction. Our method aims to enhance performance on the ORConvQA task called $ORConvQA_{4:Reranker+Reader}$ (Section 4.4.4).

Our $ORConvQA_{4:Reranker+Reader}$ method includes a model called monoQA, which uses a text generation model with multi-task learning for both the reranker and reader. Our model, which is based on the T5 (see Section 2.3.1) text generation model, is fine-tuned simultaneously for both reranking (in order to improve the precision of the top retrieved passages) and extracting the answer. Unlike previous work, monoQA makes predictions by generating the first token for the passage reranking task, followed by the other tokens for the answer extraction task. In addition to $ORConvQA_{4:Reranker+Reader}$, we also introduce an $ORConvQA_{5:MTL3Tasks}$ MTL method, which employs a uniform model to jointly learn the conversational question rewriting, passage retrieval and answer extraction tasks simultaneously.

Our contributions are summarised as follows:

1. we leverage Multi-Task Learning with a text generation model, namely monoQA, by sharing the reranker and reader's learned structure to effectively address the ORConvQA task;

2. using two different ORConvQA datasets, we compare our $ORConvQA_{4:Reranker+Reader}$ method (ConvDR + monoQA), to two strong baselines from the literature, and show that our MTL reranker and generative reader approach yields the best F1, Recall, MRR, and MAP performance improvements over the strongest baseline with statistically significant improvements ranging from +5.70% to +23.34%;

3. the proposed MTL monoQA model, which is included in $ORConvQA_{4:Reranker+Reader}$ and combines the reranker and generative reader significantly outperforms and is twice as fast for inference than the individual application of the monoT5 and UnifiedQA models for reranking and extracting the answer.

4. in our $ORConvQA_{5:MTL3Tasks}$ method, we leverage Multi-Task Learning with a uniform model to jointly learn three tasks: conversational question rewriting, passage retrieval, and answer extraction, to effectively address the ORConvQA task.

5. we compare our $ORConvQA_{5:MTL3Tasks}$ method (MTL of three tasks) to existing strong baselines and show an improvement in question answering (QA) performance.

The rest of the chapter is structured as follows: Section 8.2 recalls the definition of the Open-Retrieval Conversational Question Answering (ORConvQA) task, along with our Three-Stage

Table 8.1: Notations used in Chapter 8.

Notation	Definition
q_k	A current question
q'_k	A rewritten question of the current question q_k
a_k	An answer to current question q_k
H_k	A conversation history i.e. $H_k = [\langle q, a \rangle]$
P^+_{ret}	A retrieved passage
C	A passage collection
W	A sequence of m words i.e. $W = \{w_1, w_2, ..., w_m\}$
n_k	An importance of (or the need for) asking a clarifying question (1-4)
$\hat{T}_k$	A contextualised token-level representations
$QR(\cdot)$	A Conversational Question Rewriting function
$Retriever$	A passage retriever function
$Reranker$	A passage reranker function
$Reader$	An answer extractor function

Pipeline for an ORConvQA system using our proposed MTL $ORConvQA_{4:Reranker+Reader}$ method (see Section 4.4.4). Our $ORConvQA_{4:Reranker+Reader}$ method makes use of ConvDR as a retriever and our monoQA model as an MTL of the ranker and reader. We present our experimental setup in Section 8.3 and show the results of the experiments in Section 8.4. Moreover, Section 8.5 describes applying the Multi-Task Learning of three tasks, namely conversational question rewriting, passage retrieval, and answer extraction, by sharing a single model. Finally, we provide concluding remarks in Section 8.6.

8.2 Three-Stage Pipeline for an ORConvQA System

In this section, we describe our three-stage pipeline for an Open-Retrieval Conversational Question Answering (ORConvQA) system. Our system makes use of passage retrieval, through our proposed MTL $ORConvQA_{4:Reranker+Reader}$ method (see Section 4.4.4), which includes ConvDR as a retriever and our monoQA model as an MTL of a reranker and reader. We first explain how these models are used to perform the passage retrieval, passage reranking, and answer extraction tasks in Section 8.2.1. An overview of the three-stage pipeline and our proposed MTL $ORConvQA_{4:Reranker+Reader}$ method and its models (ConvDR + monoQA) follows in Section 8.2.2. Then, we explain how to fine-tune the monoQA model in Section 8.2.3.

8.2.1 Task Definitions

In this chapter, we aim to tackle the tasks of passage retrieval (see Section 3.7), passage reranking (see Section 3.8), and answer extraction (see Section 3.2). Table 8.1 presents the notations, which include a subset of the symbols defined in Table 4.1 along with notations specific

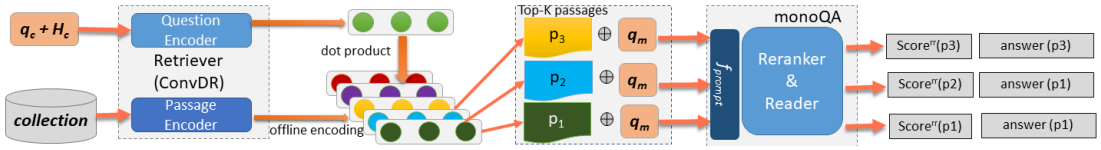


Figure 8.1: The overall framework of our ORConvQA system (consisting of ConvDR & monoQA).

to this chapter. The passage retrieval, passass reranking, and answer extraction tasks have been formalised in Equation (4.5), Equation (4.6), and Equation (4.7) of Chapter 4 as follows:

$$\begin{aligned} &Retriever(q_k, H_k, C) \rightarrow P^+_{ret} \\ &Reranker(q, P^+_{ret}) \rightarrow P^+_{reranked} \\ &Reader(q, P^+_{reranked}, \theta) \rightarrow a_k \end{aligned} \tag{8.1}$$

where $Retriever(\cdot)$ is a function that retrieves a ranked list of n passages $P^+_{ret} = [p_1, p_2, ..., p_n]$ from a passage collection C using the user's current question q_k and its conversation history H_k, $Reranker(\cdot)$ is a function that takes the retrieved passage P^+_{ret} and produces a ranked list $P^+_{reranked} = [p'_1, p'_2, ..., p'_n]$, and $Reader(\cdot)$ is a function that extracts the response answer a_k for the user. The passage retrieval task aims to retrieve a set of relevant passages or documents from a large corpus based on the user's questions. The passage reranking task aims to further rank the retrieved passages based on their relevance to the user's question to produce a more accurate and relevant ranked list. The answer extraction task aims to extract the final answer from the top-ranked passage identified. In the following section, we describe our proposed $ORConvQA_{4:Reranker+Reader}$ method and its models (ConvDR + monoQA) to address these tasks.

8.2.2 Models Overview

To tackle the tasks described in Section 8.2.1, as previously introduced in Section 4.3, our proposed $ORConvQA_{4:Reranker+Reader}$ method consists of four main components: (1) a conversational question rewriting;(2) a retriever for relevant passages retrieval; (3) a passage reranker for improving the precision of the top retrieved passages; and (4) a passage reader for generating the answer from the top retrieved passage, as defined in Equation (8.2) as follows:

$$\begin{aligned} ORConvQA_{4:Reranker+Reader} = QR(q_k, H_k) \\ \overset{q'_k}{\gg} Retriever(C, N) \\ \overset{P^+_{Ret}}{\gg} Reranker(q, N, \theta) \\ \overset{P^+_{Reranked}}{\gg} Reader(q, \theta) \end{aligned} \tag{8.2}$$

First, the conversational query rewriting component, $QR(\cdot)$, reformulates the current question

q_k and its context (conversation history) H_k into a standalone, omission-free rewritten question q'_k, which can be used in the later stages in a decontextualised manner. Then, the retriever retrieves the top K relevant passages from the collection based on the rewritten question q'_k as described in Figure 8.1. In order to produce the final answer, the reranker and reader then re-score and identify the answer in the top-K passages from the retriever. In doing so, a single model application gives an answer to both stages - i.e., whether this is a relevant passage and the position of the answer in the passage. In particular, we present ConvDR, a bi-encoder dense retrieval with MTL for conversational question rewriting and retriever, along with monoQA, which uses a text generation model with MTL for both the reranker and reader. We now describe each component of our $ORConvQA_{4:Reranker+Reader}$ method in detail.

8.2.2.1 ConvDR: Conversation Question Rewriting & Retriever

The left part of Figure 8.1 presents the conversation question rewriting and retriever components. As previously introduced in Section 3.7.4, we use a dual-encoder model named ConvDR, which consists of a question encoder and a passage encoder. Moreover, ConvDR uses a teacher-student mechanism, where a pre-trained dense retriever (the teacher) guides a query encoder (the student) to learn from teacher-generated embeddings on oracle reformulated questions (manually rewritten questions). This enables ConvDR to effectively integrate conversational question rewriting with passage retrieval tasks. Then, our $ORConvDR_4$ method, as presented in Equation (8.2), can be defined as:

$$\begin{aligned} ORConvQA_4 &= ConvDR(QR, Retriever) \\ &\overset{P^+_{Ret}}{\gg} Reranker(q, N, \theta) \\ &\overset{P^+_{Reranked}}{\gg} Reader(q, \theta) \end{aligned} \tag{8.3}$$

The ConvDR model first encodes the current question concatenated with all previous questions and passage into the embedding space:

$$\begin{aligned} E_q &= ConvDR(q_c; q_{1:c-1}), \\ E_p &= ConvDR(p) \end{aligned} \tag{8.4}$$

where q_c, $q_{1:c-1}$ and p denote the current question, the historical questions, and a passage, respectively. E_q and E_p are the embeddings of q_c concatenated with $q_{1:c-1}$ and p, respectively. ConvDR uses the dot product of E_q and E_p to calculate the retrieval score of a passage p for the current question q_c, with historical questions $q_{1:c-1}$:

$$score^{rt}(q_c; q_{1:c-1}, p) = E_q \cdot E_p \tag{8.5}$$

To retrieve the top K passages in the embedding space, ConvDR uses FAISS (Johnson et al. 2021), which is a library for efficient approximate nearest neighbour search. Yu et al. (2021) provides further details on ConvDR. Finally, we note that, in practice, a *rewritten* formulation q_r of the current question q_c can be used to resolve ambiguities such as coreference resolution.

8.2.2.2 monoQA: Reranker & Generative Reader

The right part of Figure 8.1 presents our proposed monoQA model, which uses T5 (see Section 2.4.1), a large pre-trained language model designed for text generation. To adopt an MTL approach to a text generation model for jointly learning from both passage reranking and answer extraction, the MTL monoQA model makes predictions by generating the first token for the passage reranking task and the follow-up tokens for the answer extraction task. In particular, when fine-tuning the T5 model for a downstream task, we use Prompt-based Learning (see Section 2.8.1), which is a method to modify the model by using a task-specific prompt together with the input (Liu et al. 2021). We deploy a T5 model to capture the relation between the rewritten question q_r of the current question q_c and the passage p as shown in the right part of Figure 8.1. In particular, we define a monoQA transformation function as $MTL_{\text{monoQA}}(\cdot)$ by taking the input sequence as follows:

$$MTL_{\text{monoQA}}(f_{prompt}(q_r, p)) \rightarrow w_1, w_2, ..., w_n \tag{8.6}$$

where $f_{prompt}()$ is a prompt function (template) to format q_r, and the passage into an input sequence for monoQA. The model is then fine-tuned to generate n target tokens, as shown in Equation (8.6), where the token w_1 is either "true" or "false"[1] depending on whether the passage is relevant or not to question q_r, while the follow-up tokens $w_2, ..., w_n$ are the output sequence for the answer of the question q_r. Then, our $ORConvDR_4$ method, as presented in Equation (8.3), can be defined as:

$$\begin{aligned} ORConvQA_4 &= ConvDR(QR, Retriever) \\ &\overset{P^+_{Ret}}{\gg} MTL_{\text{monoQA}}(q_r, \theta) \end{aligned} \tag{8.7}$$

At inference time, following Nogueira, Jiang, Pradeep & Lin (2020*b*), we apply a softmax only on the logits of the "true" and "false" tokens of the first generated token w_1 to calculate the reranker score as follows:

$$score^{rr} = \text{softmax}(w_1) \tag{8.8}$$

[1] We choose "true" and "false" as target tokens following monoT5 (Nogueira, Jiang, Pradeep & Lin 2020*b*).

8.2.3 monoQA Training

Given K retrieved passages (see Section 8.2.2.1) and a rewritten question q_r , our monoQA model jointly trains the reranker and the reader as follows:

Joint training: We consider how to fine-tune monoQA in order to generate the tokens for both passage reranking and answer extraction. In particular, the prompt function (Equation (8.6)) formats a question q and a passage p into an input sequence for monoQA and monoQA then outputs the contextual representation h. After that, the monoQA decoder takes the previously generated tokens as input and performs attention over h and then generates the next token. In particular, given a tuple $\langle q, p, a \rangle$, the training objective is to minimise the following loss function:

$$\mathcal{L}_{gen} = \sum_{i=1}^{M} \log P(a_i | h, a_{:i}) \tag{8.9}$$

where M is the number of tokens in the ground truth answer a, a_i is the i^{th} token in a, and a_0 is the beginning of sequence token (<s>).

We also consider *relevance accuracy* and word-level F1 (see Section 3.2.3) scores for selecting the best training model checkpoint during the evaluation step with the development set of the OR-QuAC dataset. Relevance accuracy is defined as the percentage of correct predictions for the first token generated from the model. The target token is "true" (the passage is indeed relevant) or "false" (a non-relevant passage). Following Qu et al. (2020), the word-level F1 is calculated by first removing stopwords and then considering the overlapping portion of the words in the prediction and ground truth answer.

Prompt: As previously introduced in Section2.8.1, Prompt-based Learning, is a method to tailor pre-trained language models to downstream tasks by using a task-specific prompt together with the input. To fine-tune the monoQA model for passage reranking and answer extraction, we use Prompt-based Learning to modify the model input. By doing this, we investigate several prompts in previous works (Khashabi et al. 2020, Nogueira, Jiang, Pradeep & Lin 2020*b*), and for completeness we evaluate all of them in order to choose the most effective. Details of the Prompt-based Learning and their corresponding experiments and results are provided in Section 8.4.2. We do not investigate Prompt-based Learning for question reformulation since our main contribution focuses on leveraging the output of a generative model for re-ranking and reading.

Positive and negative passages: Selecting positive and negative passages is a crucial step for training monoQA. For instance, passages relevant to a question are provided in the ORConvQA task. All other passages in the collection, which are unjudged, can be viewed as non-relevant by default. To cope with this issue, following Yu et al. (2021), we employ a hard negative sampling technique by randomly selecting the negative passage p^- for the question q from the top K retrieved passages by ConvDR. For training monoQA, the output sequence for the positive passage p^+ begins with the token "true" followed by the ground truth answer; the output sequence

for the negative passage p^- begins with the token "false" followed by "CANNOTANSWER".
Model initialisation: We consider the use of different models to initialise monoQA during training, since we propose to combine monoT5 and UnifiedQA to share a single text generation model. Moreover, both monoT5 and UnifiedQA are fine-tuned based on the `t5-base` model. Therefore, we investigate which of monoT5, UnifiedQA, and `t5-base`, are suitable for initialising monoQA (see details in Section 8.4.3).

Next, we evaluate our MTL $ORConvQA_{4:Reranker+Reader}$ method (ConvDR + monoQA) in comparison to several existing baselines as detailed in the next section.

8.3 Experimental Setup

Our experiments address the six following research questions:

RQ 8.1 How to select the best training model checkpoint in validation steps, i.e. the best validation loss, the best word-level F1, or the best relevance accuracy?

RQ 8.2 Which of the prompts namely: (a) monoT5 prompt ; (b) UnifiedQA prompt; and (c) our prompt; lead to the best performance of our monoQA model?

RQ 8.3 Which model to use for initialising monoQA, namely which of: monoT5, UnifiedQA, and `t5-base`, lead to the best performance of our monoQA model on the ORConvQA task?

RQ 8.4 How does our propose $ORConvQA_{4:Reranker+Reader}$ method (ConvDR + monoQA) compare to other existing baselines, namely:

(1). the ORConvQA system proposed by Qu et al. (2020) (see Section 3.7.4);

(2). the ORConvQA system proposed by Qu et al. (2020) but using ConvDR (see Section 3.7.4) as a retriever;

(3). using ConvDR as a retriever, monoT5 (see Section 2.4.1) as a reranker, and UnifiedQA (see Section 3.2.4) as a reader?

RQ 8.5 How does our proposed $ORConvQA_{4:Reranker+Reader}$ method (ConvDR + monoQA), which is a three-stage pipeline (retriever, reranker, and reader), compare to the two-stage pipeline baselines (retriever and reader)?

RQ 8.6 How does our proposed $ORConvQA_{5:MTL3Tasks}$ method, which is the MTL of conversational question rewriting, retriever, and reader using a uniform model, compare to the two-stage pipeline (retriever and reader) baselines?

Table 8.2: Statistics of the used datasets

Dataset	Items	Train	Dev	Test
CANARD	Dialogs	4,383	490	771
	Questions	31,526	3,430	5,571
OR-QuAC	Dialogs	4,383	490	771
	Questions	25,824	2,808	4,406
	documents	∼5.9 million		
OR-CoQA	Dialogs	1,521	100	100
	Questions	23,027	1,494	1,611
	documents	∼5.9 million		

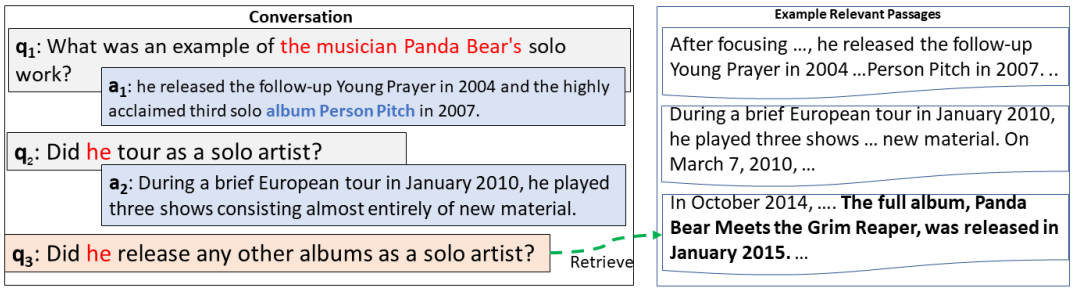


Figure 8.2: An example dialog and relevant passages from the ORConvQA dataset (Qu et al. 2020).

8.3.1 Datasets

To conduct our evaluation of our proposed $ORConvQA_{4:Reranker+Reader}$ method (ConvDR+monoQA), we choose the OR-QuAC and OR-CoQA datasets, which are extractive Question Answering (QA) datasets, as previously described in Section 3.7.2. However, the OR-CoQA dataset can be also considered as a generative QA dataset because it contains both span and freeform answers. Indeed, in this chapter, we focus on extractive QA only since we train our monoQA only on the OR-QuAC training set. In addition, following (Qu et al. 2021), we remove unanswerable questions from both datasets.

As exemplified in Figure 8.2, a question in either OR-QuAC and OR-CoQA can be ambiguous and difficult to understand without its context (e.g., q3: "Did he release any other albums as a solo artist?"). At training time, we fine-tune monoQA by using a manually rewritten query (q_r), which is provided by the OR-QuAC dataset. Thereafter, at inference time, following Dalton et al. (2020, 2021), Lin, Yang, Nogueira, Tsai, Wang & Lin (2020*a*), we employ another T5 model trained using the CANARD dataset to rewrite the OR-QuAC and OR-CoQA test set questions into context-independent questions that can be used as input for monoQA.

To validate our proposed $ORConvQA_{4:Reranker+Reader}$ method, during training our monoQA model, we use the OR-QuAC development set by selecting only positive examples (ground truth consisting of an answer and the corresponding passage for the question) after removing the unanswerable questions following (Qu et al. 2021). This development set consists of 490

Table 8.3: List of baselines.

	Retriever	Reranker	Reader
Three-stage pipeline (retriever, reranker, and reader)			
(a)	bi-encoder (ALBERT)	BERT (MTL of reranker and extractive reader)	
(b)	ConvDR	BERT (MTL of reranker and extractive reader)	
(c)	ConvDR	monoT5	UnifiedQA
Two-stage pipeline (retriever and reader)			
(d)	ConvDR	-	monoQA (reader)
(e)	ConvDR	-	UnifiedQA
(f)	TCT-ColBERT	-	UnifiedQA
(g)	CQE (Lin et al. 2021*a*)	-	ORConvQA (reader)
($ORConvQA_{4:Reranker+Reader}$)	ConvDR	monoQA (MTL of reranker and generative reader)	

dialogues with 2808 questions in total. For further information about the used datasets, we also provide a summary of their statistics in Table 8.2.

8.3.2 Baselines and Implementation Details

Baselines: To demonstrate the effectiveness of our proposed $ORConvQA_{4:Reranker+Reader}$ method (ConvDR+monoQA), we compare it with the seven baseline systems listed as ***(a)***-***(g)*** in Table 8.3

Three-stage pipelines:

(a) The first ORConvQA system has been proposed by Qu et al. (2020) (see Section 3.7.4). It adopts a duo-ALBERT encoder as the retriever and an MTL of the reranker and reader by sharing a BERT encoder. We make use of the code provided by Qu et al. (2020);

(b) This baseline is adapted from (a) by replacing the duo-ALBERT encoder passage retriever (see Section 3.7.4) with ConvDR (see Section 3.7.4) in a similar manner to our proposed $ORConvQA_{4:Reranker+Reader}$ method (consisting of ConvDR and monoQA). This is a important baseline to compare with our monoQA model in order to evaluate the reranker and reader performances. We reproduce the MTL of the reranker and reader models and its evaluation results provided by Qu et al. (2020);

(c) This baseline uses ConvDR as the passage retriever similarly to our $ORConvQA_{4:Reranker+Reader}$ method, monoT5 (see Section 2.4.1) as the passage reranker, and UnifiedQA (see Section 3.2.4) as the passage reader. It is deployed by using three models in the pipeline for comparison with our $ORConvQA_{4:Reranker+Reader}$ method (ConvDR+monoQA). This comparison is done in order to evaluate the performance of using monoT5 and UnifiedQA separately in comparison with the joint learning of the reranker and reader (monoQA).

Two-stage pipelines:

(d) This baseline uses ConvDR as the passage retriever, and our monoQA reader as the passage reader without using the reranking results from the monoQA reranker. The reader directly identifies an answer in the top passage from the retriever;

(e) This baseline uses ConvDR as the passage retriever and UnifiedQA as the passage reader.

(f) This baseline uses TCT-ColBERT (see Section 2.4.2) as the passage retriever and UnifiedQA as the passage reader;

(g) This baseline uses CQE (Lin et al. 2021*a*) as the passage retriever and UnifiedQA as the passage reader. The results of this baseline come from the previous work of Lin et al. (2021*a*).

Hyperparameter settings: For ConvDR, we reproduce the model and its evaluation results provided by Yu et al. (2021) to generate the offline passage embeddings from the passage collection of the OR-QuAC dataset as shown in Figure 8.1. We implement the monoQA model using the following PyTorch models from HuggingFace (Wolf et al. 2020), namely `t5-base`, `castorini/monot5-base-msmarco`, and `allenai/unifiedqa-t5-base`. Following Qu et al. (2020), these models are configured as follows: the maximum sequence length is set to 512, the number of training epochs is set to 10, the batch size is set to 16, and the learning rate is set to $5e^{-5}$. The models are trained on a NVIDIA RTX A6000. The average training time of monoQA is 6.3 hours. The number of parameters in monoQA is approximately 222 million parameters, i.e. the same as monoT5 and other fine-tuned versions of `t5-base`. We save the checkpoints every epoch and evaluate on the development set of the OR-QuAC dataset. We provide the details of how to select the best checkpoint in Section 8.4.1. For text generation, we use a beam search with a beam width of 5.

Evaluation metrics: Since we are using the OR-QuAC dataset, we naturally adopt the two evaluation metrics, namely the word-level F1, and the human equivalence score (HEQ). These evaluation metrics have been previously introduced in Section 3.2.3. To evaluate the retrieval performance, as previously described in Section 2.7, we use Mean Average Precision (MAP@10), Mean Reciprocal Rank (MRR@5) and Recall@5 as metrics for the reranker. For each query, the top 100 passages are considered. Finally, we use the McNemar's test to test statistical significance between the various readers' performances and the paired t-test for testing significant differences between the rerankers' performances.

8.4 Results and Analysis

We now address RQs 8.1-8.3 (see Section 8.3) and conclude with an efficiency analysis.

8.4.1 RQ 8.1: Selecting the Best Model

We investigate how to identify the optimal training checkpoint for our proposed monoQA model on the OR-QuAC development set. The monoQA model is trained on the OR-QuAC training set by using the positive p^+ and negative passages p^- described in Section 8.2.3. In particular, we use "Question Answering: {q} [sep] {p}" as the prompt function and monoT5 to

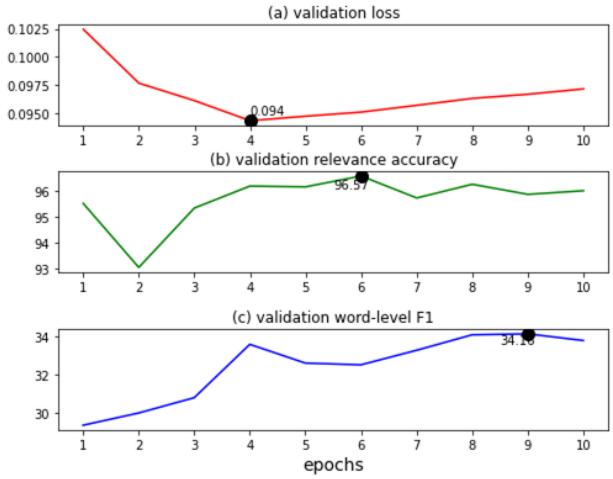


Figure 8.3: The validation scores of (a) the loss, (b) the relevance accuracy, and (c) the word-level F1, for each validation step (epochs). • denotes the best number of epoch of each score. The best number of epochs of the model on loss, relevance accuracy, and word-level F1 scores, are 4, 6, and 9, respectively.

initialise monoQA. In this section, we consider the performance of each model checkpoint on both reranking and answer extraction.

We identify the best checkpoint of the model for each measure, namely validation loss, validation relevance accuracy, and word-level F1 as discussed in Section 8.2.3. Figure 8.3 shows the best epochs of the model in terms of validation loss, relevance accuracy, and word-level F1 scores, which are 4, 6, and 9, respectively. We then evaluate the models obtained at these epochs (4, 6, and 9) on the OR-QuAC test set, as depicted in Figure 8.4. Figure 8.4 shows that the model checkpoint at epoch 9 has the best performance in terms of MAP@10, Recall@5, MRR@5, word-level F1, and HEQ-Q, whereas in HEQ-D the epoch 6 is the best. Indeed, the model that exhibits the highest word-level F1 on the validation set is also the best model when evaluated on the test set in terms of MAP@10, Recall@5, MRR@5, word-level F1, and HEQ-Q.

In response to RQ 8.1, we find that the model that archives the best validation word-level F1 score leads to the best performance model on the testing set. We further use the way to select the best model checkpoint for RQ 8.2.

8.4.2 RQ 8.2: Prompt-based Learning

To fine-tune the monoQA model for passage reranking and answer extraction, we adopt Prompt-based Learning (see Section 2.8.1) to modify the model input. We have observed that several prompts have previously been used in previous work (Khashabi et al. 2020, Nogueira, Jiang, Pradeep & Lin 2020*b*). Below we list the templates $f_{prompt}()$ that we consider in this chapter:

• *monoT5 prompt:* We adapt the monoT5's template by replacing the prefix word from "Query:" to "Question:", the separator token from "Document:" to "Passage:", without using the word

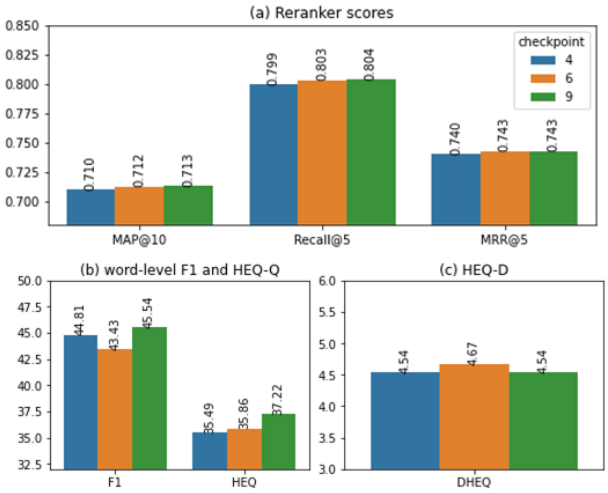


Figure 8.4: Results on the test set of the OR-QuAC dataset in terms of (a) MAP@10, Recall@5, and MRR@5, (b) word-level F1 and HEQ-Q, and (c) HEQ-D, of the models at epochs 4, 6, and 9.

Table 8.4: Effectiveness of various prompts for training monoQA.

Prompt	Retrieval			QA		
	MAP@10	Recall@5	MRR@5	F1	HEQ-Q	HEQ-D
monoT5	0.708	0.801	0.739	**45.6**	36.8	4.2
UnifiedQA	0.705	0.800	0.735	45.2	35.3	3.6
Our prompt.	**0.713**	**0.804**	**0.743**	45.2	**37.2**	**4.5**

"Relevant:":

$$\text{“}Question : \{q\}\ Passage : \{p\}\text{”} \tag{8.10}$$

• *UnifiedQA prompt:* UnifiedQA (Khashabi et al. 2020) made use of a '\n' between the current question and the passage:

$$\text{“}\{q\}\ \backslash n\ \{p\}\text{”} \tag{8.11}$$

However, under the standard T5 tokeniser, a whitespace such as '\n' does not result in a separate token, so the end result of this formulation is a simple concatenation of the question and passage:

$$\text{“}\{q\}\ \{p\}\text{”} \tag{8.12}$$

• *Our prompt:* For comparison with the above templates from the literature, we design a new template using "Question Answering:" as a prefix and a T5-provided special tokens ($[sep]$) as a separator token between the question and the passage:

$$\text{“}QuestionAnswering : \{q\}\ [sep]\ \{p\}\text{”} \tag{8.13}$$

Table 8.4 shows the results of each evaluated model for each of the above prompts. From the table, we see that the monoQA model trained by using our designed prompt (Question Answering:

Table 8.5: Effectiveness of various initialisations for training monoQA. † and ‡ denote a performance significantly worse than the model initialised using monoT5 and `t5-base`, respectively (McNemar's test, $p < 0.05$).

Model	Retrieval			QA		
Initialisation	MAP@10	Recall@5	MRR@5	F1	HEQ-Q	HEQ-D
monoT5	**0.713**	**0.804**	**0.743**	45.2	**37.2**	4.5
UnifiedQA	0.705	0.801	0.734	43.6†‡	34.9	**5.3**
`t5-base`	0.705	0.799	0.735	**45.4**	36.9	3.8

{q} [sep] {p}) has the highest performance on all measures, except when it uses the monoT5 prompt (Question: {q} Passage: {p}) for word-level F1.

Therefore, in response to RQ 8.2, we find that the model learned with our designed prompt has the best overall effectiveness. As a consequence, we use "Question Answering: {q} [sep] {p}" as the prompt for training monoQA that can be used in RQ 8.3.

8.4.3 RQ 8.3: Model Initialisation

In this section, we examine the effectiveness of the use of the models for initialising monoQA, namely monoT5, UnifiedQA, and `t5-base`, on the test set of OR-QuAC. All models are trained on the OR-QuAC training set by using positive passages p^+ and negative passages p^- as described in Section 8.2.3. In particular, we use "Question Answering: {q} [sep] {p}" as the prompt function because it performed the best according to the experiments in Section 8.4.2. Table 8.5 presents the results for each evaluated model on the retrieval and question answering (QA) metrics.

From the table, we see that training monoQA when initialised by monoT5 achieves the highest performance on the retrieval metrics (MAP@10, Recall@5, and MRR@5). However, there are no significant differences between all of the models' retrieval performances. For the QA performance, the best word-level F1, HEQ-Q, and HEQ-D scores are obtained by the models that use `t5-base`, monoT5, and UnifiedQA, respectively. In particular, in terms of word-level F1, initialising from monoT5 or `t5-base` significantly outperforms the model trained from UnifiedQA, but both monoT5 and `t5-base` initialisations lead to comparable performances.

Therefore, in response to RQ 8.3, we find that the monoQA model initialised from monoT5 has the best overall effectiveness, yielding statistically significant improvements in word-level F1 over using UnifiedQA on the test set of the OR-QuAC dataset. In the following, we use monoT5 to initialise monoQA for answering RQ 8.2 and comparing with the baselines.

8.4.4 RQ 8.4: Effectiveness of monoQA

We investigate the performances of our proposed Multi-Task Learning (MTL) method, $ORConvQA_{4:Reranker+Reader}$, which uses ConvDR as the retriever and monoQA as both the

Table 8.6: Evaluation results on OR-QuAC and OR-CoQA compared to the baselines. † denotes a performance significantly worse than our proposed *ORConvQA*$_{4:Reranker+Reader}$ (ConvDR+monoQA) method in terms of word-level F1 (McNemar's test, $p < 0.05$); ‡ denotes a performance significantly worse than our proposed monoQA model in terms of MAP@10, Recall@5, and MRR@5 (paired t-test, $p < 0.05$); The highest value for each measure is highlighted.

	Retriever	Reranker	Reader	OR-QuAC						OR-CoQA		
				Retrieval			QA			QA		
				MAP@10	Recall@5	MRR@5	F1	HEQ-Q	HEQ-D	F1	HEQ-Q	HEQ-D
	ConvDR (retriever only)	-	-	0.617	0.745	0.631	-	-	-	-	-	-
	Three-stage pipeline (retriever, reranker, and reader)											
(a)	bi-encoder (ALBERT)	BERT (MTL of reranker and extractive reader)		-	0.314‡	0.309‡	29.4†	23.7	1.3	-	-	-
(b)	ConvDR	BERT (MTL of reranker and extractive reader)		0.518‡	0.629‡	0.541‡	29.8†	22.8	2.3	28.1†	18.9	0
(c)	ConvDR	monoT5	UnifiedQA	0.590‡	0.727‡	0.618‡	22.2†	11.0	1.6	31.6†	**22.8**	0
	Two-stage pipeline (retriever and reader)											
(d)	ConvDR	-	monoQA (reader)	0.617‡	0.745‡	0.631‡	32.9‡	26.6	3.5	21.9‡	10.6	0
(e)	ConvDR	-	UnifiedQA	0.617‡	0.745‡	0.631‡	19.6‡	9.5	1.0	20.7‡	13.2	0
(f)	TCT-ColBERT	-	UnifiedQA	0.370‡	0.501‡	0.386‡	14.1‡	6.2	1.0	16.7‡	9.8	0
(g)	CQE (Lin et al. 2021*a*)	-	ORConvQA (reader)	-	0.415	0.310	32.0	-	-	-	-	-
(*ORConvQA*$_{4:Reranker+Reader}$)	ConvDR	monoQA (MTL of reranker and generative reader)		**0.713**	**0.804**	**0.743**	**45.2**	**37.2**	**4.5**	**37.3**	19.7	0

reranker and reader, in comparison to the baselines (a)-(c) (described in Section 8.3.2) on the test sets of the OR-QuAC and the OR-CoQA datasets. In Table 8.6, the first row shows the results of ConvDR as the retriever only and the last row shows the results of our proposed *ORConvQA*$_{4:Reranker+Reader}$ method. Table 8.6 (top-half) also shows the results of the existing baselines (a)-(c). In the table, on the OR-CoQA test set, we only include the question answering (QA) results since the gold passages for each question are not provided.

From the table, on the test sets of the OR-QuAC and OR-CoQA datasets, we observe that our proposed *ORConvQA*$_{4:Reranker+Reader}$ method achieves the highest performance by significantly outperforming all baselines on all measures, excepting the baseline using monoT5 as the reranker and UnifiedQA as the reader in terms of HEQ-Q on OR-CoQA. From the table, we also observe that the baselines (b) and (c) have lower retrieval performances compared to the results of using ConvDR as the retriever only. According to these findings, the reranker of the baselines (b) and (c) might have a negative impact on the retrieval performance of the top retrieved passages. Hence, this might also lead to reducing the performances of the reader of the (b) and (c) baselines. Moreover, we further analyse why our proposed *ORConvQA*$_{4:Reranker+Reader}$ method does not outperform the baseline (c) in terms of HEQ-Q on OR-CoQA. We find that the average number of tokens in the OR-CoQA's answer (2.6 tokens per answer) is remarkably short compared to that of the OR-QuAC's answer (14.7 tokens per answer) (Qu et al. 2021), and the predicted answer from the baseline (c) is shorter than that of our proposed *ORConvQA*$_{4:Reranker+Reader}$ method. This prediction may lead to our proposed *ORConvQA*$_{4:Reranker+Reader}$ method having a lower HEQ-Q score than the baseline (c). As described in Section 8.3.1, our proposed monoQA model is fine-tuned on the OR-QuAC dataset and evaluated on the OR-QuAC and OR-CoQA datasets. We postulate that this explains why evaluating the model with OR-CoQA exhibits a lower performance. Recall from Section 3.7.2 that OR-CoQA has no relevance assessments for retrieval, and hence we are unable to train a retrieval model for that dataset (which has shorter answers than OR-QuAC).

In answer to RQ 8.4, we conclude that our proposed *ORConvQA*$_{4:Reranker+Reader}$ MTL method, which includes monoQA for the joint learning of the reranker and the reader by sharing a single

Table 8.7: Comparison of average prediction times for monoQA and separate applications of monoT5 and UnifiedQA on the OR-QuAC test set.

Model	Average Prediction Time
monoQA (MTL of reranker and reader)	23ms
monoT5 (reranker) + UnifiedQA (reader)	44ms

text generation model, does help to improve the overall performance, yielding statistically significant improvements over the baselines on both the OR-QuAC and OR-CoQA datasets. It is also of note that such a joint learning can enhance the performances of the models on the ORConvQA task compared to using monoT5 and UnifiedQA separately. Later in Section 8.4.6, we also analyse the efficiency of our proposed $ORConvQA_{4:Reranker+Reader}$ method, using the monoQA model, compared with the individual application of monoT5 and UnifiedQA.

The performances of baselines (b) and (c) on OR-QuAC raise the question as to how do the baselines (b) and (c) compare to our monoQA model when using the ground truth passages provided in the OR-QuAC test set instead of using the retrieved passages. We provide such an analysis in Section 8.4.7, which shows that our monoQA reader achieves the best performance and significantly outperforms the reader of the baselines (b) and (c) on all measures.

8.4.5 RQ 8.5: Effectiveness of using a Reranker

Next, we examine the effectiveness of our proposed $ORConvQA_{4:Reranker+Reader}$ method (ConvDR+monoQA). This method employs a three-stage pipeline using ConvDR as the retriever and monoQA as both the reranker and reader. We compared our proposed $ORConvQA_{4:Reranker+Reader}$ method to the two-stage pipeline baselines (d)-(g), which each uses a bi-encoder for retrieval as input into a reader (see details in Section 8.3.2). This allows to establish the impact of the reranker. Table 8.6 (bottom-half) presents the results of the baselines (d)-(g). From the table, we observe that our proposed $ORConvQA_{4:Reranker+Reader}$ method, which is a three-stage pipeline using ConvDR as the retriever and monoQA as both the reranker and reader, achieves the highest performance by significantly outperforming all two-stage baselines on all measures – e.g. see *row (d)* vs. the last row in Table 8.6.

In answer to RQ 8.5, we conclude that integrating the reranker in the pipeline does help to improve both the retrieval and QA performances, yielding statistically significant improvements over the two-stage pipeline baselines.

8.4.6 Efficiency of monoQA

In this section, we measure the efficiency of inference of monoQA, which jointly learns the reranker and reader, in comparison with using monoT5 as the reranker and UnifiedQA as the reader separately on the test set of the OR-QuAC dataset. From Table 8.7, we find that the average

Table 8.8: Evaluation results on OR-QuAC in comparison to the baselines by extracting the answer on the *ground truth passage*. † denotes a performance significantly worse than our proposed monoQA model in terms of word-level F1 (McNemar's test, $p < 0.05$). The highest value for each measure is highlighted.

Reranker	Reader	F1	HEQ-Q	HEQ-D
(b) BERT (MTL of reranker and extractive reader)		40.0‡	33.0	3.5
(c) monoT5	UnifiedQA	30.0‡	16.5	1.9
(ours) monoQA (MTL of reranker and generative reader)		**56.5**	**48.7**	**7.1**

prediction time of monoQA is 23ms, whereas the average prediction time of using monoT5 as the reranker and UnifiedQA as the reader separately is 44ms. This is because monoQA uses a single model application for addressing both the reranker and reader stages. Indeed, we conclude that, on the test set of OR-QuAC, our monoQA model is approximately twice as fast in inference as the individual application of monoT5 and UnifiedQA for reranking and extracting the answer.

8.4.7 Effect of Providing Ground Truth Passages

In this section, we experiment to answer the question concerning how baselines (b) and (c) (listed in Section 8.3.2) compare to our monoQA model,when using the ground truth passages provided in the OR-QuAC test set instead of using the retrieved passages. In particular, recall that baseline (b) is the MTL of the reranker and reader by sharing a BERT encoder, while (c) is the individual application of monoT5 and UnifiedQA. By doing this comparison, we can control the impact of the reranker, and consider only the effectiveness of the reader. Indeed, in this setting, all models predict the answer by using the question and the ground truth passage. Table 8.8 shows the results of each evaluated model on the test set of OR-QuAC.

On analysing Table 8.8, we observe that our proposed monoQA model achieves the best performance across all measures and significantly outperforms the baselines (b) and (c) in terms of word-level F1 scores according to the McNemar's test ($p < 0.05$). Indeed our monoQA model's joint learning of the reader and the reranker can indeed help improve the performance of the answer extraction.

8.5 Multi-Task Learning of Conversational Question Rewriting, Passage Retrieval, and Answer Extraction

In this section, we investigate applying the Multi-Task Learning of the three tasks: Conversational Question Rewriting, Passage Retrieval, and Answer Extraction. By doing this, we introduce our proposed $ORConvQA_{5:MTL3Tasks}$ method, which leverages the Multi-Task Learning of the conversational question rewriting, passage retrieval, and answer extraction tasks.

Table 8.9: Evaluation results on OR-QuAC compared to the baselines. † denotes a performance significantly different compared to our proposed $ORConvQA_{5:MTL3Tasks}$ method in terms of MAP@10, Recall@5, and MRR@5 (paired t-test, $p < 0.05$); ‡ denotes a performance significantly different compared to our proposed $ORConvQA_{5:MTL3Tasks}$ method in terms of word-level F1 (McNemar's test, $p < 0.05$); The highest value for each measure is highlighted.

	Retriever	Reader	Retrieval			QA		
			MAP@10	Recall@5	MRR@5	F1	HEQ-Q	HEQ-D
	ConvDR (retriever only)	-	0.617	0.745	0.631	-	-	-
	Two-stage pipeline (retriever and reader)							
(d)	ConvDR	monoQA (reader)	**0.617**	**0.745**	**0.631**	**32.9**‡	**26.6**	**3.5**
(e)	ConvDR	UnifiedQA	**0.617**	**0.745**	**0.631**	19.6‡	9.5	1.0
(f)	TCT-ColBERT	UnifiedQA	0.370†	0.501†	0.386†	14.1‡	6.2	1.0
(g)	CQE (Lin et al. 2021*a*)	ORConvQA (reader)	-	0.415	0.310	32.0	-	-
($ORConvQA_{5:MTL3Tasks}$)	MTL of QR, Retriever, and Reader		0.616	0.744	0.627	26.4	12.2	1.5

8.5.1 Multi-Task Learning for Three Tasks

Recall that our proposed $ORConvQA_{4:Reranker+Reader}$ method employs two models consisting of ConvDR as the retriever and monoQA as the MTL of the reranker and reader (see Section 8.2.2). In this section, we investigate an approach for the ORConvQA task using a single model that integrates three components: conversational question rewriting, the retriever, and the reader. To simplify the model's complexity, we initially focus on these three components, omitting the reranker. The integration of all four components, including the reranker, is left as a direction for future work. To achieve this, we introduce the $ORConvQA_{5:MTL3Tasks}$ method. In particular, let $MTL(\cdot)$ denotes a joint learning function of the conversational question rewriting, passage retrieval, and answer extraction as follows:

$$ORConvQA_{5:MTL3Tasks} = MTL(QR, Retriever, Reader, \theta) \tag{8.14}$$

where θ are the learnable parameters of the model. The model is then fine-tuned to learn from teacher-generated embeddings on oracle reformulated questions (as mentioned in Section 3.7.4), retrieve the relevant passages, and extract the answers from retrieved passages.

8.5.2 RQ 8.6: Effectiveness of MTL for Three Tasks

In this section, we investigate the performances of our proposed $ORConvQA_{5:MTL3Tasks}$ method, which leverages the Multi-Task Learning of conversational question rewriting, reranker, and reader. We compared our proposed $ORConvQA_{5:MTL3Tasks}$ method to the two-stage pipeline baselines (d)-(g) (see details in Section 8.3.2) on the test sets of the OR-QuAC. To ensure a fair comparison with our method, we did not include the three-stage pipeline baselines (a)-(c) (including reranker in the pipeline), making them more compatible with our $ORConvQA_{5:MTL3Tasks}$ method's architecture. In Table 8.9, the first row shows the results of ConvDR as the retriever only and the last row shows the results of our proposed $ORConvQA_{5:MTL3Tasks}$ method.

From the table, when evaluating passage retrieval performance on the OR-QuAC dataset

test sets, we observe that our proposed $ORConvQA_{5:MTL3Tasks}$ method is comparable to the baselines (d)-(e) as well as when using ConvDR as the retriever only. Moreover, our proposed $ORConvQA_{5:MTL3Tasks}$ method outperforms the baselines (f)-(g) with a significantly different baseline (f) on all measures. For the question answering (QA) performance, we observe that our proposed $ORConvQA_{5:MTL3Tasks}$ method outperforms the baselines (e)-(f) on all measures. However, our proposed $ORConvQA_{5:MTL3Tasks}$ method had a lower QA performance compared to the baselines (d) and (g) with a significantly worse performance than the baseline (d) in terms of word-lever F1. According to these findings, the baseline (d), which employs our monoQA model as a reader, outperformed our $ORConvQA_{5:MTL3Tasks}$ method. This suggests that incorporating similar strategies or learning from the strengths of the monoQA model could further enhance the performance of our $ORConvQA_{5:MTL3Tasks}$ method. In contrast, our $ORConvQA_{5:MTL3Tasks}$ method, which employs MTL across three tasks within a bi-encoder network, might have a negative impact on the QA performance. We leave the investigation of how to improve the QA performance of our proposed $ORConvQA_{5:MTL3Tasks}$ method to future work.

In answer to RQ 8.6, we conclude that combining three tasks: Conversational Question Rewriting, Passage Retrieval, and Answer Extraction, does help to improve QA performances, yielding statistically significant improvements over several two-stage pipeline baselines.

8.6 Conclusions

In the thesis statement (stated in Section 1.3), we hypothesised that using Multi-Task Learning (MTL) for combining the reranker and reader roles within a single text generation model can enhance Open-Retrieval Conversational Question Answering performance. This approach aims to improve the system's effectiveness in retrieving relevant passages and extracting answers in ORConvQA. Therefore, to effectively address the ORConvQA task, we proposed the $ORConvQA_{4:Reranker+Reader}$ method (as introduced in Section 4.4.4), employing our monoQA model as the MTL of reranker and reader together with ConvDR as a retriever. Our proposed $ORConvQA_{4:Reranker+Reader}$ method directly addressed the limitation highlighted in Gap 4 in Section 3.10, which stated that there is no effective integration between Passage Reranking and Answer Extraction sharing a single text generation model. In particular, our investigation in this chapter aimed to answer six research questions.

Section8.4.1 showed that the monoQA model that exhibits the highest word-level F1 on the validation set is also the best model when evaluated on the test set (see Figure 8.4). Therefore, to fine-tune monoQA, we selected the best epochs using the best word-level F1 on the validation set. In Section 8.4.2, we showed that our designed prompt ($QuestionAnswering : \{q\}[sep]\{p\}$) has the highest performance. Therefore, we used this prompt to fine-tune our monoQA model. Then, we examined the effectiveness of the use of the models, such as monoT5, UnifiedQA, and t5-base, for initialising monoQA. Table 8.5 showed that the monoQA model initialised from monoT5 has the

best overall effectiveness. Therefore, we use monoT5 to initialise monoQA for answering RQ 8.4. Next, we showed the effectiveness of our proposed $ORConvQA_{4:Reranker+Reader}$ method, using the ConvDR model as a retriever and our monoQA model as reranker and reader, compared to several three-stage baselines (see Table 8.6 (top-half)). Moreover, in Table 8.6 (bottom-half), we also showed that integrating the reranker in the pipeline does help to improve both the retrieval and QA performances. These experimental results showed that our $ORConvQA_{4:Reranker+Reader}$ method (ConvDR+monoQA) achieved the highest performance compared to all two-stage baselines. In particular, we showed that our monoQA model is approximately twice as fast in inference as the individual application of monoT5 and UnifiedQA in Section 8.4.6. In particular, in Section 8.4.7, we further examined how the baselines (b) and (c) compare to our monoQA model when using the ground truth passages provided in the OR-QuAC test set instead of using the retrieved passages. Table 8.8 showed that our monoQA model's joint learning of the reader and the reranker can indeed help improve the performance of the answer extraction. In addition, we introduced our proposed $ORConvQA_{5:MTL3Tasks}$ method, which leverages the Multi-Task Learning of the conversational question rewriting, passage retrieval, and answer extraction tasks. Table 8.9 showed that our proposed $ORConvQA_{5:MTL3Tasks}$ method is comparable in passage retrieval performance to the baselines that use ConvDR as a retriever. However, our proposed $ORConvQA_{5:MTL3Tasks}$ method has lower question answering (QA) performance compared to the baseline that uses our monoQA model as the reader.

Hence, we can now validate our hypothesis presented in Section 1.3. Our experiments on two datasets, namely the OR-QuAC and OR-CoQA datasets, showed that our proposed $ORConvQA_{4:Reranker+Reader}$ method (ConvDR+monoQA) has the best effectiveness on these datasets, yielding statistically significant improvements over several strong baselines from the literature. Gap 4 highlighted the lack of effective integration between the passage retriever and answer extraction tasks. Our findings in this chapter showed that our $ORConvQA_{4:Reranker+Reader}$ method using our monoQA model learned through Multi-Task Learning by sharing a single text generation model, effectively addresses Gap 4.

Thus far, we have shown the effectiveness of our proposed MTL $ORConvQA_{1-5}$ methods, which instantiated our proposed ORConvQA framework in Chapter 4. In particular, starting from Chapter 5 to Chapter 8, we have proposed several methods to address the challenges (i.e. the ambiguities in conversational questions, retrieving and identifying the most relevant passages, and extracting the relevant answer) within such framework. Therefore, in the next chapter, we will summarise the main contributions and conclusions of this thesis. In addition, we will discuss possible future directions.

Chapter 9

Conclusions and Future Works

9.1 Contributions and Conclusions

This thesis addressed the challenges of leveraging Multi-Task Learning (MTL) to develop an effective Open-Retrieval Conversational Question Answering (ORConvQA) system. In particular, we postulated that by leveraging MTL to jointly learn several related tasks simultaneously in a uniform model, we can more effectively tackle the ORConvQA task. In particular, in the dynamic MTL approaches, the tasks' weights are automatically adjusted during learning, ensuring that each of the tasks' weight is adjusted by the relative importance of the different tasks. Therefore, in Chapter 4, we proposed an ORConvQA framework, which consists of seven components, namely Follow-up Question Identification, Conversational Question Rewriting, Clarification Need Classification, Asking Clarifying Questions, Pass Retrieval, Passage Reranking, and Answer Extraction (Conversational Question Answering). In particular, as outlined in Section 1.2, we argued that such a framework needs to address four main challenges, namely:

(i) **Ambiguities in Conversational Questions:** Questions in conversations can be ambiguous or unclear, requiring the system to use prior conversation context or to ask for more details to provide accurate answers.

(ii) **Retrieve Relevant Passages:** In a large-scale collection of passages, the efficient and accurate retrieval of relevant passages is important for a successful task.

(iii) **Identify the Most Relevant Passages:** Among the passages retrieved, determining which passages are the most relevant in the context of the ongoing conversation is a challenging task.

(iv) **Extract the Relevant Answers:** Once a list of relevant candidate passages is formed, extracting the most accurate and relevant answers represents the final challenge.

In particular, this thesis addressed these four challenges. Below, we discuss our main contributions and conclusions in this the thesis:

- **Conclusion 1: Effective Conversational Question Answering using a Dynamic Multi-Task Learning Method:** To address the challenge of extracting the relevant answers, we proposed an $ORConvQA_{1:dynamicMTL}$ method (see Section 4.4.1 and Chapter 5) for Conversational Question Answering (ConvQA), which learns to extract the correct answer by applying Multi-Task Learning (MTL). Our proposed $ORConvQA_{1:dynamicMTL}$ method directly addressed Gap 1 (as identified in Section 3.10), which stated that the current ConvQA approaches using MTL lack the dynamic adjustment of the tasks importance during learning. In particular, our proposed $ORConvQA_{1:dynamicMTL}$ MTL method makes use of Evolving Weighting by Abridged Linear for learning the main task, namely Answer Extraction, while the auxiliary tasks are addressed using Loss-Balanced Task Weighting (see Section 5.3.3). Next, we evaluated the effectiveness of our $ORConvQA_{1:dynamicMTL}$ MTL method on the QuAC ConvQA dataset. The experimental results in Table 5.4 showed that our proposed $ORConvQA_{1:dynamicMTL}$ MTL method significantly outperformed all baselines (see Section 5.4.3) in terms of the word-level F1 (see Section 3.2.3) metric (McNemar's test, $p < 0.05$). These results demonstrated the effectiveness of our proposed $ORConvQA_{1:dynamicMTL}$ MTL method in extracting the relevant answer within conversational contexts.

- **Conclusion 2: Effective Open-Retrieval Conversational Question Answering using Multi-Task Learning of Conversational Question Rewriting and Follow-up Question Identification:** To address the challenge of ambiguities in conversational questions, we introduced a method for Open-Retrieval Conversational Question Answering (ORConvQA), called $ORConvQA_{2:FID+QR}$ (see Section 4.4.2 and Chapter 6), which learns to predict the follow-up question and rewrites the conversational question simultaneously. Our proposed $ORConvQA_{2:FID+QR}$ MTL method directly addressed the issue highlighted in Gap 2 in Section 3.10, which stated that there is no existing effective integration of the Follow-up Question Identification and Conversational Question Rewriting tasks. In particular, our proposed $ORConvQA_{2:FID+QR}$ method makes use of text generation models including BART and T5 by generating the first token for a classification task and the follow-up tokens for a questing rewriting task (see Section 6.2.2). Next, to evaluate the effectiveness of our proposed $ORConvQA_{2:FID+QR}$ method, we compared it with the existing state-of-the-art three-way attentive pooling network, BERT, as well as the Single Task Learning (STL) of T5 and BART (see Section 6.3.3). The experimental results in Table 6.3 showed that our proposed $ORConvQA_{2:FID+QR}$ MTL method, using BART, had the best overall effectiveness, yielding statistically significant improvements in terms of F1 and Macro-F1 over the baselines, on each of the three test sets of the LIF dataset. In addition, Table 6.4 showed that our proposed $ORConvQA_{2:FID+QR}$ MTL method, using T5, improves performance in conversational question rewriting and passage retrieval, yielding statistically significant improvements over the MTL discriminative+generative BART model as well

as all the STL models. These results demonstrated the effectiveness of our proposed $ORConvQA_{2:FID+QR}$ MTL method for follow-up question identification and conversational question rewriting.

- **Conclusion 3: Effective Open-Retrieval Conversational Question Answering using Multi-Task Learning of Clarification Need Classification and Clarifying Question Generation:** To further address the challenge of ambiguities in conversational questions, we proposed an $ORConvQA_{3:CNC+Askng}$ method, including a hybrid method for generating and selecting clarifying questions, in mixed-initiative conversational search, combining the strengths of clarifying question generation and selection. Our proposed $ORConvQA_{3:CNC+Askng}$ MTL method directly addressed the issue identified in Gap 3 of Section 3.10, which stated that prior works have not focused on a comprehensive approach for asking clarifying questions. In particular, our proposed $ORConvQA_{3:CNC+Askng}$ method leverages Multi-Task Learning to simultaneously determine clarification needs and to generate clarifying questions in a model called T5MI, then uses GTR to select questions from the pool. The candidate questions from both generation and selection approaches are then scored using a text generation model for point-wise question classification called T5Ranking (see Section 7.2.2). Next, to evaluate the effectiveness of our proposed $ORConvQA_{3:CNC+Askng}$ method, we compared it with the 11 baselines listed as (a)-(k) in Table 7.4. These baselines include generative models like GPT-3 and T5, as well as models such as BM25 and miniLM-BERT. The experimental results in Table 7.6 (left side) showed that our hybrid method for generating and selecting clarifying questions had the best overall effectiveness, yielding statistically significant improvements in terms of Relevance@1 and Diversity@1 over the baselines (a)-(k) on the CAsT 2022 dataset. In addition, Table 7.6 (right side) showed that our proposed $ORConvQA_{3:CNC+Askng}$ method, including our hybrid method for generating and selecting clarifying questions, does help to improve the overall passage retrieval performance of a mixed-initiative conversational search system, yielding statistically significant improvements over the baselines (a) - (b) and (g) on TREC CAsT 2022. Furthermore, the results in Figure 7.5 confirmed that our hybrid method for generating and selecting clarifying questions improves the user's predicted satisfaction with the clarifying questions – compared to other baseline methods – based on the users' sentiment expressed in response to the asked clarifying questions.

- **Conclusion 4: Effective Open-Retrieval Conversational Question Answering using Multi-Task Learning of Reranking and Answer Extraction:** To address the challenge of retrieving and identifying the most relevant passages and extracting the relevant answer, we proposed an $ORConvQA_{4:Reranker+Reader}$ MTL method (see Section 4.4.4 and Chapter 8), employing a new proposed model called monoQA as the Multi-Task Learning (MTL) of the reranker and reader together with ConvDR as a retriever (see Section 8.2.2). Our

proposed *ORConvQA*$_{4:Reranker+Reader}$ method directly addressed the limitation highlighted in Gap 4 of Section 3.10, which stated that there is no effective existing integration between Passage Reranking and Answer Extraction sharing a single text generation model. Next, to evaluate the effectiveness of our proposed *ORConvQA*$_{4:Reranker+Reader}$ method, we compared it with seven baseline systems, including a three-stage pipeline and a two-stage pipeline, listed as (a)-(g) in Section 8.3.2. Table 8.6 showed the effectiveness of our proposed *ORConvQA*$_{4:Reranker+Reader}$ method on both the retrieval and conversational question answering performances, through its use of the ConvDR model as a retriever and our monoQA model as the reranker and reader, compared to several three-stage and two-stage baselines, yielding statistically significant improvements over the baselines on both the OR-QuAC and OR-CoQA datasets.

- **Conclusion 5: Effective Open-Retrieval Conversational Question Answering using Multi-Task Learning of Conversational Question Rewriting, Passage Retrieval, and Answer Extraction:** Our investigation into Multi-Task Learning (MTL) for Conversational Question Rewriting, Passage Retrieval, and Answer Extraction led to the development of the *ORConvQA*$_{5:MTL3Tasks}$ method. This approach simultaneously learns these three tasks using MTL. In evaluating our proposed *ORConvQA*$_{5:MTL3Tasks}$ method against two-stage pipeline baselines (d)-(g) on OR-QuAC test sets, we found that it performed comparably to baselines (d)-(e). Moreover, our proposed *ORConvQA*$_{5:MTL3Tasks}$ method had better performance over the baselines (f)-(g). In particular, our proposed *ORConvQA*$_{5:MTL3Tasks}$ method significantly outperformed the baseline (f) on all measures. For the question answering (QA) performance, we observe that our proposed *ORConvQA*$_{5:MTL3Tasks}$ method outperforms the baselines (e)-(f) on all measures.

Next, based on the experimental results from Chapters 5 to 8, we validate our thesis statement that we introduced in Section 1.3. We stated that the effectiveness of Open-Retrieval Conversational Question Answering can be improved by leveraging Multi-Task Learning that jointly learns several related tasks simultaneously in a uniform model. In the following, we discuss the experimental results and observations that validate our proposed thesis statement.

- **Claim 1:** *By leveraging the dynamic Multi-Task Learning (MTL) approach, we simultaneously train the main task of answer extraction, along with auxiliary tasks such as follow-up question identification, yes/no question prediction, and unanswerable prediction and by incorporating these tasks into a unified model, we sarafcan enhance the system's effectiveness for Conversational Question Answering (ConvQA).* The experiments of the three studies in Chapter 5 validated this claim by showing that our proposed *ORConvQA*$_{1:dynamicMTL}$ dynamic MTL method significantly outperforms 9 existing baselines (see Table 5.4). In particular, there is little difference between the efficiency of our proposed *ORConvQA*$_{1:dynamicMTL}$ MTL method, and that of the static task weighting

methods, or Single-Task Learning in both the training and evaluation phases even though our approach has a more complex implementation. Then, we showed the effectiveness of different combinations of auxiliary tasks compared with Single Task Learning (see Table 5.6). We found that models combining multiple auxiliary tasks outperformed the model learning only the main Answer Extraction task. Finally, we examined the performance of the auxiliary tasks after training in a Multi-Task setting (see Table 5.7). We concluded that our method cannot enhance the performance of the auxiliary tasks.

- **Claim 2:** *By leveraging the shared learned structure of the follow-up question identification and conversational question rewriting tasks in a text generation model, we aim to enhance the system's effectiveness in follow-up question identification, conversational question rewriting and passage retrieval.* Our experiments in Chapter 6 validated this claim by showing that, our proposed $ORConvQA_{2:FID+QR}$ MTL method, using BART, significantly outperformed in terms of F1 and Macro-F1 the strongest existing baseline on the follow-up question identification task (see Table 6.3). Finally, we examined the effectiveness of the Conversational Question Rewriting models including our proposed $ORConvQA_{2:FID+QR}$ MTL method, using the T5 and BART models, and the existing baselines (see Table 6.4). We showed the effectiveness of our proposed $ORConvQA_{2:FID+QR}$ MTL method, using T5, in both the conversational question rewriting and passage retrieval tasks by significantly outperforming all existing baselines.

- **Claim 3:** *By leveraging Multi-Task Learning, we can generate more effective clarifying questions through the simultaneous learning of clarification need classification and clarifying question generation.* We have validated this claim in Chapter 7, where we proposed the $ORConvQA_{3:CNC+Askng}$ MTL method, including the hybrid method for generating and selecting clarifying questions. As described in Section 7.2.2, in generating clarifying questions, our method leverages the Multi-Task Learning (MTL) of clarification need classification and the generation of clarifying questions, namely T5MI. For selecting the clarifying question, we used a state-of-the-art dense retrieval model, namely GTR. To rank the candidate clarifying questions obtained from both generating and selecting clarifying questions, we scored the question using the text generation model for point-wise question clarification, namely T5Ranking. First, we examine the effectiveness of the mixed-initiative system for asking clarifying questions on the clarification need classification (see Table 7.5). The experimental results showed our MTL T5MI model is overall effective on the clarification need classification task. Finally, we investigate the performance of our hybrid method (T5MI+GTR+T5Ranking) for generating and selecting clarifying questions in improving the asking clarifying questions performance (see Table 7.6). In particular, we showed the effectiveness of our proposed $ORConvQA_{3:CNC+Askng}$ MTL method through significant improvements over the baselines on passage retrieval.

- **Claim 4:** *By leveraging Multi-Task Learning (MTL), we can enhance the performance of the Open-Retrieval Conversational Question Answering (ORConvQA) task by sharing the learned structure of the reranker and reader in a single text generation model.* This claim has been validated in Chapter 8, where we proposed the $ORConvQA_{4:Reranker+Reader}$ MTL method, employing our proposed monoQA model for the Multi-Task Learning (MTL) of reranker and reader together with ConvDR as a retriever. In particular, we investigate the performances of our proposed Multi-Task Learning (MTL) method, $ORConvQA_{4:Reranker+Reader}$, which uses ConvDR as the retriever and monoQA as both the reranker and reader, in comparison to the baselines (see Table 8.6). The results showed the effectiveness of our proposed $ORConvQA_{4:Reranker+Reader}$ MTL method through significant improvements over the baselines. Therefore, we concluded that our proposed $ORConvQA_{4:Reranker+Reader}$ MTL method can indeed yield strong ORConvQA performances, which validates our proposed claim.

- **Claim 5:** *By leveraging the dynamic Multi-Task Learning (MTL) approach, we jointly learn conversational question rewriting, passage retrieval, and answer extraction in a unified model and by integrating these tasks, we can enhance the overall effectiveness of Open-Retrieval Conversational Question Answering.* This claim has been validated in Chapter 8, where we proposed the $ORConvQA_{5:MTL3Tasks}$ MTL method. We investigated the performances of our proposed $ORConvQA_{5:MTL3Tasks}$ method, which leverages the Multi-Task Learning of conversational question rewriting, reranker, and reader (see Table 8.9). The experimental results showed the effectiveness of our proposed $ORConvQA_{5:MTL3Tasks}$ MTL method in improving the question answering performance.

In summary, we argue that we have validated each of the claims of our thesis statement in Section 1.3 using publicly available datasets. Indeed, we showed that we can improve the effectiveness of Open-Retrieval Conversation Question Answering (ORConvQA) using Multi-Task Learning (MTL) to learn several related tasks simultaneously in a uniform model. In particular, in the dynamic MTL approaches, the tasks' weights are automatically adjusted during learning, ensuring that each of the tasks' weights is adjusted by the relative importance of the different tasks. Furthermore, we showed that our five ORConvQA MTL methods within our proposed ORConvQA framework, can enhance the effectiveness of the ORConvQA performance. Our proposed framework aimed to bridge the five gaps in the current approaches for ORConvQA and the advancements needed to address the ORConvQA task effectively. Next, in Section 9.4, we describe some future research directions for ORConvQA.

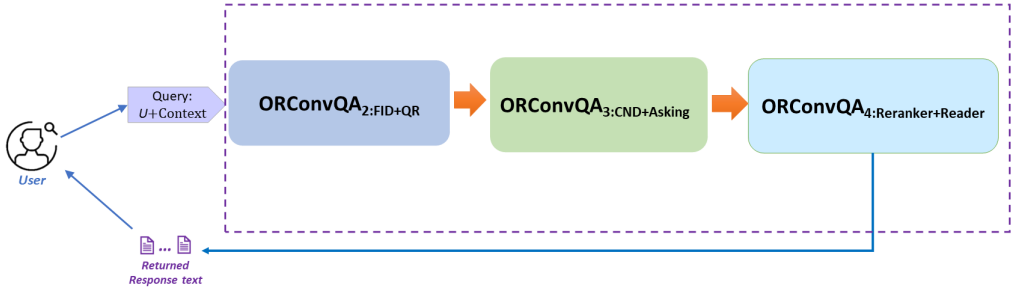


Figure 9.1: A hypothetical end-to-end system for ORConvQA, integrating three proposed MTL methods: $ORConvQA_{2:FID+QR}$, $ORConvQA_{3:CNC+Asking}$, $ORConvQA_{4:Reranker+Reader}$.

9.2 Integration of ORConvQA Methods into a Unified System

In this thesis, we have prooposed $ORConvQA_{1-5}$ methods, each addressing specific sub-tasks within the ORConvQA task using Multi-Task Learning. In this section, we suggest integrating these methods into hypothetical systems, forming a comprehensive end-to-end framework.

Figure 9.1 presents a comprehensive end-to-end ORConvQA system that combines the three proposed methods: $ORConvQA_{2:FID+QR}$, $ORConvQA_{3:CNC+Asking}$, $ORConvQA_{4:Reranker+Reader}$. We start by receiving the user's query and conversation history ($U+Context$). $ORConvQA_{2:FID+QR}$ first identifies the need for follow-up questions and conversational question rewriting to address possible ambiguities in the user's input. Then, $ORConvQA_{3:CNC+Asking}$ performs clarification need classification to determine whether asking clarifying questions is necessary. If needed, contextually relevant clarifying questions are generated. Based on the clarified query, $ORConvQA_{4:Reranker+Reader}$ retrieves relevant passages, reranks the retrieved passages to prioritize the most relevant ones, and then extracts the final answer from the top-ranked passage to provide it as a response to the user. Therefore, the system can be written as follows:

$$\begin{aligned} \text{ORConvQA} = {} & ORConvQA_{2:FID+QR} \\ & \overset{q'_k}{\gg} ORConvQA_{3:CNC+Asking} \\ & \overset{q''_k}{\gg} ORConvQA_{4:Reranker+Reader} \end{aligned} \tag{9.1}$$

This section culminates our research by presenting a comprehensive end-to-end ORConvQA system. The system integrates MTL methods that have been proposed, trained, and evaluated throughout this thesis. It serves as a practical demonstration, showcasing how these methods can be combined to deliver precise and context-aware responses to users within a conversational search setting.

9.3 Limitations

While our research has shown promising results in leveraging Multi-Task Learning (MTL) for the Open-Retrieval Conversational Question Answering (ORConvQA) task, several limitations need to be acknowledged.

- **Focus on Multi-Turn Conversational Questions:** Our approach focuses on multi-turn conversational question-answering, which, while effective for detailed dialogues, limits its utility in simpler, single-turn interactions common in many real-world applications. In future work, we aim to extend our methodology to encompass single-turn dialogues, broadening the model's applicability to various conversational scenarios.

- **Focus on Extractive Answers:** Although our generative model is trained to produce only word sequences that appear in the input passage, we observe that 1.5% of the generated tokens are not extracted from the input. While this may not affect user satisfaction, the extractive evaluation measures may underestimate the model's utility. For this reason, it is also worth investigating the multi-task learning of the reranker and an extractive reader by sharing a single model.

- **Reliance on Rewritten Questions:** Our model, monoQA, depends on rewritten questions provided by another T5 model. Ideally, we would like a single model to be able to use the original question (without needing it to be first rewritten). We leave both these investigations to future work.

- **Question Pool Dependence:** Our method's performance is also tied to the quality and diversity of the question pool for the selection-based approaches. Thus, improving the robustness and variety of the question pool could enhance the system's performance.

- **Limited Clarification Rounds:** Our current method only considers a single round of clarification. In real-world scenarios, multiple rounds of clarifying questions may be necessary to fully understand a user's intent. Therefore, exploring models that can handle multiple rounds of clarifications could be a worthwhile direction for future work.

- **Specificity of Our Proposed Hybrid Task Weighting:** Our proposed Hybrid Task Weighting method provides advantages for encoder-only models like BERT, effectively combining and optimising multiple loss functions to enhance performance across related tasks. However, it's important to note that this method is not suitable for autoregressive text generation models like T5. These models rely on sequential token generation rather than simultaneous loss optimization across tasks. The training dynamics of text generation, where each token's generation depends on previously generated tokens, require a different approach than our Hybrid Task Weighting method provides. Therefore, although our method is

effective with encoder-only models, it is not suitable for models that rely on sequence generation techniques.

9.4 Directions for Future Work

In this section, we discuss possible future directions that could further benefit Multi-Task Learning in the Open-Retrieval Conversation Question Answering (ORConvQA) task.

- **Dependency on Pre-Rewritten Questions in MTL Frameworks:** In Chapters 6, 7, and 8, we explored leveraging Multi-Task Learning (MTL) to jointly learn several related tasks using a single T5 text generation model. However, our proposed models currently rely on input from a rewritten question generated by another T5 model. Ideally, we aim for a single model that can directly process the original user question, along with the conversation history, without the need for prior rewriting. Therefore, an interesting future direction would be to adapt our MTL T5 model to handle the original user questions and conversation history directly to further enhance the performance of Open-Retrieval Conversational Question Answering.

- **Scalability and Diversity Challenges in Mixed-initiative Conversational Search:** In Chapter 7, while our proposed hybrid method has shown promising results in generating and selecting clarifying questions in mixed-initiative conversational search, several current limitations provide avenues for future research. The current approach relies heavily on Multi-Task Learning and Generative Text Rewriting, which could potentially face scalability issues in larger or varied datasets. Our model's performance is also tied to the quality and diversity of the question pool for the selection-based approaches. Hence, improving the robustness and variety of the question pool could enhance the system's performance. Moreover, our current model only considers a single round of clarification. In real-world scenarios, multiple rounds of clarifying questions may be necessary to fully understand a user's intent. Therefore, exploring models that can handle multiple rounds of clarifications could be a worthwhile direction for future work. Furthermore, whilst our model outperformed strong baselines on the relevance and novelty criteria, there is room for improvement in relation to the diversity of the generated questions. Future work could involve exploring methods to increase the diversity of the generated questions without sacrificing their relevance and novelty.

- **Optimising Task Combinations in Dynamic Multi-Task Learning for Conversational QA:** In this thesis, we have shown how to leverage Multi-Task Learning (MTL) to jointly learn several related tasks when sharing a uniform model. However, we observed that the model that combines three tasks did not outperform the models that combine only two tasks (see Sections 5.5.2 and 8.5.1). We conjecture that negative transfer (see Section 5.3.2)

might be a possible reason explaining the drop in the performance of MTL. However, to alleviate this problem, many different dynamic MTL approaches could be useful in conversational question answering such as Dynamic Weight Averaging (Guo et al. 2018) and Dynamic Task Prioritisation (Xie et al. 2022). Therefore, it would be interesting to explore these dynamic MTL approaches further. Investigating how different task combinations and weight adjustment strategies affect overall performance could provide valuable new insights.

- **Integration of Extractive and Generative Approaches in QA Systems:** In the scope of this thesis, we only studied an extractive type of question answering (QA), where the goal is to generate/extract tokens that are a subset of a passage. Our studies did not consider the generative type of QA, where the goal is to generate factual or creative text. However, generative QA can provide more flexible and comprehensive answers, especially when direct answers are not available in the passage (Luo et al. 2022). However, note also that the generative QA model may hallucinate (Gospodinov et al. 2023, Maynez et al. 2020), produce non-relevant or nonsensical questions. Recently, the combination of the answers of extractive and generative approaches is becoming popular (Fajcik et al. 2021, Wen et al. 2022). By integrating these approaches, we can leverage the precision of extractive QA while harnessing the creativity and comprehensiveness of generative QA. Therefore, an interesting direction is to consider investigating the combination of extractive and generative QA using Multi-Task Learning to share a single text generation model.

9.5 Concluding Remarks

This thesis has addressed a challenging task: the Open-Retrieval Conversation Question Answering (ORConvQA) task. In particular, this thesis contributed to leveraging Multi-Task Learning by proposing effective methods within our ORConvQA framework. However, in Section 9.4, we identified a number of interesting challenges and exciting topics that still remain open in this research field. This work provides a solid motivation and the groundwork for exploring these further research directions in the future. We believe that using Multi-Task Learning to jointly learn several related tasks in a uniform model will continue to benefit the future development of the Open-Retrieval Conversation Question Answering domain.

Bibliography

Aliannejadi, M., Kiseleva, J., Chuklin, A., Dalton, J. & Burtsev, M. (2020*a*), Convai3: Generating clarifying questions for open-domain dialogue systems (clariq), *in* 'arXiv preprint arXiv:2009.11352'.
URL: *https://doi.org/10.48550/arXiv.2009.11352*

Aliannejadi, M., Kiseleva, J., Chuklin, A., Dalton, J. & Burtsev, M. (2020*b*), 'Convai3: Generating clarifying questions for open-domain dialogue systems (clariq)', *arXiv preprint arXiv:2009.11352* .
URL: *https://doi.org/10.48550/arXiv.2009.11352*

Aliannejadi, M., Kiseleva, J., Chuklin, A., Dalton, J. & Burtsev, M. (2021), Building and evaluating open-domain dialogue corpora with clarifying questions, *in* 'Proc. EMNLP', pp. 4473–4484.
URL: *https://aclanthology.org/2021.emnlp-main.367*

Aliannejadi, M., Zamani, H., Crestani, F. & Croft, W. B. (2019), Asking clarifying questions in open-domain information-seeking conversations, *in* 'Proc. SIGIR', New York, NY, USA, p. 475–484.
URL: *https://doi.org/10.1145/3331184.3331265*

Amati, G. (2003), 'Probability models for information retrieval based on divergence from randomness', *Glasgow University* .

Amati, G., Amodeo, G., Bianchi, M., Gaibisso, C. & Gambosi, G. (2008), Fub, iasi-cnr and university of tor vergata at trec 2008 blog track, Technical report, Foundazione Ugo Bordoni Rome (Italy).

Amati, G. & Van Rijsbergen, C. J. (2002), 'Probabilistic models of information retrieval based on measuring the divergence from randomness', *ACM Trans. Inf. Syst.* .

Anantha, R., Vakulenko, S., Tu, Z., Longpre, S., Pulman, S. & Chappidi, S. (2021), Open-domain question answering goes conversational via question rewriting, *in* 'Proc. NAACL', pp. 520–534.
URL: *https://aclanthology.org/2021.naacl-main.44*

Baxter, J. (1997), 'A bayesian/information theoretic model of learning to learn via multiple task sampling', *Machine learning* **28**, 7–39.

Belharbi, S., Hérault, R., Chatelain, C. & Adam, S. (2016), Deep multi-task learning with evolving weights, *in* 'Proc. ESANN'.
URL: *https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2016-87.pdf*

Bertomeu, N., Uszkoreit, H., Frank, A., Krieger, H.-U. & Jörg, B. (2006), Contextual phenomena and thematic relations in database qa dialogues: results from a wizard-of-oz experiment, *in* 'Proc. HLT-NAACL', pp. 1–8.
URL: *https://www.aclweb.org/anthology/W06-3001*

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I. & Amodei, D. (2020), Language models are few-shot learners, *in* 'Proc. NIPS'.
URL: *https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf*

Caruana, R. (1993), Multitask Learning: A Knowledge-Based Source of Inductive Bias, *in* 'Proc. ICML', ICML'93, p. 41–48.

Casola, S., Lauriola, I. & Lavelli, A. (2022), 'Pre-trained transformers: an empirical comparison', *Machine Learning with Applications* p. 100334.
URL: *https://www.sciencedirect.com/science/article/pii/S2666827022000445*

Chen, D., Fisch, A., Weston, J. & Bordes, A. (2017), Reading Wikipedia to answer open-domain questions, *in* 'Proc. ACL', pp. 1870–1879.
URL: *https://aclanthology.org/P17-1171*

Chen, Z., Badrinarayanan, V., Lee, C.-Y. & Rabinovich, A. (2018), GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks, *in* 'Proc. ICML', pp. 794–803.
URL: *https://arxiv.org/abs/1711.02257*

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. (2014), Learning phrase representations using RNN encoder–decoder for statistical machine translation, *in* 'Proc. EMNLP', pp. 1724–1734.
URL: *https://aclanthology.org/D14-1179*

Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.-t., Choi, Y., Liang, P. & Zettlemoyer, L. (2018), QuAC: Question answering in context, *in* 'Proc. EMNLP.', p. 2174–2184.
URL: *https://aclanthology.org/D18-1241*

Clark, K., Luong, M.-T., Le, Q. V. & Manning, C. D. (2020), ELECTRA: Pre-training text encoders as discriminators rather than generators, *in* 'ICLR'.
URL: *https://openreview.net/pdf?id=r1xMH1BtvB*

Clark, P., Etzioni, O., Khashabi, D., Khot, T., Mishra, B. D., Richardson, K., Sabharwal, A., Schoenick, C., Tafjord, O., Tandon, N., Bhakthavatsalam, S., Groeneveld, D., Guerquin, M. & Schmitz, M. (2020), 'From f to a on the new york regents science exams — an overview of the aristo project', *AI Mag.* p. 39–53.
URL: *https://doi.org/10.1609/aimag.v41i4.5304*

Clarke, C. L., Craswell, N., Soboroff, I. & Voorhees, E. M. (2011), Overview of the trec 2011 web track., *in* 'TREC'.

Coden, A., Gruhl, D., Lewis, N. & Mendes, P. N. (2015), Did you mean A or B? supporting Clarification Dialog for Entity Disambiguation., *in* 'Proc. Sumpre-HSWI@ ESWC'.

Collobert, R. & Weston, J. (2008), A unified architecture for natural language processing: Deep neural networks with multitask learning, *in* 'Proceedings of the 25th international conference on Machine learning', pp. 160–167.

Crawshaw, M. (2020), 'Multi-task learning with deep neural networks: A survey', *arXiv preprint arXiv:2009.09796* .

Croft, W. B., Metzler, D. & Strohman, T. (2010), *Search engines: Information retrieval in practice*, Vol. 520, Addison-Wesley Reading.

Dai, Z. & Callan, J. (2020), Context-aware term weighting for first stage passage retrieval, *in* 'Proc. SIGIR', p. 1533–1536.
URL: *https://doi.org/10.1145/3397271.3401204*

Dalton, J., Xiong, C. & Callan, J. (2019), TREC CAsT 2019: The conversational assistance track overview, *in* 'Proc. TREC'.
URL: *https://trec.nist.gov/pubs/trec28/papers/OVERVIEW.CAsT.pdf*

Dalton, J., Xiong, C. & Callan, J. (2020), CAsT 2020: The conversational assistance track overview, *in* 'Proc. TREC'.
URL: *https://trec.nist.gov/pubs/trec29/papers/OVERVIEW.C.pdf*

Dalton, J., Xiong, C. & Callan, J. (2021), CAsT 2021: The conversational assistance track overview, *in* 'Proc. TREC'.
URL: *https://trec.nist.gov/pubs/trec30/papers/Overview-CAsT.pdf*

Darji, H., Mitrović, J. & Granitzer, M. (2023), 'German bert model for legal named entity recognition', *arXiv preprint arXiv:2303.05388* .
URL: *https://doi.org/10.5220/0011749400003393*

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2019), BERT: Pre-training of deep bidirectional transformers for language understanding, *in* 'Proc. NAACL', pp. 4171–4186.
URL: *http://dx.doi.org/10.18653/v1/N19-1423*

Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M. & Hon, H.-W. (2019), Unified language model pre-training for natural language understanding and generation, *in* 'Proc. NIPS'.

dos Santos, C., Ma, X., Nallapati, R., Huang, Z. & Xiang, B. (2020), Beyond [CLS] through ranking by generation, *in* 'Proc. EMNLP', pp. 1722–1727.
URL: *https://www.aclweb.org/anthology/2020.emnlp-main.134*

Duong, L., Cohn, T., Bird, S. & Cook, P. (2015), Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser, *in* 'Proc. ACL', Association for Computational Linguistics, pp. 845–850.
URL: *https://www.aclweb.org/anthology/P15-2139*

Elgohary, A., Peskov, D. & Boyd-Graber, J. (2019), Can you unpack that? Learning to rewrite questions-in-context, *in* 'Proc. EMNLP', pp. 5918–5924.
URL: *https://aclanthology.org/D19-1605*

Fajcik, M., Docekal, M., Ondrej, K. & Smrz, P. (2021), R2-D2: A modular baseline for open-domain question answering, *in* 'Proc. EMNLP', pp. 854–870.
URL: *https://aclanthology.org/2021.findings-emnlp.73*

Fraser, D., Kane, A. & Tompa, F. W. (2018), Choosing math features for bm25 ranking with tangent-l, *in* 'Proc. DocEng'.
URL: *https://doi.org/10.1145/3209280.3209527*

Gospodinov, M., MacAvaney, S. & Macdonald, C. (2023), 'Doc2query: When less is more', *arXiv preprint arXiv:2301.03266* .
URL: *https://doi.org/10.48550/arXiv.2301.03266*

Guo, M., Haque, A., Huang, D.-A., Yeung, S. & Fei-Fei, L. (2018), Dynamic task prioritization for multitask learning, *in* 'Proc. ECCV', pp. 270–287.
URL: *https://link.springer.com/chapter/10.1007/978-3-030-01270-0*$_1$7

Hochreiter, S. & Schmidhuber, J. (1997), 'Long short-term memory', *Neural Comput.* p. 1735–1780.
URL: *https://doi.org/10.1162/neco.1997.9.8.1735*

Huang, H.-Y., Choi, E. & Yih, W.-t. (2019), 'FlowQA: Grasping flow in history for conversational machine comprehension', *Proc. ICLR* .

Ide, T. & Kawahara, D. (2021), Multi-task learning of generation and classification for emotion-aware dialogue response generation, *in* 'Proc. NAACL', pp. 119–125.
URL: *https://aclanthology.org/2021.naacl-srw.15*

Izacard, G. & Grave, E. (2021), Leveraging passage retrieval with generative models for open domain question answering, *in* 'Proc. EACL', pp. 874–880.
URL: *https://aclanthology.org/2021.eacl-main.74*

Järvelin, K. & Kekäläinen, J. (2002), 'Cumulated gain-based evaluation of ir techniques', *ACM Transactions on Information Systems (TOIS)* pp. 422–446.

Jiang, Z., Gao, L., Wang, Z., Araki, J., Ding, H., Callan, J. & Neubig, G. (2022*a*), Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer, *in* 'Proc. EMNLP', pp. 2336–2349.
URL: *https://aclanthology.org/2022.emnlp-main.149*

Jiang, Z., Gao, L., Wang, Z., Araki, J., Ding, H., Callan, J. & Neubig, G. (2022*b*), Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer, *in* 'Proc. EMNLP', pp. 2336–2349.
URL: *https://aclanthology.org/2022.emnlp-main.149*

Johnson, J., Douze, M. & Jégou, H. (2021), 'Billion-scale similarity search with GPUs', *In Proc. IEEE Transactions on Big Data* **7**(3), 535–547.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D. & Yih, W.-t. (2020), Dense passage retrieval for open-domain question answering, *in* 'Proc EMNLP', pp. 6769–6781.
URL: *https://aclanthology.org/2020.emnlp-main.550*

Kendall, A., Gal, Y. & Cipolla, R. (2018), Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, *in* 'Proc. CVPR', pp. 7482–7491.
URL: *https://arxiv.org/abs/1705.07115*

Keyvan, K. & Huang, J. X. (2022), 'How to approach ambiguous queries in conversational search: A survey of techniques, approaches, tools, and challenges', *ACM Comput. Surv.* .
URL: *https://doi.org/10.1145/3534965*

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P. & Hajishirzi, H. (2020), UnifiedQA: Crossing format boundaries with a single QA system, *in* 'Proc. EMNLP', pp. 1896–1907.
URL: *https://aclanthology.org/2020.findings-emnlp.171*

Khattab, O. & Zaharia, M. (2020), Colbert: Efficient and effective passage search via contextualized late interaction over bert, *in* 'Proc. SIGIR', p. 39–48.
URL: *https://doi.org/10.1145/3397271.3401075*

Kirschner, M. & Bernardi, R. (2007), An empirical view on IQA follow-up questions, *in* 'Proc. SIGdial', pp. 43–46.
URL: *https://www.aclweb.org/anthology/2007.sigdial-1.8*

Kirschner, M. & Bernardi, R. (2009), Exploring topic continuation follow-up questions using machine learning, *in* 'Proc. NAACL', pp. 13–18.
URL: *https://www.aclweb.org/anthology/N09-3003*

Kočiskỳ, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G. & Grefenstette, E. (2018), 'The narrativeqa reading comprehension challenge', *Transactions of the Association for Computational Linguistics* **6**, 317–328.

Kongyoung, S., Macdonal, C. & Ounis, I. (2022), monoQA: Multi-task learning of reranking and answer extraction for open-retrieval conversational question answering, *in* 'Proc. EMNLP'.

Kongyoung, S., Macdonal, C. & Ounis, I. (2023), Multi-task learning of query generation and classification for generative conversational question rewriting, *in* 'Proc. EMNLP'.

Kongyoung, S., Macdonald, C. & Ounis, I. (2020), Multi-task learning using dynamic task weighting for conversational question answering, *in* 'Proc. SCAI', pp. 17–26.
URL: *https://aclanthology.org/2020.scai-1.3*

Krasakis, A. M., Aliannejadi, M., Voskarides, N. & Kanoulas, E. (2020), Analysing the effect of clarifying questions on document ranking in conversational search, *in* 'Proc. ICTIR', New York, NY, USA, p. 129–132.
URL: *https://doi.org/10.1145/3409256.3409817*

Kudo, T. & Richardson, J. (2018), SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, *in* 'Proc. EMNLP', pp. 66–71.
URL: *https://www.aclweb.org/anthology/D18-2012*

Kundu, S., Lin, Q. & Ng, H. T. (2020), Learning to identify follow-up questions in conversational question answering, *in* 'Proc. ACL', pp. 959–968.
URL: *https://www.aclweb.org/anthology/2020.acl-main.90*

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q. & Petrov, S. (2019), 'Natural questions: A benchmark for question answering research', *Transactions of the Association for Computational Linguistics*

7, 452–466.
URL: *https://aclanthology.org/Q19-1026*

Lai, G., Xie, Q., Liu, H., Yang, Y. & Hovy, E. (2017), RACE: Large-scale ReAding comprehension dataset from examinations, *in* 'Proc. EMNLP', pp. 785–794.
URL: *https://aclanthology.org/D17-1082*

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P. & Soricut, R. (2020), ALBERT: A lite bert for self-supervised learning of language representations, *in* 'Proc. ICLR'.
URL: *https://openreview.net/pdf?id=H1eA7AEtvS*

Lee, H., Kedia, A., Lee, J., Paranjape, A., Manning, C. & Woo, K.-G. (2022*a*), You only need one model for open-domain question answering, *in* 'Proc. EMNLP', pp. 3047–3060.
URL: *https://aclanthology.org/2022.emnlp-main.198*

Lee, H., Kedia, A., Lee, J., Paranjape, A., Manning, C. & Woo, K.-G. (2022*b*), You only need one model for open-domain question answering, *in* 'Proc. EMNLP', pp. 3047–3060.
URL: *https://aclanthology.org/2022.emnlp-main.198*

Lee, K., Chang, M.-W. & Toutanova, K. (2019), Latent retrieval for weakly supervised open domain question answering, *in* 'Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics', pp. 6086–6096.
URL: *https://www.aclweb.org/anthology/P19-1612*

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V. & Zettlemoyer, L. (2019), BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, *in* 'Proc. ACL', pp. 7871–7880.
URL: *https://www.aclweb.org/anthology/2020.acl-main.703*

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T. et al. (2020), 'Retrieval-augmented generation for knowledge-intensive NLP tasks', *In Proc. NeurIPS* pp. 9459–9474.
URL: *https://proceedings.neurips.cc/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf*

Li, H., Liu, T., Kang, Y., Xu, G., Ding, W. & Liu, Z. (2020), 'State-of-the-art query2query similarity'.
URL: *https://slideslive.com/38940067/clarifying-questions-in-conversations-with-augmented-syntax-features-and-side-information*

Liang, T., Jiang, Y., Xia, C., Zhao, Z., Yin, Y. & Yu, P. S. (2022), 'Multifaceted improvements for conversational open-domain question answering', *arXiv preprint arXiv:2204.00266* .
URL: *https://doi.org/10.48550/arXiv.2204.00266*

Lin, C.-Y. (2004), ROUGE: A package for automatic evaluation of summaries, *in* 'Text Summarization Branches Out', Association for Computational Linguistics, Barcelona, Spain, pp. 74–81.
URL: *https://aclanthology.org/W04-1013*

Lin, J. & Ma, X. (2021), 'A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques', *arXiv preprint arXiv:2106.14807* .

Lin, S.-C., Yang, J.-H. & Lin, J. (2020), 'Distilling dense representations for ranking using tightly-coupled teachers', *arXiv preprint arXiv:2010.11386* .
URL: *https://doi.org/10.48550/arXiv.2010.11386*

Lin, S.-C., Yang, J.-H. & Lin, J. (2021*a*), Contextualized query embeddings for conversational search, *in* 'Proc. EMNLP', pp. 1004–1015.
URL: *https://aclanthology.org/2021.emnlp-main.77*

Lin, S.-C., Yang, J.-H. & Lin, J. (2021*b*), In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval, *in* 'Proc. RepL4NLP', pp. 163–173.
URL: *https://aclanthology.org/2021.repl4nlp-1.17*

Lin, S.-C., Yang, J.-H., Nogueira, R., Tsai, M.-F., Wang, C.-J. & Lin, J. (2020*a*), 'Conversational question reformulation via sequence-to-sequence architectures and pretrained language models', *arXiv preprint arXiv:2004.01909* .
URL: *https://doi.org/10.48550/arXiv.2004.01909*

Lin, S.-C., Yang, J.-H., Nogueira, R., Tsai, M.-F., Wang, C.-J. & Lin, J. (2020*b*), 'Query reformulation using query history for passage retrieval in conversational search', *ArXiv* **abs/2005.02230**.
URL: *https://arxiv.org/abs/2005.02230*

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H. & Neubig, G. (2021), 'Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing', *arXiv preprint arXiv:2107.13586* .
URL: *https://arxiv.org/abs/2107.13586*

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H. & Neubig, G. (2023), 'Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing', *ACM Computing Surveys* **55**, 1–35.

Liu, S., Liang, Y. & Gitter, A. (2019), Loss-balanced task weighting to reduce negative transfer in multi-task learning, *in* 'Proc. AAAI', Vol. 33, pp. 9977–9978.
URL: *https://www.aaai.org/ojs/index.php/AAAI/article/view/5125*

Liu, T.-Y. et al. (2009), 'Learning to rank for information retrieval', *Foundations and Trends® in Information Retrieval* **3**(3), 225–331.

Liu, X., Gao, J., He, X., Deng, L., Duh, K. & Wang, Y.-y. (2015), Representation learning using multi-task deep neural networks for semantic classification and information retrieval, *in* 'Proc. NAACL', pp. 912–921.
URL: *https://www.aclweb.org/anthology/N15-1092*

Liu, X., He, P., Chen, W. & Gao, J. (2019), Multi-task deep neural networks for natural language understanding, *in* 'Proc. ACL', pp. 4487–4496.
URL: *https://www.aclweb.org/anthology/P19-1441*

Luan, Y., Eisenstein, J., Toutanova, K. & Collins, M. (2021*a*), 'Sparse, dense, and attentional representations for text retrieval', *Transactions of the Association for Computational Linguistics* pp. 329–345.
URL: *https://aclanthology.org/2021.tacl-1.20*

Luan, Y., Eisenstein, J., Toutanova, K. & Collins, M. (2021*b*), 'Sparse, dense, and attentional representations for text retrieval', *Transactions of the Association for Computational Linguistics* **9**, 329–345.

Luo, M., Hashimoto, K., Yavuz, S., Liu, Z., Baral, C. & Zhou, Y. (2022), Choose your QA model wisely: A systematic study of generative and extractive readers for question answering, *in* R. Das, P. Lewis, S. Min, J. Thai & M. Zaheer, eds, 'Proc. SPANLP', pp. 7–22.
URL: *https://aclanthology.org/2022.spanlp-1.2*

Macdonald, C. & Tonellotto, N. (2020), Declarative experimentation in information retrieval using pyterrier, *in* 'Proc. ICTIR', p. 161–168.
URL: *https://doi.org/10.1145/3409256.3409829*

Macdonald, C., Tonellotto, N. & Ounis, I. (2021), On single and multiple representations in dense passage retrieval, *in* 'Proc. IIR'.
URL: *https://ceur-ws.org/Vol-2947/paper5.pdf*

Mallia, A., Khattab, O., Suel, T. & Tonellotto, N. (2021), Learning passage impacts for inverted indexes, *in* 'Proc. SIGIR', p. 1723–1727.
URL: *https://doi.org/10.1145/3404835.3463030*

Mass, Y., Cohen, D., Yehudai, A. & Konopnicki, D. (2021), 'Conversational search with mixed-initiative–asking good clarification questions backed-up by passage retrieval', *arXiv preprint arXiv:2112.07308* .

Maynez, J., Narayan, S., Bohnet, B. & McDonald, R. (2020), On faithfulness and factuality in abstractive summarization, *in* 'Proc. ACL', pp. 1906–1919.
URL: *https://aclanthology.org/2020.acl-main.173*

Mele, I., Muntean, C. I., Nardini, F. M., Perego, R., Tonellotto, N. & Frieder, O. (2021), 'Adaptive utterance rewriting for conversational search', *Information Processing & Management* p. 102682.
URL: *https://www.sciencedirect.com/science/article/pii/S0306457321001679*

Mihaylov, T., Clark, P., Khot, T. & Sabharwal, A. (2018), Can a suit of armor conduct electricity? a new dataset for open book question answering, *in* E. Riloff, D. Chiang, J. Hockenmaier & J. Tsujii, eds, 'Proc. EMNLP', pp. 2381–2391.
URL: *https://aclanthology.org/D18-1260*

Nair, S., Yang, E., Lawrie, D., Duh, K., McNamee, P., Murray, K., Mayfield, J. & Oard, D. W. (2022), Transfer learning approaches for building cross-language dense retrieval models, *in* 'Proc. ECIR', p. 382–396.
URL: *https://doi.org/10.1007/978-3-030-99736-6*$_2$6

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R. & Deng, L. (2016), 'Ms marco: A human generated machine reading comprehension dataset'.
URL: *https://www.microsoft.com/en-us/research/publication/ms-marco-human-generated-machine-reading-comprehension-dataset/*

Ni, J., Qu, C., Lu, J., Dai, Z., Ábrego, G. H., Ma, J., Zhao, V. Y., Luan, Y., Hall, K. B., Chang, M.-W. et al. (2021), 'Large dual encoders are generalizable retrievers', *arXiv preprint arXiv:2112.07899* .
URL: *https://doi.org/10.48550/arXiv.2112.07899*

Nogueira, R. & Cho, K. (2019), 'Passage re-ranking with bert', *arXiv preprint arXiv:1901.04085* .
URL: *https://doi.org/10.48550/arXiv.1901.04085*

Nogueira, R., Jiang, Z. & Lin, J. (2020), Document ranking with a pretrained sequence-to-sequence model, *in* 'Proc. EMNLP', pp. 708–718.
URL: *https://www.aclweb.org/anthology/2020.findings-emnlp.63*

Nogueira, R., Jiang, Z. & Lin, J. (2021), 'Investigating the limitations of the transformers with simple arithmetic tasks', *arXiv preprint arXiv:2102.13019* .
URL: *https://arxiv.org/pdf/2102.13019.pdf*

Nogueira, R., Jiang, Z., Pradeep, R. & Lin, J. (2020*a*), Document ranking with a pretrained sequence-to-sequence model, *in* 'In Proc. EMNLP 2020', pp. 708–718.
URL: *https://aclanthology.org/2020.findings-emnlp.63*

Nogueira, R., Jiang, Z., Pradeep, R. & Lin, J. (2020*b*), Document ranking with a pretrained sequence-to-sequence model, *in* 'Proc. EMNLP', pp. 708–718.
URL: *https://aclanthology.org/2020.findings-emnlp.63*

Ou, W. & Lin, Y. (2020), A clarifying question selection system from ntes_along in convai3 challenge, *in* 'arXiv preprint arXiv:2010.14202'.
URL: *https://doi.org/10.48550/arXiv.2112.07308*

Owoicho, P., Dalton, J., Aliannejadi, M., Azzopardi, L. & Trippas, J. R. (2022), CAsT 2022: Trec cast 2022 going beyond user ask and system retrieve with initiative and response generation, *in* 'Proc. TREC'.

Papineni, K., Roukos, S., Ward, T. & Zhu, W.-J. (2002), Bleu: a method for automatic evaluation of machine translation, *in* 'Proc. ACL', pp. 311–318.
URL: *http://www1.cs.columbia.edu/nlp/sgd/bleu.pdf*

Pillai, I., Fumera, G. & Roli, F. (2017), 'Designing multi-label classifiers that maximize f measures: State of the art', *Pattern Recognition* **61**, 394–404.
URL: *https://www.sciencedirect.com/science/article/pii/S0031320316302217*

Powers, D. M. (2020), 'Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation', *Journal of Machine Learning Technologies* pp. 37–63.
URL: *https://doi.org/10.48550/arXiv.2010.16061*

Qu, C., Yang, L., Chen, C., Croft, W. B., Krishna, K. & Iyyer, M. (2021), Weakly-supervised open-retrieval conversational question answering, *in* 'Proc. ECIR'.
URL: *https://arxiv.org/pdf/2103.02537.pdf*

Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W. B. & Iyyer, M. (2020), Open-retrieval conversational question answering, *in* 'Proc. SIGIR', pp. 539–548.
URL: *https://doi.org/10.1145/3397271.3401110*

Qu, C., Yang, L., Qiu, M., Croft, W. B., Zhang, Y. & Iyyer, M. (2019), BERT with history answer embedding for conversational question answering, *in* 'Proc. SIGIR', pp. 1133–1136.
URL: *https://arxiv.org/abs/1905.05412*

Qu, C., Yang, L., Qiu, M., Zhang, Y., Chen, C., Croft, W. B. & Iyyer, M. (2019), Attentive history selection for conversational question answering, *in* 'Proc. CIKM', pp. 1391–1400.
URL: *https://arxiv.org/abs/1908.09456*

Radev, D. R., Qi, H., Wu, H. & Fan, W. (2002), Evaluating web-based question answering systems., *in* 'LREC', Citeseer.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. et al. (2018), 'Improving language understanding by generative pre-training', *OpenAI* .
URL: *https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language$_u$nderstanding$_p$aper.pdf*

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. (2019), 'Language models are unsupervised multitask learners', *OpenAI blog* p. 9.
URL: *http://www.persagen.com/files/misc/radford2019language.pdf*

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. & Liu, P. J. (2020), Exploring the limits of transfer learning with a unified text-to-text transformer, *in* 'Proc. JMLR', pp. 1–67.
URL: *http://jmlr.org/papers/v21/20-074.html*

Rajpurkar, P., Jia, R. & Liang, P. (2018), Know what you don't know: Unanswerable questions for SQuAD, *in* 'Proc. ACL', pp. 784–789.
URL: *https://www.aclweb.org/anthology/P18-2124*

Rajpurkar, P., Zhang, J., Lopyrev, K. & Liang, P. (2016), SQuAD: 100,000+ questions for machine comprehension of text, *in* 'Proc. EMNLP', p. 2383–2392.
URL: *https://arxiv.org/pdf/1606.05250.pdf*

Rao, S. & Daumé III, H. (2018), Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information, *in* 'Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', pp. 2737–2746.
URL: *https://aclanthology.org/P18-1255*

Reddy, S., Chen, D. & Manning, C. D. (2019), CoQA: A conversational question answering challenge, *in* 'Proc. ACL', pp. 249–266.
URL: *https://www.aclweb.org/anthology/Q19-1016*

Ren, G., Ni, X., Malik, M. & Ke, Q. (2018), Conversational query understanding using sequence to sequence modeling, *in* 'Proc. WWW', pp. 1715–1724.
URL: *https://dl.acm.org/doi/pdf/10.1145/3178876.3186083*

Robertson, S. E. & Walker, S. (1994), Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval, *in* 'SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University', Springer, pp. 232–241.

Rosset, C., Xiong, C., Song, X., Campos, D., Craswell, N., Tiwary, S. & Bennett, P. (2020), Leading conversational search by suggesting useful questions, *in* 'Proceed. WWW', pp. 1160–1170.

Ruder, S. (2017), 'An overview of multi-task learning in deep neural networks', *arXiv preprint arXiv:1706.05098* .
URL: *https://arxiv.org/abs/1706.05098*

Salton, G. (1971), *The SMART Retrieval System—Experiments in Automatic Document Processing*, Prentice-Hall, Inc., USA.

Seo, M., Kembhavi, A., Farhadi, A. & Hajishirzi, H. (2017), 'Bidirectional attention flow for machine comprehension', *Proc. ICLR* .

Standley, T., Zamir, A., Chen, D., Guibas, L., Malik, J. & Savarese, S. (2020), Which tasks should be learned together in multi-task learning?, *in* 'Proc. PMLR', pp. 9120–9132.
URL: *https://proceedings.mlr.press/v119/standley20a.html*

Sun, S., Shawe-Taylor, J. & Mao, L. (2017), 'Pac-bayes analysis of multi-view learning', *Information Fusion* **35**, 117–131.

Sun, X., Panda, R., Feris, R. & Saenko, K. (2020), 'Adashare: Learning what to share for efficient deep multi-task learning', *In Proc. NeurIPS* **33**, 8728–8740.
URL: *https://doi.org/10.48550/arXiv.1911.12423*

Sun, Z., Sarma, P. K., Liang, Y. & Sethares, W. (2021), A new view of multi-modal language analysis: Audio and video features as text "styles", *in* 'Proc. ACL', pp. 1956–1965.
URL: *https://aclanthology.org/2021.eacl-main.167*

Taher, E., Hoseini, S. A. & Shamsfard, M. (2019), Beheshti-NER: Persian named entity recognition using BERT, *in* 'Proc. NSURL', pp. 37–42.
URL: *https://aclanthology.org/2019.nsurl-1.6*

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A. & Gurevych, I. (2021), BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models, *in* 'Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)'.
URL: *https://openreview.net/forum?id=wCu6T5xFjeJ*

Vakulenko, S., Longpre, S., Tu, Z. & Anantha, R. (2021), Question rewriting for conversational question answering, *in* 'Proc. WSDM', pp. 355–363.
URL: *https://dl.acm.org/doi/pdf/10.1145/3437963.3441748*

Vakulenko, S., Voskarides, N., Tu, Z. & Longpre, S. (2021), A comparison of question rewriting methods for conversational passage retrieval, *in* 'Proc. ECIR 2021', p. 418–424.
URL: *https://arxiv.org/pdf/2101.07382.pdf*

van Rijsbergen, C. J. (1979), 'Information retrieval', *London: Butterworths* .
URL: *https://www.dcs.gla.ac.uk/Keith/Preface.html*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017), 'Attention is all you need', *Advances in neural information processing systems* **30**.

Voorhees, E., Alam, T., Bedrick, S., Demner-Fushman, D., Hersh, W. R., Lo, K., Roberts, K., Soboroff, I. & Wang, L. L. (2021), 'Trec-covid: Constructing a pandemic information retrieval test collection', *SIGIR Forum* .
URL: *https://doi.org/10.1145/3451964.3451965*

Voorhees, E. M. (2003), Common evaluation measures, *in* 'Proc. Trec'.

Voorhees, E. M. et al. (1999), The trec-8 question answering track report., *in* 'Trec', Vol. 99, pp. 77–82.

Voskarides, N., Li, D., Ren, P., Kanoulas, E. & de Rijke, M. (2020), Query resolution for conversational search with limited supervision, *in* 'Proc. SIGIR', pp. 921–930.
URL: *https://arxiv.org/abs/2005.11723*

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. & Bowman, S. (2018), GLUE: A multi-task benchmark and analysis platform for natural language understanding, *in* 'Proc. EMNLP', pp. 353–355.
URL: *https://aclanthology.org/W18-5446*

Wang, J. & Li, W. (2021), Template-guided clarifying question generation for web search clarification, *in* 'Proc. CIKM', p. 3468–3472.
URL: *https://doi.org/10.1145/3459637.3482199*

Wang, S., Zhuang, S. & Zuccon, G. (2021), Bert-based dense retrievers require interpolation with bm25 for effective passage retrieval, *in* 'Proc. ICTIR', p. 317–324.
URL: *https://doi.org/10.1145/3471158.3472233*

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N. & Zhou, M. (2020), Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, *in* 'Proc. NIPS'.
URL: *https://dl.acm.org/doi/pdf/10.5555/3495724.3496209*

Wang, X., MacAvaney, S., Macdonald, C. & Ounis, I. (2021*a*), University of glasgow terrier team at the trec 2021 deep learning track., *in* 'TREC'.
URL: *https://trec.nist.gov/pubs/trec30/papers/uogTr-DL.pdf*

Wang, X., MacAvaney, S., Macdonald, C. & Ounis, I. (2021*b*), University of glasgow terrier team at the trec 2021 deep learning track., *in* 'TREC'.
URL: *https://trec.nist.gov/pubs/trec30/papers/uogTr-DL.pdf*

Wang, X., Macdonald, C., Tonellotto, N. & Ounis, I. (2021), Pseudo-relevance feedback for multiple representation dense retrieval, *in* 'Proc. ICTIR', pp. 297–306.

Wang, X., Wu, Y., Wang, X., Macdonald, C. & Ounis, I. (2020), University of glasgow terrier team at the trec 2020 deep learning track., *in* 'TREC'.

Wang, Z., Chen, S. & Gao, D. (2011), 'A novel multi-view learning developed from single-view patterns', *Pattern Recognition* **44**(10-11), 2395–2413.

Wen, L., Wang, H., Luo, Y. & Wang, X. (2022), M3: A multi-view fusion and multi-decoding network for multi-document reading comprehension, *in* 'Proc. EMNLP', pp. 1450–1461.
URL: *https://aclanthology.org/2022.emnlp-main.94*

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q. & Rush, A. (2020), Transformers: State-of-the-art natural language processing, *in* 'Proc. EMNLP', pp. 38–45.
URL: *https://www.aclweb.org/anthology/2020.emnlp-demos.6*

Worsham, J. & Kalita, J. (2020), 'Multi-task learning for natural language processing in the 2020s: where are we going?', *Pattern Recognition Letters* pp. 120–126.

Wu, Z., Parish, R., Cheng, H., Min, S., Ammanabrolu, P., Ostendorf, M. & Hajishirzi, H. (2022), Inscit: Information-seeking conversations with mixed-initiative interactions, *in* 'arXiv'.
URL: *https://doi.org/10.48550/arXiv.2207.00746*

Xie, Z., Chen, J., Feng, Y., Zhang, K. & Zhou, Z. (2022), 'End to end multi-task learning with attention for multi-objective fault diagnosis under small sample', *Journal of Manufacturing Systems* **62**, 301–316.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0278612521002521*

Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J. & Overwijk, A. (2020), Approximate nearest neighbor negative contrastive learning for dense text retrieval, *in* 'Proc. ICLR'.
URL: *https://arxiv.org/pdf/2007.00808.pdf*

Xu, Y., Liu, X., Shen, Y., Liu, J. & Gao, J. (2019*a*), Multi-task learning with sample re-weighting for machine reading comprehension, *in* 'Proc. NAACL', pp. 2644–2655.
URL: *https://www.aclweb.org/anthology/N19-1271*

Xu, Y., Liu, X., Shen, Y., Liu, J. & Gao, J. (2019*b*), Multi-task learning with sample re-weighting for machine reading comprehension, *in* 'Proc. NAACL', pp. 2644–2655.
URL: *https://aclanthology.org/N19-1271*

Yang, J.-H., Ma, X. & Lin, J. (2021), 'Sparsifying sparse representations for passage retrieval by top-*k* masking', *arXiv preprint arXiv:2112.09628* .
URL: *https://arxiv.org/abs/2112.09628*

Yang, Q., Zhang, Y., Dai, W. & Pan, S. J. (2020), *Transfer Learning*, Cambridge University Press.

Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M. & Lin, J. (2019), End-to-end open-domain question answering with BERTserini, *in* 'Proc. NAACL', pp. 72–77.
URL: *https://aclanthology.org/N19-4013*

Yang, Y. & Hospedales, T. M. (2017), Trace norm regularised deep multi-task learning, *in* 'Proc. ICLR'.
URL: *https://doi.org/10.48550/arXiv.1606.04038*

Yatskar, M. (2019), A qualitative comparison of CoQA, SQuAD 2.0 and QuAC, *in* 'Proc. NAACL', pp. 2318–2323.
URL: *https://www.aclweb.org/anthology/N19-1241*

Yeh, Y.-T. & Chen, Y.-N. (2019), FlowDelta: Modeling flow information gain in reasoning for conversational machine comprehension, *in* 'Proc. MRQA', pp. 86–90.
URL: *https://www.aclweb.org/anthology/D19-5812*

Yu, S., Liu, J., Yang, J., Xiong, C., Bennett, P., Gao, J. & Liu, Z. (2020), Few-shot generative conversational query rewriting, *in* 'Proc. SIGIR', p. 1933–1936.
URL: *https://doi.org/10.1145/3397271.3401323*

Yu, S., Liu, Z., Xiong, C., Feng, T. & Liu, Z. (2021), Few-shot conversational dense retrieval, *in* 'Proc. SIGIR', pp. 829–838.
URL: *https://doi.org/10.1145/3404835.3462856*

Zamani, H., Dehghani, M., Croft, W. B., Learned-Miller, E. & Kamps, J. (2018), From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing, *in* 'Proc. CIKM', p. 497–506.
URL: *https://doi.org/10.1145/3269206.3271800*

Zamani, H., Dumais, S., Craswell, N., Bennett, P. & Lueck, G. (2020), Generating clarifying questions for information retrieval, *in* 'Proc. WWW', p. 418–428.
URL: *https://doi.org/10.1145/3366423.3380126*

Zamani, H., Trippas, J. R., Dalton, J. & Radlinski, F. (2022), 'Conversational information seeking', *arXiv preprint arXiv:2201.08808* .
URL: *https://doi.org/10.48550/arXiv.2201.08808*

Zangerle, E., Gassler, W. & Specht, G. (2013), 'On the impact of text similarity functions on hashtag recommendations in microblogging environments', *Social network analysis and mining* **3**, 889–898.

Zhang, M.-L. & Zhou, Z.-H. (2014), 'A review on multi-label learning algorithms', *IEEE Transactions on Knowledge and Data Engineering* pp. 1819–1837.

Zhang, S., Liang, Y., Gong, M., Jiang, D. & Duan, N. (2022), Multi-view document representation learning for open-domain dense retrieval, *in* 'Proc. ALC', pp. 5990–6000.
URL: *https://aclanthology.org/2022.acl-long.414*

Zhang, Y. & Yang, Q. (2021), 'A survey on multi-task learning', *IEEE Transactions on Knowledge and Data Engineering* **34**(12), 5586–5609.

Zhao, J., Xie, X., Xu, X. & Sun, S. (2017), 'Multi-view learning overview: Recent progress and new challenges', *Information Fusion* **38**, 43–54.

Zhu, C., Zeng, M. & Huang, X. (2018), 'SDNet: Contextualized attention-based deep network for conversational question answering', *arXiv preprint arXiv:1812.03593* .
URL: *https://arxiv.org/abs/1812.03593*

Zhuang, H., Qin, Z., Jagerman, R., Hui, K., Ma, J., Lu, J., Ni, J., Wang, X. & Bendersky, M. (2022), 'Rankt5: Fine-tuning t5 for text ranking with ranking losses', *arXiv preprint arXiv:2210.10634* .

Zhuang, L., Wayne, L., Ya, S. & Jun, Z. (2021), A robustly optimized BERT pre-training approach with post-training, *in* 'Proceedings of the 20th Chinese National Conference on Computational Linguistics', pp. 1218–1227.
URL: *https://aclanthology.org/2021.ccl-1.108*