Image Classification with Foveated Neural Networks

George William Killick

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

September 2024

Abstract

Foveated vision draws inspiration from the way many biological vision systems process visual information. It features space-variant resolution, concentrating high-resolution sampling in a small area known as the fovea. This approach aims to deliver the same visual acuity and field of view as uniform vision but with significantly fewer pixels. This reduction in pixel count can potentially lower the computational demands of subsequent visual processes without compromising their effectiveness in performing visual perception tasks. Despite these qualities of foveated vision, a uniform approach remains the dominant paradigm in computer vision. This thesis investigates the use of deep neural networks on foveated images, aiming to determine whether foveated vision can improve the ability of such systems to classify challenging datasets comprised of natural images.

In Chapter 1, we outline the motivations for exploring foveated vision in conjunction with deep neural networks, the research gaps, and the corresponding questions that we aim to answer through this thesis. Furthermore, we provide an overview of biological vision processes, computational models of foveated vision, and the relationship between foveated vision systems and the active vision paradigm.

In Chapter 2, we explore the application of convolutional neural networks (CNNs) to foveated images. Prior works have frequently shown that foveated sampling does not improve the accuracy of CNNs. Motivated by this observation, we analyse the implications of convolutional processing of foveated images through the lens of geometric deep learning. We hypothesise that the application of CNNs to foveated images often requires imposing a suboptimal coordinate frame for representing foveated image data, inhibiting classification accuracy. We test this hypothesis through a novel graph convolution layer that allows for coordinate frames to be freely defined. We show that the classification accuracy of a foveated CNN is highly sensitive to the choice of coordinate frame.

In Chapter 3, we expand upon the studies conducted in Chapter 2 and explore foveated CNNs in the presence of visual attention to guide the sensor. We propose a two-stage approach where a separate CNN first localises objects, informing the foveated classifier where to centre its gaze. Empirical results corroborate the findings of the previous chapter on the importance of coor-

dinate frames. Furthermore, our novel graph convolution layer allows us to build a foveated CNN that significantly outperforms a uniform CNN under an equivalent pixel budget. Furthermore, we propose a novel foveated sensor with a parameterised sampling layout. We show the sensitivity of classification accuracy to this parameterisation and find that having smaller higher-resolution foveae for sensors with fewer pixels is favourable.

In Chapter 4, we conduct studies similar to those in Chapter 3, but in the context of non-convolutional models such as vision transformers. We propose a simple reformulation of image tokenisation to a foveated setting. We also show how the sampling layout of this method can be optimised by backpropagation using only gradients from a classification loss. We show that foveated sensing can improve the classification accuracy of these models and is increasingly beneficial as the number of total pixels in the sensor decreases. Furthermore, we explore the parameterisation of the sampling layout and how the optimal configuration is related to the properties of the data itself. We show that as the range in the scale of objects increases, it becomes increasingly beneficial to have smaller, higher resolution fovea in order to classify objects accurately at all scales.

Chapter 5 explores a sequential approach for foveated vision systems, where they can repeatedly attend to an image. For each observation, feature vectors are computed using a foveated CNN, integrated into a single representation, and used as input to a classifier. Despite using only a single dedicated convolution layer to implement attention, we show that these models can perform as well as a two-stage method where a dedicated CNN is used to perform attention. Furthermore, we show that classification accuracy increases the more times the model attends to an image and that a simple averaging approach suffices for integrating information from multiple observations. Finally, we explore an architecture based on vision transformers that maintains a memory of previous observations in all hidden layers. We show that Legendre Memory Units can effectively replace self-attention and allow such a system to run in O(1) time complexity, as opposed to self-attention's O(N) complexity, where N is the number of previous observations.

In Chapter 6, we summarise the contributions made in this thesis in relation to the research questions we set out to answer and provide several avenues for future work that can further the field of foveated vision.

This work was supported by the Engineering and Physical Sciences Research Council, grant number 2443519, and has appeared in the following papers:

1. Killick, G., Aragon-Camarasa, G. and Siebert, J.P., 2022. Monte-Carlo Convolutions on Foveated Images. (VISAPP2022)

2. Killick, G., Henderson, P., Siebert, P. and Aragon-Camarasa, G., 2023. Foveation in the Era of Deep Learning. (BMVC2023)

Contents

Abstract **i**

Acknowledgements **xiv**

1 Introduction **1**

1.1 Thesis Statement . . . 1
1.2 Motivation . . . 1
1.3 Background . . . 2
1.3.1 Biological Vision . . . 2
1.3.2 Foveated Vision . . . 5
1.3.3 Active Vision . . . 8
1.4 Research Gap . . . 9
1.5 Research Questions . . . 10
1.5.1 Thesis Structure . . . 12

2 Convolution on Non-Grid Structured Visual Data **13**

2.1 Introduction . . . 13
2.1.1 Chapter Structure . . . 14
2.2 Background . . . 15
2.2.1 Inductive Biases . . . 15
2.2.2 Equivariance and Invariance . . . 15
2.2.3 Geometric Deep Learning . . . 16
2.3 Related Work . . . 18
2.3.1 Foveated Convolutional Neural Networks . . . 18
2.3.2 Deep Learning on Non-Grid Structured Domains . . . 20
2.4 Graph Convolution on Foveated Images . . . 22
2.4.1 Graph Construction . . . 22
2.4.2 Edge Conditioned Filters . . . 23
2.4.3 Motivation for Coordinate Frame and Filter Scaling . . . 24
2.4.4 Implicit Neural Filters . . . 25

2.4.5 Foveated Sampling . . . 26
2.4.6 Architecture . . . 27
2.5 Image Classification on Imagewoof . . . 27
2.5.1 Training Details and Hyperparameters . . . 29
2.5.2 Results . . . 29
2.5.3 Effect of Depth, Hidden Size and Omega Hyperparameters on Classification Accuracy . . . 31
2.6 Limitations . . . 33
2.7 Conclusion . . . 34

3 Foveated Convolutional Neural Networks 36
3.1 Introduction . . . 36
3.1.1 Chapter Structure . . . 37
3.2 Architecture Overview . . . 37
3.3 Localisation Network . . . 38
3.4 Differentiable GPU Accelerated Sampling . . . 40
3.5 Sunflower Foveated Sensor . . . 40
3.6 Basis Filters . . . 42
3.7 ImageNet-100 Classification Experiments . . . 45
3.7.1 Implementation Details . . . 45
3.7.2 Training Hyperparameters . . . 47
3.8 Results and Discussion . . . 48
3.8.1 Foveated vs. Uniform CNNs . . . 49
3.8.2 Discussion on Coordinate Frames . . . 49
3.8.3 Discussion on Biologically Implausible Methods . . . 51
3.9 Comparison of Different Basis Functions . . . 51
3.10 Fovea Radius . . . 52
3.11 Limitations . . . 55
3.12 Conclusion . . . 55

4 Non Convolutional Foveated Vision Architectures 57
4.1 Introduction . . . 57
4.1.1 Chapter Structure . . . 58
4.2 Background . . . 59
4.3 Methods and Materials . . . 61
4.4 Patch Based Foveated Sensor . . . 61
4.5 Learnable Sampling Layout . . . 63
4.6 Architecture . . . 64
4.7 Experiments . . . 64

4.7.1 Implementation Details . . . 64
4.7.2 Object Recognition on Imagenet-100 . . . 65
4.7.3 Learnable Sampling Layout . . . 67
4.7.4 To Crop or Not to Crop . . . 68
4.8 MNIST Experiments . . . 70
4.9 Limitations . . . 72
4.10 Conclusion . . . 74

5 Sequential Active Vision Architectures 76
5.1 Introduction . . . 76
5.1.1 Chapter Structure . . . 77
5.2 Background and Related Work . . . 78
5.2.1 Memory in Neural Networks . . . 78
5.2.2 Sequential Vision Systems . . . 79
5.3 A Simple Sequential Model . . . 82
5.4 Classifying ImageNet100 with Sequential Foveated CNNs . . . 83
5.4.1 Implementation Details . . . 84
5.4.2 Classification Accuracy vs. Time Steps . . . 84
5.4.3 Contrasting Against a Two Stage Approach . . . 85
5.5 Memory in Sequential Foveated CNNs . . . 86
5.5.1 Implementation Details . . . 86
5.5.2 Results on ImageNet100 . . . 87
5.6 Optimal Sensor Layout in a Sequential Context . . . 87
5.6.1 Implementation Details . . . 88
5.6.2 Results on Imagewoof . . . 89
5.7 Memory for Intermediate Layers . . . 90
5.7.1 Transformers as Sequential Vision Models . . . 90
5.7.2 Fixation Policies . . . 91
5.7.3 Implementation Details . . . 92
5.7.4 Results on ImageNet100 . . . 94
5.8 Limitations . . . 95
5.9 Conclusion . . . 95

6 Conclusion 97
6.1 Contributions . . . 97
6.1.1 Summary . . . 101
6.2 Future Work . . . 101
6.2.1 Image Classification Datasets for Foveated Vision Systems . . . 101
6.2.2 Beyond Image Classification . . . 102

6.2.3 Architectural Design and Training Schemes 103
6.2.4 Biologically Inspired Models 105

List of Tables

2.1 Classification accuracy on Imagewoof using different sensors and convolution methods. Log-polar (B.S) refers to the log-polar blindspot method for generating foveated images. Resolution refers to the number of pixels in the sensor. G refers to the learnable function that maps edge labels to filter values in the graph convolution. All results are reported on a held-out test set, averaged over three independent training runs from different random seeds. 30

3.1 Stem refers to the initial convolution layer in the ConvNeXt architecture, Blocks refers to the configuration of the depthwise convolution layers in the ConvNeXt Blocks. Downsampling refers to the configuration of the downsampling layers that reduce spatial dimensionality between stages in the ConvNeXt architecture. Kernel size is analogous to kernel size in ordinary convolution layers, presented as the total number of spatial elements in the filter. Sigma determines the size of Gaussian derivative basis filters; max order refers to the maximum order of partial derivatives used in the basis. 48

3.2 Top-1 Accuracy on Imagenet100. We split the table into two sections. Top: Passive vision models. Bottom: Active vision models. GFLOPs for graph convolutional models are reported as local filters but are trained with global filters and masking for computational efficiency. 48

4.1 Top-1 accuracy (Mean and SD over 3 random seeds) on Imagenet100 under a variety of different image sampling methods and vision backbones. ResMLP-49 indicates a 49 patch resmlp, with other model names analogously defined. Methods with '*' indicate that a separate network adapts how the image is sampled (e.g. a localisation network). Baseline refers to backbones in their conventional form i.e. no fixation mechanism. 66

4.2 ImageNet-100 classification accuracy with 49 patch and 196 patch ResMLP-S12 models. We consider two variants of each model, one with a fixed foveated sensor in which we vary the parameterisation by adjusting the radius of the fovea and a learned sensor in which the sampling density over the visual field is optimized jointly with the network weights. For the learned sensor we show the absolute difference in accuracy between the worst (left) and best (right) performing fixed sensor parameterisations. 67

5.1 Top-1 Accuracy on the Imagenet100 test set. We split the table into three sections. Top: non-attentive models. Middle: Spatial Transformer like models. Bottom: Sequential Models. 84

5.2 Results on the ImageNet100 test set. We omit Positional Encoding for the Averaging method as it has no mechanism to make use of this information. 87

5.3 Classification Accuracy on Imagenet100 test set when using different methods for aggregating visual information from different fixations. Identity refers to the case when no aggregation mechanism is used (i.e. the layer is replaced by an identity transform). We omit results for self-attention with no Causal mask and identity for random policies as they are equivalent to the raster scan policy . . . 93

List of Figures

1.1 A foveated arrangement of 1024 pixels generated by a Self-Similar Neural Network [1] with a fovea radius of 20% of the field of view. The Self-Similar Neural Network yields a locally pseudo-uniform lattice, which presents challenges for neural systems such as convolutional neural networks to process as the data does not lie on a regular grid. 3

1.2 Figure reproduced from Malkin et al. [2]. Left: visual acuity across the field of view for different animals. Lighter areas denote higher visual acuity. Middle: An image sampled and displayed with the corresponding resolution of each animal. Right: A representative image of each animal's habitat and how it might be sampled and represented in their visual field. Each animal displays a distinctive space-variant strategy for sampling the visual world. 6

2.1 A commutative diagram visualizing the equivariance of convolution layers with respect to transformations from the translation group. 16

2.2 Visualisation of the same transformation in different coordinate frames. In log-polar coordinates, translating visual data corresponds to rotating and scaling visual data, centred at the origin. Hence, translation equivariant functions applied to log-polar images are equivariant to Cartesian rotation and scaling. 17

2.3 Visualisation of the local spatial connections between vertices U and output vertices V with edges removed for visualisation purposes. Using K-nearest neighbours leads to a fixed number of elements for each receptive field $R(v)$, which scales the spatial support of the receptive field as sampling rate decreases. . . . 23

2.4 Visualisation of the spatial offsets of pixels in a receptive field, relative to the centre of the receptive field for grid-structured images and non-grid structured images. On grid-aligned structures, the offsets are a discrete set of (x, y) coordinates, that are consistent for all receptive fields. On non-grid-structures, the offsets for a receptive field can be viewed as being drawn from a distribution of (x,y) coordinates. We visualize the offsets for all receptive fields in blue, and the offsets for a single receptive field in pink. 25

2.5 Visualisation of the process of generating edge conditioned filter weights for an arbitrary arrangement of pixels. A) shows the spatial positions of pixels where $(\delta_{xi}, \delta_{yi})$ is the normalized spatial offset between the i^{th} pixel and the centre of the receptive field. B) Spatial offsets are computed for all pixels in the receptive field. C) an MLP maps each spatial offset to a filter value. D) Visualisation of the underlying filter function represented by an MLP. Orange points indicate where the function is evaluated. 26
2.6 Schematic overview of the Isometric Foveated Graph ConvNeXt. 28
2.7 The effect of increasing the hidden size for both Sine and ReLU activated MLPs on classification accuracy on Imagewoof. Larger sizes allow for more expressive functions to be modeled, however we show scaling the size does not yield significant performance gains. This suggests that small MLPs should suffice to convolutional filters. 32
2.8 The effect of ω, the pre-activation scaling in Sine activated MLPs, on classification accuracy on Imagewoof. We consider an MLP of hidden size 9 and 1 hidden layer in these experiments. 32
2.9 The effect of increasing MLP depth on classification accuracy on Imagewoof. Much like scaling the hidden size, increasing the depth of the MLP only marginally improves performance 33

3.1 Visualisation of the forwards and backwards pass of the foveated CNN with fixations. The localisation network computes a single channel feature map which the attention module uses to compute an (x, y) coordinate where the foveated sensor should centre its gaze. Differentiable sampling is used to backpropagate through the foveated sampling stage to optimize the localisation network using only gradients from the cross-entropy loss. 38
3.2 Visualisation of the attention module. Left: The input image. Middle: The 1 channel saliency map computed by the localisation network. Right: visualisation of the fixation coordinate computed by the soft argmax operation respective to the saliency map. 39
3.3 Left to Right: Vogel's model of a sunflower capitulum [3], our foveated adaptation (eq. 3.2) where fovea sampling density is well parameterised, and our adaptation where the fovea sampling density is poorly parameterised. 41
3.4 Visualisation of Gaussian Derivative Basis Functions. The order of the partial derivative in the y direction increases from left to right, and the x direction increases from top to bottom. Our choice of basis functions for a given layer is determined by the hyperparameter M. The basis includes all Gaussian derivatives where the maximum order of their partial derivatives is $\leq M$ 43

3.5 Visualisation of a Cosine Basis, u and v refer to the frequency in the horizontal and vertical directions respectively. The Gaussian derivative basis bears resemblance to a Cosine basis multiplied by a Gaussian windowing function. 44
3.6 Visualisation of the real and imaginary components of a Fourier basis. u and v refer to the horizontal and vertical frequencies respectively. Directional frequencies are given by $\frac{u}{v}$ 45
3.7 A visualisation of different foveated sampling methods and the resultant image. From top to bottom. log-polar (not used in experiments), Blindspot log-polar, Cartesian Foveal Geometry and Multi-FoV crops. 46
3.8 Visualisation of the Cartesian Foveal Geometry sampling layout in Cartesian Coordinates. In red, we show the shape of 7×7 filters on various parts of the image. As the filter moves across the field, its shape is transformed from square in the fovea, to kite like in the corners. We can describe the shape of the filter as it moves across the visual field through a projective transform. 50
3.9 Top-1 accuracy on Imagewoof for Foveated Isometric Graph ConvNeXts when using different edge-conditioned filter methods. INF refers to Implicit Neural Filters 2.4.4. For sake of comparison between Chapter 2. we perform experiments with and without a fixation mechanism. All architectures use a fovea radius of 30%. 53
3.10 Top-1 accuracy on Imagewoof for different fovea radii. We contrast three architectures. ConvNeXt atto operating on a 112^2 foveated sensor, and two Isometric ConvNeXt architectures (depth 8) operating on 112^2 and 56^2 foveated sensors. Each experiment was run three times with mean and standard deviation reported. 54

4.1 Visualisation of the proposed patch sensor. Blue points show the spatial positions of sampling kernels for the entire sensor. Red points show the spatial centres of each patch. These centres are generated by the Sunflower sensor proposed in Section 3.5; however, other methods for generating a foveated arrangement of points are viable. 62
4.2 Schematic for non-convolutional architectures. The CNN localisation adapts the sensor, in this case a foveated sensor. The sensor extracts image tokens and feeds them to a non-convolutional architecture which then makes a class prediction. The primary difference between all methods evaluated in the next section is the choice of sensor. 65
4.3 Graph of ζ_x for different fovea radii parameterisations of the Sunflower Sensor 3.5 (solid lines) and a learned parameterisation represented by a polynomial (dashed line). Darker lines represent better performing parameterisations on Imagenet100 and 49 Patch ResMLP. 68

4.4 Graph of ζ_x for different fovea radii parameterisations of the Sunflower Sensor 3.5 (solid lines) and a learned parameterisation represented by a polynomial (dashed line). Darker lines represent better performing parameterisations on Imagenet100 and 196 Patch ResMLP. 69
4.5 Illustration of the region of an input image that is sampled by the learnable crop sensor for different values of S. Note that the cropped region is always resampled to a consistent size (e.g. 224×224). As such, a lower value of S yields a higher resolution sampling of the input image. 69
4.6 Visualisation of MNIST digits with different values for M. As M increases, digits take up a smaller percentage of the full image. 71
4.7 Classification Accuracy of Foveated Neural Networks with different fovea radii on the scaled MNIST dataset. Each subfigure plots the classification accuracy at different scales, with each subfigure representing a different scale distribution (or scale range). Characterized by an approximately flat response in classification accuracy, we can show that foveation is helpful in classifying objects that exhibit a high dynamic range of scale variation. 73

5.1 Overview of the simple sequential model. At each time step, a foveated CNN makes a class prediction followed and a fixation location for the next time step. This fixation is used to adjust the sensor at the next time step. After T iterations the class predictions are averaged to arrive at a final class prediction. 83
5.2 Classification accuracy on the Imagewoof test set for different fovea radii and number of timesteps. The optimal fovea radius is largely invariant to the number of time steps the network is allowed to perform. 88
5.3 Lines of best fit for Figure 5.2. In general steeper lines can be observed for small fovea radii, suggesting that they benefit more greatly from attending to images multiple times. 89
5.4 Schematic overview of our sequential model based on a reformulation of vision transformers. The red region indicates the small cropped region that is viewed on a given time step. This patch is flattened and linearly projected, before being processed by N transformer blocks ($N = 12$ in our experiments). Intermediate representations from previous time steps are used in future time steps through the form of a memory module (we use an LMU as an example). 91
5.5 Visualisation of fixation patterns for each policy for two different samples. Each square represents a cropped region viewed at a given time step. The raster scan policy scans left to right, top to bottom and is the same for all samples. The random policy randomly attends to the image and is different for each sample. . 93

6.1 A visual search task where the system must find the orange triangle among blue triangles and orange circles. 104

Acknowledgements

Firstly, I would like to express my gratitude to Dr. Paul Siebert, my supervisor for the past six years. Paul introduced me to this field of research and encouraged me to pursue a PhD. Throughout this time, he has been an unwavering source of support and inspiration, for which I am deeply thankful.

I am also profoundly grateful to my co-supervisor, Dr. Gerardo Aragon-Camarasa, whose support and instrumental guidance have been crucial in realizing the work presented in this thesis. This research would not be what it is without his contributions.

I would like to thank Dr. Nicolas Pugeault, Dr. Emma Li, and Dr. Paul Henderson for their critical reviews and invaluable feedback throughout various stages of my PhD. A special thanks to Dr. Paul Henderson for his assistance with the BMVC2023 paper and his general support.

I would like to also thank members of the CVAS group who patiently endured numerous presentations on Foveated vision over the years. In particular, I would like to thank David, Lipeng, Florent and Ollie, whom I shared an office with, and made days of training neural networks in a basement an enjoyable experience. I am also thankful to Piotr, Daniela, Ozan and Richard for their technical and personal support throughout my PhD.

To my friends in Glasgow, thank you for making the past nine years an unforgettable experience, which I will look back on fondly forever. In particular, a heartfelt thanks to Aimee for her unwavering support, especially during the challenging moments of this PhD. I couldn't have done it without you.

Finally, I am eternally grateful to my family for making my years as a student possible and for their continuous support. I will always be forever thankful for their encouragement and love.

Chapter 1

Introduction

1.1 Thesis Statement

Computer vision systems typically operate on uniformly sampled visual data. In contrast, many biological systems exhibit space-variant (or foveated) resolution where high-resolution sampling is limited to a small area of their field-of-view [4]. The emergence of this strategy in many species suggests it could play an important role in visual perception. Accordingly, this thesis aims to build and assess a range of foveated computer vision systems and ascertain whether such a strategy can also provide important functional benefits for computer vision systems based on deep neural networks.

1.2 Motivation

Foveated vision systems allocate high-resolution visual processing to a small central region of the field of view (the fovea) and operate at a comparatively lower resolution for the remainder (the periphery) (Figure 1.1). Simply, it aims to reconcile three mutually opposing properties: a wide field of view, high visual acuity and few pixels for the downstream vision system to process. Each of these properties can offer distinct advantages for a vision system. A wide field of view lets a system quickly reason about scene information, such as the spatial relations between objects or see very large objects. High visual acuity lets the vision system extract high-frequency visual information from a scene, facilitating tasks such as recognizing very small or very far away objects. Achieving this in as few pixels as possible is highly desirable as most, if not all, computer vision systems have a computational complexity that scales with the number of pixels it must process.

While it is possible to achieve a wide field of view and high visual acuity with uniform sampling (i.e. conventional grid images), foveated vision offers an avenue to realize these same quali-

ties in far fewer pixels, in turn facilitating a far more computationally efficient system. This is particularly important given the current climate of computer vision systems, which are typically based on deep neural networks. Practitioners of deep learning will be well aware of their large compute requirements, typically necessitating (multiple) powerful GPUs for training to produce state-of-the-art systems. This also limits their applicability in environments with constrained computing resources or where rapid inference times are required, like in real-time applications such as robotics. Moreover, the trend of ever larger deep neural network systems raises environmental concerns given the corresponding increases in carbon emissions associated with training and deploying these models [5]. As such, any method that can meaningfully reduce the computational overhead of these systems, particularly ones that do not noticeably diminish performance in any way, are of great interest and can have far-reaching benefits for the future of computer vision.

Despite its promise, foveated vision has seldom been utilized in computer vision, and the conventional uniform approach remains the dominant paradigm. A strong stance would be that a compelling case for adopting foveated vision in the current climate of computer vision is yet to be made. However, a cursory glance at biological vision systems will quickly reveal the prevalence of foveated (or space-variant) strategies, particularly among higher-order animals, including humans and primates. The efficacy of biological vision is undeniable and still outperforms computer vision systems in many facets of visual perception. While the prevalence of foveated vision in biological systems does not guarantee its usefulness in computer vision, it provides compelling motivation for further research. Especially since, much like computer vision systems, biological systems desire to be energy efficient. A simple illustration of how foveated sampling strategies can subserve a more efficient vision system has been made by Schwartz [6], where he estimates that if the entire field of view of the human eye operated at the resolution of the fovea, the human brain would weigh upwards of 5000lbs.

1.3 Background

1.3.1 Biological Vision

To provide the reader with some context for this thesis, we will give a brief overview of biological vision, specifically human vision, from light entering the eye through to the early stages of the visual cortex and some of the underlying processes that are believed to be taking place. This is not a comprehensive and detailed view of all stages, just those that are immediately relevant to the computer vision systems explored in this thesis. Similarly, in this thesis, we do not seek to emulate biological processes in any particular way but rather aim to assess whether the principles behind foveated vision allow us to build better computer vision systems based on deep neural networks.

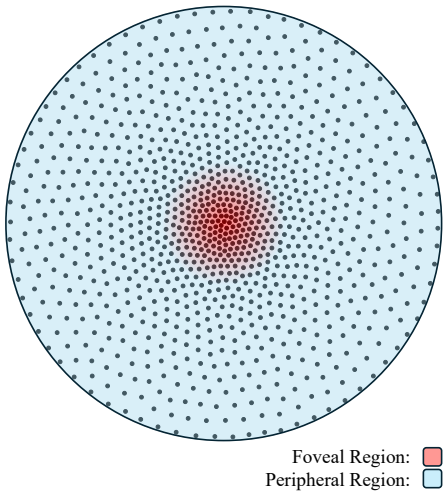

Figure 1.1: A foveated arrangement of 1024 pixels generated by a Self-Similar Neural Network [1] with a fovea radius of 20% of the field of view. The Self-Similar Neural Network yields a locally pseudo-uniform lattice, which presents challenges for neural systems such as convolutional neural networks to process as the data does not lie on a regular grid.

Visual information processing begins in the retina, a sheet of photoreceptors that lines the inside of the eye. The two primary categories of photoreceptors are cone cells and rod cells. Cone cells have three distinct variations, each responding optimally to different wavelengths of light, approximately corresponding to red, green, and blue light. Cone cells provide us with colour vision and decrease in population with eccentricity from the centre of our field of view. The densest region of cone cells is found in the fovea, a small pit in the retina composed entirely of cone cells. This region is responsible for the high visual acuity in the centre of the field of view. Their decreasing population with eccentricity additionally explains why the peripheral regions of our field of view are blurry.

In contrast, rod cells provide little colour information but are far more sensitive to light intensity than cone cells and are responsible for vision in low-light scenarios. Rod cells follow a similar distribution to cone cells but are absent in the fovea . The reader may be aware of their own experiences of being able to see a dim light in the corner of their eye that disappears when viewing it directly. The absence of rod cells allows for a higher density of cone cells in the fovea, making the fovea predominantly responsible for sharp colour vision. For the purposes of this work, rod cells can largely be ignored, with cone cells being a closer analogue to RGB pixels we conventionally think of in digitized images.

The excitation of photoreceptors produces signals that are fed to retinal ganglion cells. These cells come in various forms and have different functionalities. In the fovea, each ganglion cell may receive input from only a single cone, while in the periphery, many cones might feed a single ganglion cell, which, in turn, dictates its receptive field. Much like photoreceptors,

ganglion cells also decrease in population with eccentricity and typically exhibit larger receptive fields with increasing eccentricity. Additionally, there are relatively fewer ganglion cells than cone cells. Ganglion cells can be categorized into many distinct subgroups. Magnocellular cells are responsible for perceiving low-contrast stimuli and rapidly changing stimuli. Parvocellular cells respond primarily to red and green colour and are responsible for high-resolution visual processing. Koniocellular cells similarly provide colour processing, but for blue and yellow colour information [7]. A population of ganglion cells can perform a wide range of functions relating to the extraction of features such as discerning texture, orientation, and motion direction, to name a few.

The optic nerve carries information from the ganglion cells to the visual cortex for further processing. During this process, an interesting transformation takes place. The left and right hemifields are separated and passed to different sides of the brain. The left side receives input from the right hemifields of each eye and vice versa for the right. Furthermore, visual data appears to undergo a geometric transformation. Schwartz [8] provides a model of this transform based on the log-polar transform, dubbed the $\log(z+\alpha)$ transform, which we will discuss further in the next section.

Following this transformation, the signals from ganglion cells are processed by the V1 layer of the visual cortex. Hubel and Weisel's seminal work [9] discovered a range of functionalities that cells in V1 perform, including edge detection, oriented edge detection and grating detection. The general interpretation is that these cells detect low-level features that subserve the extraction of richer features in downstream processing that happen in later stages such as V2 and V4. Similar observations can be made in convolutional neural networks (CNNs) trained on images, which learn edge detectors and grating detectors in early layers that exhibit strong similarities to cells in early visual cortex layers.

Earlier computer vision models such as HMAX [10] and the Neocognitron [11] were heavily inspired by the human visual pathway and sought to emulate certain stages, particularly the early visual cortex layers. These went on to inspire the first convolutional neural network architecture. Despite their biologically inspired origins, subsequent research on CNNs has mostly departed from biological inspiration. Nonetheless, for many CNN architectures, we can still draw loose comparisons between them and biological vision. RGB pixels can be interpreted as responses from cone cells or ganglion cells, and convolutional layers can be interpreted as visual cortex layers. Schrimpf et al. [12] devised "brain score" as a metric to compare activations between neural network architectures and that of human vision. They show that many convolutional-based architectures are quite predictive of the brain activity of humans, suggesting that, to a certain degree, they process visual information in a similar way or operate on similar principles. CNNs are a loose approximation to the visual cortex, nonetheless, an element that is entirely missing is space-variant (or foveated) resolution and the geometric transform of retinal

coordinates to cortical coordinates.

1.3.2 Foveated Vision

Many researchers have looked to foveated vision as a source of inspiration for designing computer vision systems. Over time, this has led to many incarnations of this idea, which, while broadly speaking accomplishing the same underlying goal of sampling visual information at a higher resolution at the centre of the field of view, do differ somewhat dramatically in their design.

The log-polar transform is perhaps the most eminent of all foveated imaging techniques. Schwartz [8] observed that the mapping of retinal information to the cortex follows a log(z+a) formulation in many animal species. As such, the log-polar transform is interesting as a method for computing foveated representations of visual data and due to its biological plausibility. Changing the value for alpha allows for better fits to the exact mapping exhibited in different species to be achieved. This highlights an interesting point: while many species adopt foveated vision or space-variant vision, there is a reasonable degree of inter-species variation, suggesting the optimal strategy could be contingent on each species' ecological niche and other aspects of their morphology such as eye positioning (Figure. 1.2). We can hypothesise that for computer vision systems, there may similarly not be a one-size-fits-all foveated strategy, and instead, the optimal strategy might depend on the architecture, task, and other computational constraints.

While the log(z+a) transform may be more biologically plausible, in general, the use of a pure log-polar transform is often preferred as it is comparatively more straightforward to use due to the discontinuities between visual data in the angular axis being represented at the boundary of the image, and not within the image itself. log-polar methods yield some interesting properties in conjunction with convolution, as global rotation and scale transformations manifest as translations, to which convolution is equivariant. Building a CNN on log-polar images yields a global scale and rotation invariant system (assuming the CNN is translation invariant). This transformation equivariance similarly holds for the log(z+a) method, but only for the peripheral regions. Schwartz's transformation does yield some useful qualities for the representation as, unlike the pure log-polar transformation, it does not oversample visual data in the fovea. This oversampling is due to the asymptotic nature of the log function creating a singularity of sampling kernels [13]. Other workarounds for this problem have been proposed in the form of blindspot log-polar models, in which sampling begins at a fixed radius from the origin, however this leads to the discarding of some visual data at the centre of the field-of-view. Bolduc and Levine [14] propose the use of a separate sampling step that fills the missing data by copying the small missing region from the original image.

As mentioned previously, loose comparisons between convolution and visual cortex layers such

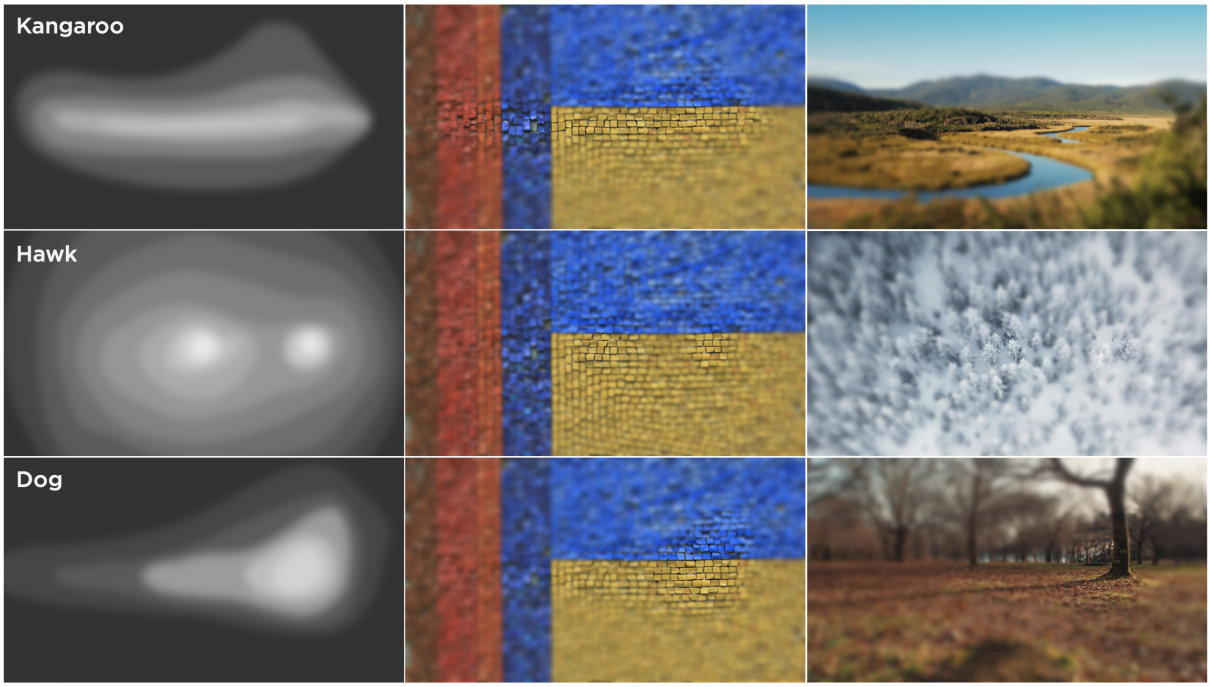

Figure 1.2: Figure reproduced from Malkin et al. [2]. Left: visual acuity across the field of view for different animals. Lighter areas denote higher visual acuity. Middle: An image sampled and displayed with the corresponding resolution of each animal. Right: A representative image of each animal's habitat and how it might be sampled and represented in their visual field. Each animal displays a distinctive space-variant strategy for sampling the visual world.

as V1 and V2 can be made. This prompts the question of whether this log-polar like transform also provides some useful functionality in biological vision. Schwartz argues that this transformation could play some role in perceptual invariances to size and rotation transformations of visual data [8, 15]. However, Cavanagh questions how much utility it can provide given that it is limited to transformations centred at the origin (i.e. centre of the field-of-view) and does not explain perceptual invariances for transformations not at the origin [16]. Cavanagh posits that the transform may simply facilitate a mapping of visual data to a compact physical representation for the visual cortex. Ultimately, it remains unclear how much benefit a global mapping of visual data under this transformation provides. While the application of CNNs to log-polar images has shown greatly improved generalisation to the aforementioned global rotation and scale transformations [17, 18, 19], there is also a noticeable decline in general performance in image classification settings [20, 21].

Cavanagh's hypothesis [16] that the log(z+a) transform may subserve a compact physical representation for foveated visual data does touch on a similar problem that needs to be solved for certain computer vision systems operating on foveated images. Many computer vision operations have been built on the presumption that visual signals exist on a grid domain, which, in a Cartesian coordinate frame, foveated images are not by definition. As such, it is necessary to geometrically transform this data so that it is grid-aligned. A reasonable strategy for building foveated sensors is to perform a grid discretisation of an arbitrary coordinate frame and define a mapping that transforms this grid into a foveated arrangement in image coordinates (i.e. Cartesian coordinates), which then informs the centres of sampling kernels. The transform between a log-polar grid to Cartesian coordinates is one such example of this, however other works have considered sensors that adopt a transform that maintains more Cartesian-like coordinates [21, 22, 23].

These geometric transformations have interesting implications for processing with convolutional layers, as filter weights are shared in this new coordinate frame rather than the conventional Cartesian coordinate frame we think of when processing uniform images. In turn, this affects the equivariance properties of the layer, as it is now equivariant to transformations that manifest as translations in this new coordinate frame. In some cases, the transform is well-principled and provides interpretable functionality that could be beneficial to visual perception, such as in the case of the log-polar transform providing global scale and rotation equivariance. In other cases, it is less clear if there is any inherent advantage to the particular transform. We can hypothesise that in CNN-based systems, the transform should affect task performance due to the change in weight sharing and equivariance properties.

Another popular choice for implementing foveated sampling, particularly for deep learning systems, has been to approximate foveation through a series of crops of increasing field-of-view [24, 25]. This method is closely related to image pyramids and represents foveated images

through multiple sub-images of differing fields of view and resolutions rather than a singular contiguous representation. They are simple to implement and have been utilized in many CNN-based studies [26, 27, 28, 29]. We will refer to this method as the Multi-FoV crop method, and much like the geometric transformation methods, are convenient to use as they represent visual data on uniform grids of pixels. The geometric transformation in this method is limited to rescaling visual data through downsampling, retaining translation equivariance within each representational level for all translations within their field of view.

There are a few methods which do not produce a foveated image with a uniform grid representation. Balasuriya [30] proposed the use of a self-similar neural network [1] to generate a foveated arrangement of points with a locally pseudo-uniform arrangement that serve as sampling kernel centres. Nakada et al. [31] randomly generated a foveated arrangement of points. The aforementioned sensors do not have an obvious solution that will map this data to a uniform grid, posing challenges for their use in conjunction with CNNs. Subsequent works [32, 33] that have built upon Balasuriya's method have made use of the Schwartz' log(z+a) transform and resampled the data to a uniform grid to circumvent this problem. There is potential utility in random or pseudo-uniform arrangements in mitigating aliasing problems such as moire patterns [34]. It is unclear whether such aliasing problems are meaningfully problematic for deep neural networks. Lukanov et al. [21] demonstrated that adopting a pseudo-uniform arrangement did not improve the classification accuracy of their system for image classification problems.

1.3.3 Active Vision

Active vision, coined by Aloimonos [35], refers to a subclass of computer vision systems that can control the geometry of the sensor or camera. Unlike passive vision systems, which process visual data as provided to the system, active vision systems can intelligently sample new visual data. Such a strategy offers distinct advantages for visual perception that can be predominantly understood from the perspective that a visual sensor is typically constrained. As such, a single observation of a visual scene might not elaborate all possible useful information about a scene. By actively adjusting the sensor, multiple scene observations can be collected, processed and integrated into a singular percept. Aloimonos showed that several problems related to fundamental visual perception mechanisms, such as shape from texture and structure from motion, are ill-posed for a passive vision system but well-posed for active systems.

Foveated vision systems naturally align with active vision systems due to the space-variant nature of their resolution. There is a need to fixate (align the foveated sensor's gaze) on salient parts of a scene to make adequate use of the high-resolution fovea. In biological systems, this occurs through eye movements (saccades). In humans, this happens an average of three times a second and is influenced by top-down and bottom-up attention processes. Furthermore, there is the possibility of integrating information from these different observations into a unified per-

cept. Ballard [36] describes systems with this behaviour as having low-resolution (in terms of the number of pixels they process) but a high virtual resolution. More generally, Tsotsos et al. [37] argued that fixating on different regions of a scene with a uniform sensor should be helpful in a vision system that processes visual information locally and in a hierarchical fashion as pixels in the centre of the field of view contribute more to the output of the system. CNNs are an example of such a system, and recent works have demonstrated this bias towards the centre of the field-of-view [38, 39]. While active vision is a necessary component to adopting foveated vision, this work will predominantly focus on the latter. For the purposes of this thesis, we will use "active vision" to describe any system that can control its sensor geometry to influence how visual information is sampled (e.g. translating the sensor). We will use fixation to refer to the position in the image that is at the centre of the sensor's field of view, and the ability to fixate as the ability to translate the sensor to a fixation location.

1.4 Research Gap

Foveated vision has been researched for several decades but ultimately remains a fringe idea within the computer vision community. As such, many facets of it remain understudied. Below, we outline several research gaps within the literature.

While a variety of sensors have been evaluated in the context of deep learning systems [20, 21, 26, 27, 28, 32], there lacks a systematic comparison between them. It is hard to quantify whether different methods have meaningful differences regarding task performance as they are individually evaluated on different tasks and with different architectures. Torabian et al. [20] and Lukanov et al. [21] provide some studies in this area and show that the sensor choice can have a significant impact on performance. We aim to expand on these works to provide clearer evidence on how the choice of sensor impacts the system's performance and highlight, in particular, how the geometric transformations applied to foveated data play a role in its overall efficacy.

Another aspect of foveated vision that is seldom explored is how sensors should be parameterized. Many proposed sensors can be parameterized (for example changing the size of the fovea) allowing control over the sampling resolution across the visual field. Evidence from biology shows that different species have different space-variant resolution strategies [4], suggesting that the optimal design is not general for all tasks and systems but dependent on these constraints. To the best of our knowledge, no works have attempted to characterize sensor parameterisation and how it affects task performance, nor have they attempted to understand any causal factors that may help us pick better parameterisations without resorting to manual tuning.

The majority of deep learning research adopting foveated vision has done so using a CNN backbone [20, 21, 27, 32]. Recently, non-convolutional architectures such as vision transformers have been shown to be comparable or even outperform CNNs across a variety of vision tasks .

Relatively few studies [40] have explored the use of foveated sensing; however, we will make the case that these architectures are highly suitable for such data, while foveated vision is similarly complementary to them in mitigating their large compute requirements.

From a utilitarian point of view, the motivating principles of foveated sampling could similarly be realized by other biologically implausible methods, such as those that adapt the sampling layout of the sensor on a per instance basis [41, 42, 43, 44]. It is essential to contrast these methods as well as uniform methods to understand where foveated vision stands in the general field of computer vision. However, such comparisons are yet to be made.

In this thesis, we aim to address the aforementioned research gaps in the foveated vision literature by building and empirically evaluating a variety of foveated vision architectures based on deep neural networks. While foveated vision could be applied in various subfields of computer vision, for this thesis, we will use object recognition on both natural image datasets and synthetic datasets as a representative performance indicator. Accordingly, we outline the following research questions that this thesis aims to answer.

1.5 Research Questions

In this section, we outline several research questions that guide the research conducted in this thesis and that we aim to answer in the subsequent chapters.

1. *To what degree does the geometric transformation of foveated data to a grid-aligned representation play a role in the accuracy of foveated convolutional neural networks?*

   We show that the change in coordinate frames imposed by the geometric transformations required to map foveated data to a grid-aligned representation plays a large role in the accuracy of foveated CNNs. Additionally, we propose a graph convolutional approach to processing foveated images that allows for arbitrary coordinate frames to be used, yielding more accurate foveated CNNs.

2. *Can foveated computer vision systems outperform uniform systems in object recognition for a given pixel budget?*

   We show that foveated object recognition systems can outperform their uniform counterparts in terms of classification accuracy for convolutional and non-convolutional systems. We further show that this performance gap increases as the scale distribution of objects in the dataset increases.

3. *Does adopting foveated vision in non-convolutional architectures such as vision transformers and mixer architectures lead to better classification accuracy than a uniform sensing approach?*

Recently, non-convolutional architectures such as vision-transformers [45] have emerged as powerful alternatives to CNNs. We argue that these architectures are a natural fit for operating on foveated images, as they have fewer image-specific inductive priors and can perform space-variant computation. We show that such networks exhibit improved performance when using a foveated sampling strategy.

4. *Is a foveated sampling strategy more useful for image classification than biologically implausible methods such as learning-to-zoom [44] in terms of classification accuracy?*

   We show that for convolutional systems, biologically implausible methods slightly outperform foveated methods. However, they are approximately on par with each other when used in conjunction with non-convolutional systems such as ResMLP [46], or Vision Transformers [45].

5. *Is the optimal parameterisation of a foveated sensor (in terms of classification accuracy of the system) dependent on the architecture, data or both?*

   Using a foveated sensor in which the sampling resolution and radius can be controlled, we sweep over a range of parameterisations and show that architectures that operate on a lower number of total pixels favour smaller, higher resolution fovea and vice-versa. Furthermore, we show that the scale variation of objects in a dataset plays an important role in shaping the optimal sampling layout for foveated sensors.

6. *Can we learn an optimal parameterisation of a sensor jointly with deep neural network weights through backpropagation?*

   Optimizing the sensor layout through hyperparameter tuning is costly for deep neural networks. We introduce a novel sensor that allows the sampling layout to be learned via backpropagation. We show that the sensor converges on a similar layout to the optimal one found via hyperparameter search.

7. *For sequential active vision architectures, does integrating visual information from multiple observations improve classification accuracy and what mechanisms are needed to achieve this?*

   We show that foveated object recognition systems benefit from repeated observations of images. We find that a simple averaging mechanism suffices in image classification over relatively more complex methods such as self-attention or recurrent networks.

1.5.1 Thesis Structure

The structure of this thesis is as follows:

- Chapter 2. introduces a graph convolutional approach to processing foveated images and presents a pilot study comparing classification accuracy when representing foveated visual data in different coordinate frames. This work is presented in part in *Monte-Carlo Convolution on Foveated Images* [47]
- Chapter 3. expands upon the pilot study conducted in the previous chapter and introduces a fixation mechanism to guide the sensor to salient regions of images. Additional variations of the graph convolution are also explored. This work is presented in part in *Foveation in the Era of Deep Learning* [48]
- Chapter 4. looks towards non-convolutional vision architectures, such as vision transformers [45] and ResMLP [46], as candidates for processing foveated images. Additionally, a differentiable parameterized foveated sensor is proposed which allows for sampling resolution over the visual field to be optimized via backpropagation.
- Chapter 5. investigates foveated vision architectures that repeatedly attend to visual data in a sequential manner. In particular, different methods for integrating visual information over time are compared as well as utilizing recurrent processing in all hidden layers.
- Chapter 6. presents a summary of the findings of the work conducted in this thesis, and several avenues for future work.

Chapter 2

Convolution on Non-Grid Structured Visual Data

2.1 Introduction

Convolutional neural networks (CNNs) are a prominent class of vision models that have been applied effectively to various vision tasks, including recognition, segmentation and detection. Unsurprisingly, much recent prior work concerned with foveated computer vision systems has also adopted CNNs for the downstream processing of visual data [20, 21, 27, 32]. An emerging trend in these prior works is that adopting a foveated strategy over a uniform one does not yield significant improvements to task performance (i.e. classification accuracy). This is best illustrated in a study from Torabian et al. [20], which shows that foveated CNNs exhibit lower classification accuracy than uniform CNNs on the ImageNet-1k dataset [49].

Two reasonable hypotheses can be formulated to explain this trend. Firstly, foveated vision provides no functional benefit over uniform vision in completing the visual perception tasks covered in this thesis and prior works (i.e. image classification on natural images), and is an overarching research question guiding this thesis (RQ.2). Secondly, the application of convolutional neural networks to foveated image data as conducted in prior works, is sub-optimal, diminishing task performance. This hypothesis is the subject of this chapter, as to reasonably answer Research Question 1, it is necessary to seek or devise suitable models for processing foveated data.

In this chapter [1], we attempt to identify any characteristics of foveated CNNs that may indicate pathologies within the system that hamper performance. In particular, we look at the fundamental principles of CNNs (i.e. filtering with shared weights) and their implications for operating on foveated images. We do this through the lens of geometric deep learning [50, 51], a frame-

[1] The work in this chapter has been presented, in part, in *"Monte Carlo Convolution on Foveated Images"* [47].

work for understanding machine learning systems in terms of how they utilize the underlying geometric structure of the data on which they operate.

Through this analysis, we identify the geometric transform of foveated data to a grid-aligned representation amenable for convolutional processing as a potentially problematic trait of previous foveated CNN designs. These transforms are necessary because CNNs require data to exist in a grid-structured domain. However, these transforms impose specific coordinate frames for representing visual data and change the behaviour of convolution layers in potentially harmful ways.

To this end, we propose a novel graph convolutional approach to processing foveated images. Crucially, it allows foveated image data to be represented in arbitrary coordinate frames as it does not necessitate visual data to lie on a grid. We present a set of pilot experiments exploring different coordinate frames for representing foveated visual data free from other compounding factors, such as the specific pixel arrangement of different foveated sensors. We show that the coordinate frame does indeed play an important role in the overall efficacy of a foveated CNN in classifying natural images.

2.1.1 Chapter Structure

This chapter is structured as follows:

1. Firstly, we introduce the relevant background knowledge required to understand the work conducted in this chapter.
2. Secondly, we introduce relevant prior works presented in the literature.
3. The methodology behind the graph convolution approach and the overall model architecture is outlined.
4. We then outline pilot experiments and the corresponding results and analysis.
5. Finally, we provide a discussion of the work conducted in this chapter and its limitations.

2.2 Background

2.2.1 Inductive Biases

A prominent theme within this chapter is inductive bias, particularly from equivariance (and invariance) to transformations of the input data. We can view building a machine learning model for a given problem as searching a function space for a function that allows us to make predictions that generalize to unseen data. Inductive biases play a role in biasing or constraining this search space to functions that we expect to perform well, simplifying this procedure, and potentially providing some guarantees to how the system behaves.

Inductive biases in machine learning models come in various forms. One example is the use of a linear or polynomial model for regression tasks. The former constrains the function space to linear ones, while polynomials consider a broader class of functions that can be characterized as polynomials up to some maximum order. Using these models as examples, we can comment on both the advantageous and inhibitory role of inductive biases. Considering a hypothetical regression task in which there is a highly non-linear relationship between input variables and target variables, the use of a linear model results in an inductive bias that constrains the function space to functions that are insufficient to characterize this data correctly, and as such a polynomial model may be favourable due to its ability to represent non-linear functions. Conversely, suppose a linear function characterizes the data well, the function space of polynomial models is much broader and have the potential to overfit to spurious features in the training data (e.g. noise), leading to solutions that do not generalize well to unseen data. In this case, the linear inductive bias is helpful as the optimal solution lies within the space of linear functions.

2.2.2 Equivariance and Invariance

Convolution exhibits one of the most well-known inductive biases in deep learning models, namely translation equivariance, which is a product of filtering with shared weights. Formally, a function f is equivariant to some transformation g of the input x if $f(g(x)) = g(f(x))$. In the context of convolution, it is equivariant to translations of the input image. I.e. translating an image and then applying convolution is equivalent to applying convolution to an image and then translating. We can visualize this in the form of a commutative diagram (Figure 2.1).

A related concept is that of invariance. While in the case of equivariance, the transformation of the input manifests as a corresponding transformation of the output, a function f that is invariant to a transformation g produces the same output for all g in a transformation group. Formally, we can define this as $f(x) = f(g(x))$.

The translation equivariance inductive bias of convolution layers means features can be detected in images regardless of their absolute position. Equivariance over invariance is essential in this

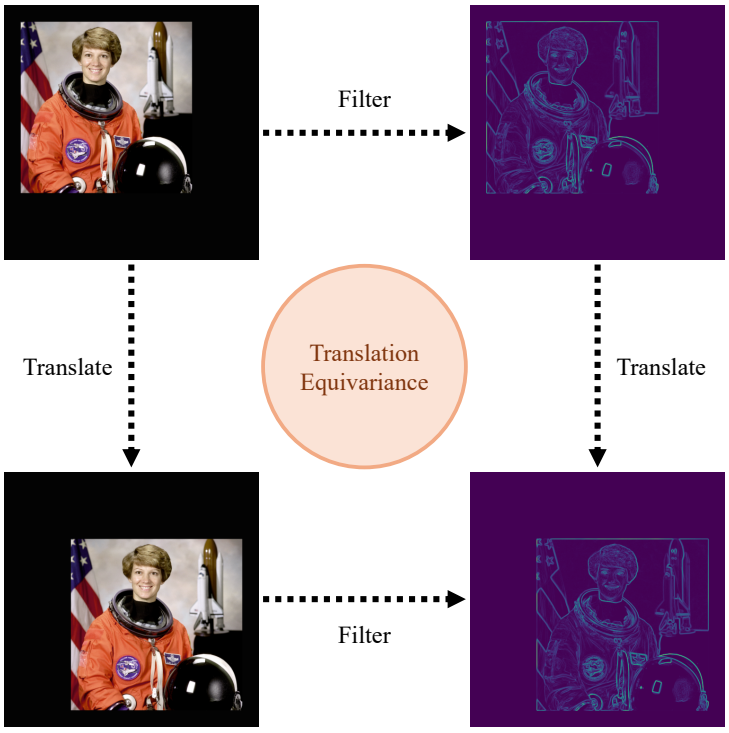

Figure 2.1: A commutative diagram visualizing the equivariance of convolution layers with respect to transformations from the translation group.

case, as we would like to preserve spatial relationships between features over multiple layers, which would be lost if convolution was invariant to translation. In the context of image classification datasets, it is often assumed that the translation group (for Cartesian images) is a symmetry of the data and does not change the label. We can say that the label of an image is invariant to translation of the image. Convolution layers can be used with global pooling layers to build a translation invariant CNN, which has become the de-facto design in state-of-the-art CNNs. This incorporates a useful inductive bias into the architecture, as invariance to translations no longer has to be learned, as we have constrained the function space to translation invariant functions.

2.2.3 Geometric Deep Learning

Convolution layers operate on grid-structured domains. In the context of images, the signal and the domain come from the discretisation of a visual signal, which conventionally uses a Cartesian coordinate frame. Convolution leverages the grid structure of the domain to infer spatial relationships between different parts of the signals, which allows for spatial features to be extracted from said signals. Furthermore, grids have translation symmetry. Convolution layers exploit this symmetry via filter weight sharing, yielding translation equivariant filtering of signals on the domain.

Notably, translation has different meanings depending on the grid-structure's coordinate frame. For example, if a Cartesian coordinate frame is used, translations correspond to translations in

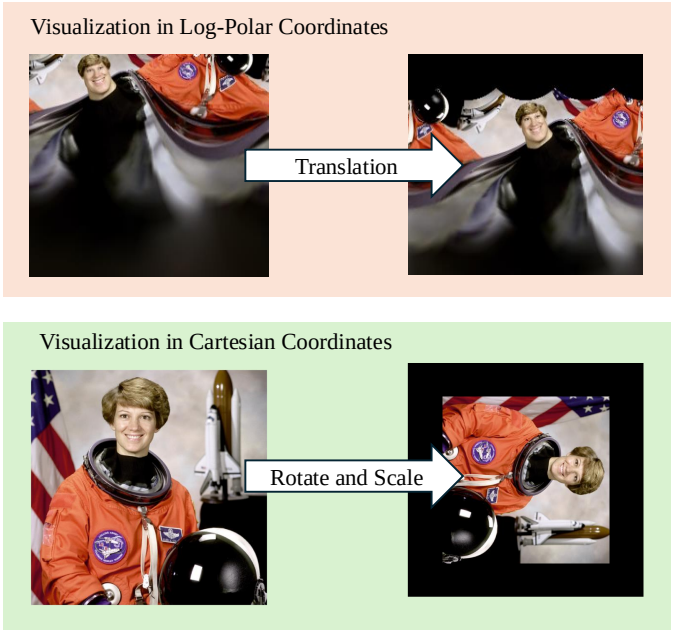

Figure 2.2: Visualisation of the same transformation in different coordinate frames. In log-polar coordinates, translating visual data corresponds to rotating and scaling visual data, centred at the origin. Hence, translation equivariant functions applied to log-polar images are equivariant to Cartesian rotation and scaling.

the Cartesian sense. If a log-polar coordinate frame is used, translations correspond to rotation and scaling centred at the origin (Figure 2.2). This is why CNNs applied to log-polar images are rotation and scale invariant, and translation equivariant when applied to conventional images.

In order to apply conventional CNNs to foveated images, the data must be represented on a grid-structured domain. A foveated sampling arrangement must satisfy two qualities: a foveated arrangement in Cartesian coordinates and a grid-structured arrangement in another coordinate frame. We can interpret this as a geometric transform of Cartesian foveated image data to another coordinate frame, and therefore, this imposes a change to the transformation to which convolution is equivariant.

As discussed previously, inductive biases can be helpful but also inhibitory. Equivariance to a transformation of a visual signal is helpful if the transformation is a symmetry of the data, i.e., a transformation that does not change the task's relevant semantics. We argue that the necessary geometric transform of foveated image data to a grid-structure does not necessarily yield equivariance properties that are helpful for image classification and could explain why foveated CNNs seldom outperform uniform CNNs. A simple illustration of this is a log-polar CNN applied to MNIST. The digits 6 and 9 are rotations of each other, so a rotation invariant CNN will be unable to distinguish between these digits.

2.3 Related Work

2.3.1 Foveated Convolutional Neural Networks

There have been several attempts to incorporate foveated vision into CNNs. Log-polar preprocessing has been utilized in several works, motivated from the perspective of achieving global scale and rotation equivariance, implementing foveated vision, or both. [17, 18, 52] apply log-polar preprocessing as input to a CNN that performs image classification. In particular, they show that such networks implicitly generalize to global rotation and scaling transformations much better than CNNs applied to Cartesian images. Kim et al. [17] showed the importance of wrap-around padding in addressing the discontinuity of the visual representation in the angular axis and, in turn, improving classification accuracy. An important observation can be made from Kim et al's work in that this improved performance only holds for log-polar networks trained on non-rotated data. When both Cartesian networks and log-polar networks have rotation data augmentation applied, Cartesian approaches outperform log-polar methods. Esteves et al. [19] propose Polar Transformer Networks (PTN), an active vision system based on a reformulation of Spatial Transformer Networks (STN) [43]. PTNs use a separate localisation network to inform a classifier where to look, giving the system the ability to intelligently fixate on an input image. They aim to use the log-polar transform to solve rotation and scale transformations, and fixation to solve translation transformations. As such the network can choose the origin of the log-polar transform and achieve a jointly rotation, scale and translation invariant system. Dabane et al. [53] similarly use a log-polar transform in a Spatial Transformer Network, but in the localisation network and not the classifier. They show that the log-polar transform helps the network localise objects in the image.

The aforementioned works predominantly focused on relatively simple datasets such as MNIST [54] and CIFAR10 [55]. Torabian et al. [20] and Lukanov et al. [21] conduct experiments on ImageNet-1k [49] with fixation mechanisms based on DeepGaze [56, 57, 58] and a heuristic based method using class activation maps [59] respectively. They compare the log-polar method [13] to uniform approaches. Additionally they compare against Cartesian like foveated approaches. Lukanov et al. make use of the Cartesian Foveal Geometry sensor proposed by Martinez et al [22] while Torabian et al. [20] make use of strong barrel distortion. In both works improved classification accuracy was seen with foveated approaches that were more Cartesian like. Ozimek et al. [32] conduct a similar study, but without fixations, adopting a GPU accelerated version of Balasuriya's [30] software retina and Schwartz' log(z+a) transform [8, 15]. They compare against a uniform strategy, and resampling the foveated image back to a uniform image. It was found that the log(z+a) method performed the worst, and there was a noticeable performance increase by simply resampling the foveated image back to a uniform grid. In such a scenario, the amount of visual information remains the same, suggesting that the reduced per-

formance was not due to the data compression, but rather the way the data was represented and provided to the CNN.

The Multi-FoV approach mentioned previously and originally proposed by [24, 25], has been popular approach for implementing foveated vision for CNNs [26, 27, 29, 60]. These systems function broadly in the same way, but differ slightly in the details. For example, Karpathy et al. [26] make use of two CNNs, one for processing the foveal crop, and one for processing the peripheral crop. Sermanet et al [27] and Li et al [60] make use of a similar method, but share weights between the different streams. Sharing weights might have intuitive relations to some form of scale equivariance and the similarities of this sensor and image pyramids, while allowing weights to be decoupled might allow the network to learn more nuanced computations for the fovea and periphery. Karpathy et al. [26] do show that the foveal and peripheral convolution layers do learn distinctly different filters suggesting that decoupling the weights might be a good idea.

Multi-FoV methods, or indeed any sensor that represents visual data as discontiguous sub-images such as the sensor proposed in [14], also introduce discontinuities in the visual representation, as high-resolution foveal data is processed separately to the low resolution peripheral data. This limits the ability of foveal and peripheral features to interact until they are unified into a single representation. This unification is typically performed very late in the network where features are typically high-level, however it may be beneficial to allow lower-level foveal and peripheral features to interact. Architectures such as Feature-Pyramid Networks [61] offer potential avenues to address this, but introduce additional complexity and parameters. Beyond this there is also the non-obvious choices of how many crops to use, what their respective fields-of-view should be, and how much each should be downsampled. Although we do not explore this method in great detail in this work, these uncertainties are indicative of the nascent stages of research into foveated vision and a lack of guiding principles on how best to build such systems.

An important detail to highlight from these works is the continual empirical evidence that the coordinate frame used to represent the foveated image seems to have a significant impact on the accuracy of the CNN in image classification problems. The coordinate frame stems from the geometric mapping required to map the foveated data to a uniform grid, and is an integral part of allowing convolutional neural networks to operate on certain foveated images. In this chapter, we attempt to verify that the coordinate frame of the foveated representation is indeed a contributing factor to this difference in performance. We achieve this through a graph convolutional based approach, which allows us to change the coordinate frame of convolution arbitrarily as it does not require foveated images to represented as a uniform grid. It should be noted that this commentary is in the context of object recognition. For other visual perception tasks such as motion estimation [62] and robotic hand-eye coordination tasks, log-polar preprocessing for CNNs has shown to be beneficial. Providing commentary on these ideas for a general vision

system that performs a wide-repertoire of perception tasks is beyond the scope of this work.

2.3.2 Deep Learning on Non-Grid Structured Domains

As discussed previously, CNNs operate on grid-structured domains. In order to explore alternative coordinate frames for convolution, we must relax the requirement for grid-structured domains, and seek a more general form of convolution to non-grid structured data. A natural way to view non-grid-structured visual data is a graph, as grids are simply a special case of graphs with a specific geometric structure [50, 51]. In turn this suggests graph convolution as a viable candidate for addressing the aforementioned concerns. Owing to their generality, graphs emerge in a variety of different contexts, however we can focus the discussion to situations where nodes in the graph (in our case pixels) can be associated with spatial coordinates, such as graph deep learning on point clouds [63] and chemical molecules [64].

PointNet [65] stands as a major milestone in neural network approaches to processing point clouds without resorting to voxelisation. The approach predominantly makes use of multi-Layer perceptrons (MLPs) to embed each point independently of one another into a higher-dimensional space before aggregating these embeddings into a single representation by max-pooling over the spatial dimension. PointNet can be viewed as a particular case of a graph neural network with an empty edge set. This is distinctly different from CNNs which capture local patterns in a hierarchical fashion. In the geometric deep learning framework [50], this local and hierarchical design pattern is referred to as the scale separation prior and has been a common property of state-of-the-art CNNs, and many other architectures, since their inception.

PointNet++ [66] addresses this by applying PointNet locally and hierarchically improving performance on Point Cloud classification and segmentation. In the context of image classification, Li et al. [67] demonstrate that PointNet++ is unable to perform better than random guessing on CIFAR10 suggesting it has limited application to image understanding. PointNets have inspired a number of successor architectures that aimed to more faithfully introduce the strength of convolution to point cloud processing. These architectures incorporate edge information and can be more readily described as spatial graph convolution. Simonovsky and Komodakis [68] propose a graph convolution layer for processing point clouds and general graphs with edge conditioned filters. These filters map the spatial offsets between points through an MLP to filter weights. Many similar works based on edge conditioned filters have been proposed [63, 69, 70, 71, 72]. Largely they are consistent in mapping spatial offsets between points to filter weights through an MLP, however Hermosilla et al. [70] and Wu et al. [71] make the connection to Monte-Carlo integration and introduce a weighting term for each point based on the inverse probability density function of each point to de-bias the filter approximation to oversampled regions, improving robustness to non-uniform sampling. Additionally, it is shown that on application to grid data this formulation recovers the standard discrete convolution operation. Other related architectures

achieve similar functionality but do not directly use an MLP to produce filter weights. KPConv [73] interpolates point defined filter weights to match the local sampling pattern. PAConv[74] applies a small neural network to the spatial offsets of points to softly select weights from a weight bank. PointCNN [67] learns a non-linear transformation to permute and weight a local set of points in an attempt to enforce a canonical representation for arbitrary arrangements of local points.

A few works considered graph convolution processing for foveated images using predefined filters. Wallace et al. [75] propose a graph based approach for arbitrary space-variant sensors, and in particular those of the log(z + a) family [8, 15, 76] to address the discontinuities in the visual representation in the angular axis. They demonstrate that certain filtering operations can be achieved such as pseudo-laplacian filtering. Our approach builds upon this work by introducing a parameterised convolution filter able to perform filtering with arbitrary filter functions. In a similar vein, Balasuriya and Siebert uses a graph convolution method to filter foveated images [77]. This method uses Gabor filters due to their resemblance to neurons in the visual cortex [78, 79, 80]. Again, we extend upon this work by reformulating it as a parameterised graph convolution layer able to learn arbitrary filters.

2.4 Graph Convolution on Foveated Images

In this section we describe our graph convolutional approach to processing foveated images. The convolution layers in CNNs typically use filters with a limited spatial support, meaning that output neurons are locally connected to a set of input neurons based on their spatial proximity. It is easy to define these local connections on grid structured domains without ever taking into consideration explicit spatial position information. Instead one can rely on each neuron's index within the data structure (e.g. a tensor) to implicitly define the spatial relationships between neurons. The regular sampling of grid structured domains ensures that these implicit spatial relationships between connected input and output neurons are consistent for all output neurons.

For some foveated sensors, such as those with pseudo-uniform pixel arrangements [1, 81], there are not obvious indexing rules that can be used to define local connections limiting the application of indexing based methods to define spatial locality. For the purpose of a general form of graph convolution that can accommodate images with irregular domain structures, we require a graph construction method that can define local receptive fields on an arbitrary arrangement of points in a 2D plane.

2.4.1 Graph Construction

Our convolution layer takes as input a set of feature vectors, each associated with a vertex $u \in U$ of a 2D planar graph and a position (x,y) that defines the feature's spatial position within the feature map. For example, U could be a foveated image, where u is a pixel with an RGB value and its Cartesian coordinate. Similarly, we define a set of vertices V pertaining to output features, each with an (x,y) coordinate and a feature vector $\hat{f}$ that we aim to compute. We define local connections as the edges E from input neurons U, to output neurons V, to form a bipartite graph $G = (U,V,E)$. Note that it is not strictly necessary for the size of U and V to be the same. Decreasing the number of vertices in V relative to U allows for downsampling like operations to be implemented.

The edges E are determined by finding the K-Nearest Neighbours in U for each $v \in V$, based on the nodes' spatial coordinates (x,y) and a Euclidean distance metric. K in this case is analogous to the filter size of the layer. The receptive field of an output neuron v (i.e. the vertices in U that connect to $v \in V$) is given as $R(v) = \{u \in U | (u,v) \in E\}$ (Figure 2.3). It is possible to use locality measures other than K-Nearest Neighbours, such as fixed-radius near neighbours. In such cases $|R(v)|$ can vary for each $v \in V$, whereas with K-Nearest Neighbours $|R(v)| = K$.

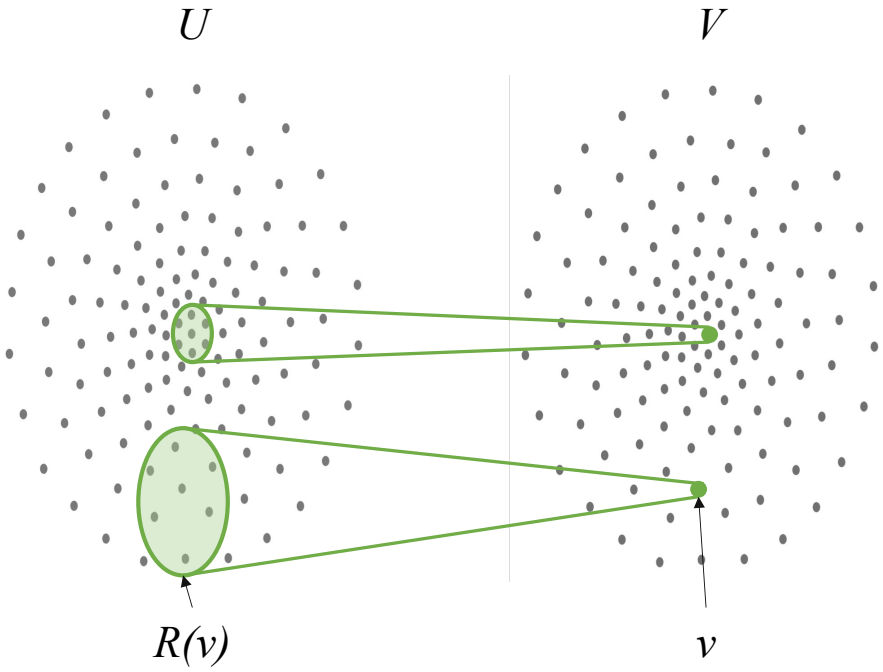

Figure 2.3: Visualisation of the local spatial connections between vertices U and output vertices V with edges removed for visualisation purposes. Using K-nearest neighbours leads to a fixed number of elements for each receptive field $R(v)$, which scales the spatial support of the receptive field as sampling rate decreases.

2.4.2 Edge Conditioned Filters

While the edges E tell us how input neurons are connected to output neurons, it does not say how we can implement weight sharing. Recall that we are trying to compute the feature vector $\hat{f}_v$ for all $v \in V$ which will be a weighted sum of the features f_u associated with $u \in R(v)$, referred to as f_{uv} from now on for brevity. A naive implementation would be to define a weight vector $w \in \mathbb{R}^K$ where w_u denotes the weight applied to f_{uv}. The feature vector f_v is computed as:

$$\hat{f}_v = \sum_{u \in R(v)} w_u \cdot f_{uv} \tag{2.1}$$

This method is problematic as it is an asymmetric function and thus not equivariant to the permutation of its variables. This makes it difficult to apply this method to non-uniformly sampled data, where $R(v)$ might lack a canonical ordering. Moreover, the coordinates associated with $u \in R(v)$ might not be consistent for different $v \in V$. This limits the applicability of using a single discrete set of weights to compute all the output features.

To address this we leverage edge conditioned convolutional filters [68], which have been successfully applied to convolutional processing of point clouds [63, 69, 70, 71, 72]. Rather than

apply a discrete set of weights, we compute filter weights through a learnable parametesized function $G(\cdot)$ applied to edge labels. We define the label δ_{uv} for edge $(u,v) \in E$ as the spatial offset between vertices u and v, normalized by the average spatial offset for a receptive field $R(v)$.

$$\delta_{uv} = \frac{x_u - x_v}{\sum_{u \in R(v)} x_u - x_v} \tag{2.2}$$

Where x_u and x_v are the x coordinates for the vertices $u \in U$ and $v \in V$ respectively. We have simplified to the 1D case for simplicity. This allows us to compute $\hat{f}_v$ as a symmetric function invariant to the permutation of input features:

$$\hat{f}_v = \sum_{u \in R(v)} G(\delta_{uv}) \cdot f_{uv} \tag{2.3}$$

By conditioning filter weights on relative spatial offsets, we allow the graph convolution operator to model relative spatial relationships much like ordinary convolution operators. Normalizing the spatial offsets has the effect of scaling the filter function as sampling density becomes sparser (i.e. in the periphery). Different methods for labelling edges will yield different equivariance properties within the convolution layer. This allows the graph convolution method to be highly general. For example, we could use log-polar coordinates to label the edges, or both Cartesian and log-polar.

2.4.3 Motivation for Coordinate Frame and Filter Scaling

While our graph convolution layer allows for arbitrary coordinate frames to be used, it does not help us decide which one to actually use. In this work, we have adopted a Cartesian coordinate frame and rescaling the offsets so that the filter function scales as resolution decreases. This design is motivated in the following ways.

Firstly, the choice of Cartesian coordinates is motivated due to several empirical results in prior works showing that log-polar foveated images generally exhibit lower classification accuracy than Cartesian-like foveated images [20, 21, 32]. The term Cartesian-like is somewhat vague. In the context of this thesis, we are specifically referring to coordinate frames where moving in a positive direction in the y axis, always manifests as a positive movement in the y axis in Cartesian coordinates, but potentially with an additional x component. Moving in a positive direction in the x axis has corresponding behaviour. In other words, the notions of up down left and right are largely consistent with respect to Cartesian coordinates.

Secondly, the choice of scaling the filter can be motivated from several angles. Without scaling,

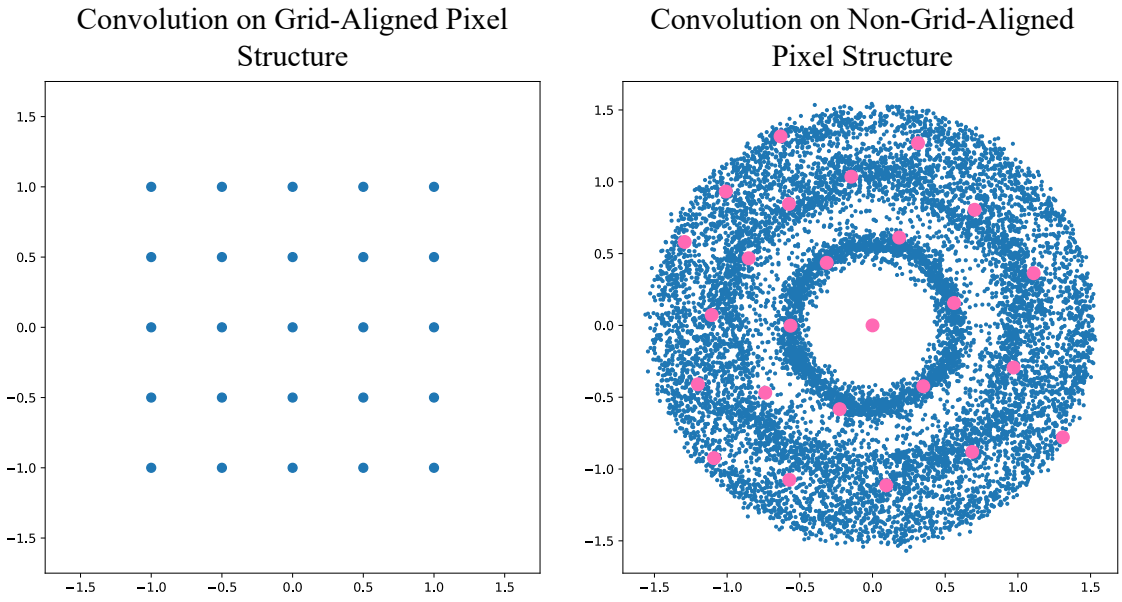

Figure 2.4: Visualisation of the spatial offsets of pixels in a receptive field, relative to the centre of the receptive field for grid-structured images and non-grid structured images. On grid-aligned structures, the offsets are a discrete set of (x, y) coordinates, that are consistent for all receptive fields. On non-grid-structures, the offsets for a receptive field can be viewed as being drawn from a distribution of (x,y) coordinates. We visualize the offsets for all receptive fields in blue, and the offsets for a single receptive field in pink.

the area of the filter is consistent across the field of view. Tiezzi et al. [82] refer to this as the principle of uniform coverage. This incurs a risk of undersampling the filter in sparse peripheral regions, leading to poor filter response approximation and aliasing. This could be alleviated by enforcing large low frequency filters. However, this does not leverage the high-resolution fovea, as we only extract low frequency features from this region.

Tiezzi et al. [82] propose to address this problem by low passing filters in sparser regions. This approach could be implemented in our graph convolution layer. However, for the purpose of this thesis we aim to use a purely convolutional approach without the use of additional specialized behaviour, beyond our graph convolution, to understand the effect of coordinate frames in isolation. Finally, scaling is also consistent with the receptive fields of neurons in the visual cortex, which scale with eccentricity.

2.4.4 Implicit Neural Filters

Unlike uniform images, there is the possibility for the edge labels δ on foveated images to take on any real values rather than a small finite set of discrete values. Conceptually it is easiest to understand the domain of δ as continuous (Figure 2.4) and thus necessitates that our filter function G is also continuous over the spatial domain. Note that to compute a given $\hat{f}_v$, we

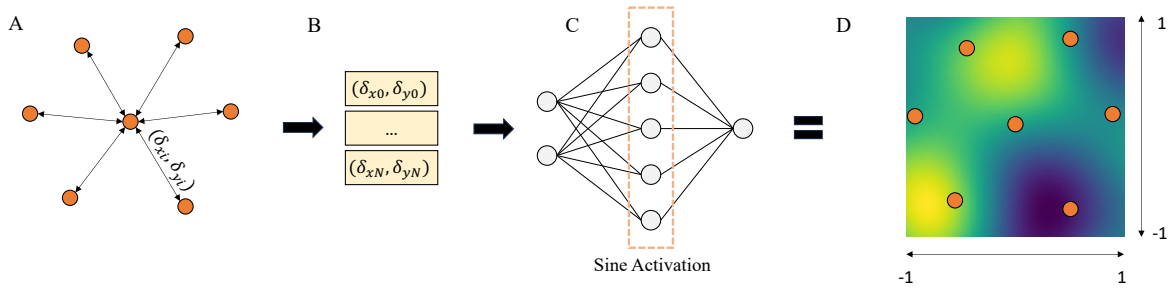

Figure 2.5: Visualisation of the process of generating edge conditioned filter weights for an arbitrary arrangement of pixels. A) shows the spatial positions of pixels where $(\delta_{xi}, \delta_{yi})$ is the normalized spatial offset between the i^{th} pixel and the centre of the receptive field. B) Spatial offsets are computed for all pixels in the receptive field. C) an MLP maps each spatial offset to a filter value. D) Visualisation of the underlying filter function represented by an MLP. Orange points indicate where the function is evaluated.

only evaluate G using the edge labels δ_{uv} for $u \in R(v)$. This bears resemblance to Monte-Carlo integration techniques where we know we can approximate the convolution of two continuous functions by drawing a finite number of samples for each.

Previous works have utilized multi-layer perceptrons (MLPs) with ReLU activation as continuous function approximators for edge conditioned filters [68, 70, 71]. Here we make an additional connection between edge conditioned filters and implicit neural representations [83, 84, 85] to inform a more principled design for the MLPs used in our convolution layer. Briefly, the aim of implicit neural representations is to parameterise a neural network such that it maps the domain of a signal to the value of the signal. It has been shown that ReLU activated MLPs operating on low dimensional inputs are biased to learning low frequency functions [86]. In the context of edge conditioned filters this bias may limit the expressivity of the convolution layer and its ability to extract high frequency information from feature maps. Work from Mildenhall et al. [84] and Sitzmann et al. [83] has shown that periodic activations or Fourier projections can better equip an MLP to model high frequency functions over low dimensional domains. Accordingly, we opt for a single hidden layer sine activated MLP for our function G. Similarly to Sitzmann et al. we also apply a scalar multiplier ω before the activation functions which is a hyperparameter to be tuned. A higher ω allows for higher frequency sine activations at initialisation.

2.4.5 Foveated Sampling

In order to produce a foveated image we resample a uniform image using Balsuriya's proposed method [81]. This requires a set of sampling locations that serve as the centres for Gaussian receptive fields. Briefly, we generate a foveated sampling layout (e.g. a log-polar grid) to serve as receptive field locations. The size of each receptive field is given by the mean distance to its 4

closest receptive fields. The sigma of the Gaussian is parameterised as $\frac{1}{3}$ of this distance. These Gaussians are then realized as kernels to sample a uniform input image.

2.4.6 Architecture

For all CNNs we use an isometric (sometimes referred to as isotropic) ConvNeXt [87]. They consist of an initial convolution stage with a filter size of 16 (or 4×4 on grid-structured) and downsample the input image by a factor of 4. we use 8 identical ConvNeXt blocks, each consisting of a depth-wise convolution with a kernel size of 9 (equivalent to 3x3 on a grid), followed by a non-linear feed-forward network that operates in a pointwise manner across the spatial dimension. Finally, we apply global average pooling to the final convolution features and use a single fully connected layer to make class predictions. In the case of application to Log-polar images, we apply circular padding on the angular axis to address the discontinuity at the $0-2\pi$ boundary.

For all graph convolutional layers we use a single hidden layer MLP with 16 hidden units and Sine Activation unless stated otherwise. The spatial dimensionality of the feature-maps is the same for all ConvNeXt blocks which simplifies the implementation of skip connections and removes the need for downsampling operations to be considered. Despite their simplicity, isometric architectures have demonstrated competitive with the more conventional hierarchical CNN architectures [88, 89].

2.5 Image Classification on Imagewoof

In these experiments we aim to answer three main questions. Firstly, how sensitive is classification accuracy to the coordinate frame used when applying convolution to foveated images? Secondly, can we achieve better classification accuracy with a graph convolutional approach to processing foveated images (rather than conventional convolution)? Finally, can foveated CNNs outperform a uniform CNN in classification accuracy?

We conduct our experiments on Imagewoof [90], a 10-class fine-grained subset of ImageNet-1k [49]. Our motivation for this choice of dataset is firstly that it exhibits reasonably large image sizes, suitable for applying foveated downsampling. Secondly, it has a sufficient number of samples per class, allowing for stable training of each model while still being small in total size (9,025 training images), allowing for quick iterations in these early stages of development. Finally, we speculate its fine-grained nature means it can't rely as heavily on low-resolution cues such as the general shape of objects.

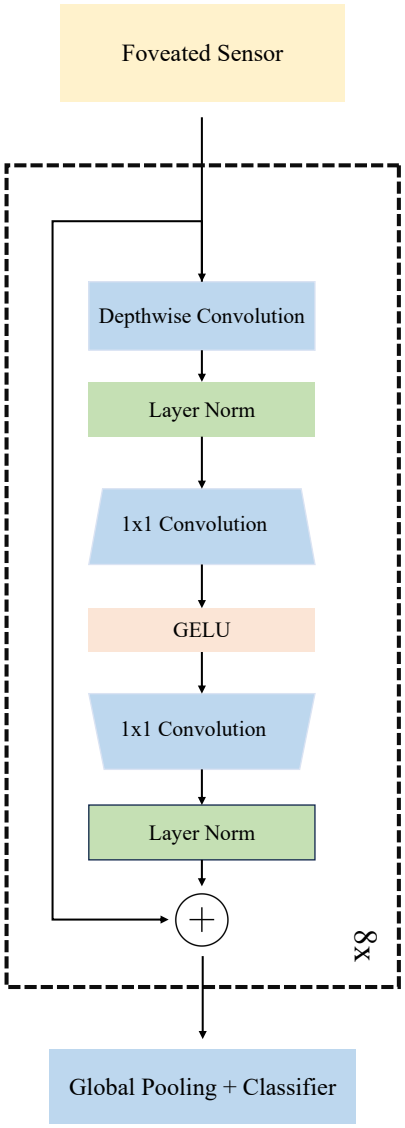

Figure 2.6: Schematic overview of the Isometric Foveated Graph ConvNeXt.

2.5.1 Training Details and Hyperparameters

We consider three methods for generating a foveated sampling layout. A log-polar grid of 80×160 resolution. A blindspot log-polar grid, with 80×160 resolution, but with a fovea radius of 11 pixels chosen through random hyperparameter search. (i.e sampling begins from a radius of 11 pixels from the origin). A self-similar neural network (SSNN) [1], with a 112^2 resolution and a fovea radius of 30% of the field of view, chosen through random hyperparameter search.

For training, we use the AdamW[91, 92] optimizer with a batch size of 256 and train the network for 90 epochs. We use a linear warm-up learning rate scheduler for 5 epochs followed by 85 epochs of cosine decay where the maximum learning rate throughout the schedule is 0.003. We use a weight decay of 0.01 applied to all parameters except the bias terms.

We use a data augmentation procedure of resizing the input image so its shortest side is 256px and perform a random resized crop of 224x224px and a scale distribution of (0.08 to 1.0). We additionally use the TrivialAugment [93] data augmentation policy which randomly applies an additional augmentation to the cropped image. At test time, we resize the image so its shortest side is 256px and perform of centre crop of 224×224px. All input images to the network are normalized by the Imagenet [49] mean and standard deviation.

We train all networks with gradient descent using the cross-entropy loss and use label smoothing of 0.1. Hyperparameter selection was carried out via random search using a validation set constructed from 10% of the training data. The final reported accuracy is on a held out test set using the model checkpoint with the highest validation accuracy over the course of training.

2.5.2 Results

Does Coordinate Frame Matter? : We present results for all methods in Table 2.1. We show that for both log-polar sensor variants, applying Graph Convolution in a Cartesian coordinate frame improves classification accuracy relative to using a log-polar coordinate frame (71.3% vs 66.4% and 76.6% vs 72.0%). An independent t-test shows both of these results are statistically significant, p=0.0006 and p=0.0006. Importantly, the only difference between these methods is the coordinate frame convolution is applied, in verifying that these observations are dependent on this variable (RQ 1).

It is challenging to ascertain the exact underlying reason as to why one coordinate frame is better than another. In the context of image data of the natural world, one potential reason is gravity likely imposes a strong bias in the distribution of rotations. As a result, the rotation equivariance derived from log-polar coordinate frames might be of limited use in datasets such as images. Another reasonable hypothesis is that Cartesian-like coordinate frames more closely resemble translation equivariance, and are useful in cases when fixations can't be used to address

Sensor	Resolution	Convolution Coordinate Frame	Conv Method	G	Accuracy (%)
Uniform	112×112	Cartesian	Conv	-	81.4 ± 0.4
Uniform	112×112	Cartesian	Graph Conv	MLP (Sine)	82.3 ± 0.8
Log-Polar	80×160	Log-Polar	Conv	-	66.6 ± 1.4
Log-Polar	80×160	Log-Polar	Graph Conv	MLP (Sine)	66.4 ± 0.3
Log-Polar	80×160	Cartesian	Graph Conv	MLP (Sine)	71.3 ± 0.8
Log-Polar (B.S)	80×160	Log-Polar	Conv	-	72.0 ± 0.7
Log-Polar (B.S)	80×160	Log-Polar	Graph Conv	MLP (Sine)	72.0 ± 0.2
Log-Polar (B.S)	80×160	Cartesian	Graph Conv	MLP (Sine)	76.6 ± 0.4
SSNN [1]	112^2	Cartesian	Graph Conv	MLP (Sine)	79.0 ± 0.6
SSNN [1]	112^2	Cartesian	Graph Conv	MLP (ReLU)	67.9 ± 1.8

Table 2.1: Classification accuracy on Imagewoof using different sensors and convolution methods. Log-polar (B.S) refers to the log-polar blindspot method for generating foveated images. Resolution refers to the number of pixels in the sensor. G refers to the learnable function that maps edge labels to filter values in the graph convolution. All results are reported on a held-out test set, averaged over three independent training runs from different random seeds.

translation. In the following chapter we introduce a fixation mechanism to the system to better verify these findings.

Are Foveated Graph CNNs Viable? : While our graph convolution showed that coordinate frame has an effect on classification accuracy, it does not guarantee that graph convolution, even when utilizing a Cartesian-like coordinate frame, outperforms a standard convolutional approach. We conduct experiments showing that our graph convolution approach, applied to log-polar sensors, improves performance relative to a standard convolutional approach (+ 4.7% for the log-polar sensor, and + 4.6% for the blindspot log-polar sensor). Additionally, we show that our graph convolution can be applied effectively to sensor layouts generated by a self-similar neural network. Owing to its randomly generated layout, applying CNNs to this pixel structure usually requires resampling to a grid, as performed by Ozimek et al. [32]. We show that our graph convolution can be applied directly to this data, without resampling, and also yields the best performing foveated CNN across all methods.

This performance improvement can be attributed to both the coordinate frame and the use of a sine activated MLP to represent graph filters. We show that ReLU activated MLPs exhibit significantly worse performance on the SSNN sensor (79.0% vs 67.9%). We also observe that our graph convolution achieves the same performance as standard convolution when both using the same coordinate frames, suggesting they are approximately as expressive as each other. In the next section we conduct further experiments that support these observations.

Can Foveated CNNs outperform Uniform CNNs? : Despite showing that our graph convolution layer can make significant improvements to the classification accuracy of foveated CNNs,

we still find that none are able to outperform a standard CNN operating on uniform images (RQ2). It is likely that in order to fully leverage foveated sensing, a fixation mechanism is necessary and may explain why no method could outperform a uniform approach in these experiments. In the following chapter we introduce a fixation mechanism to the system to test this claim and provide more complete answers to Research Questions 1 and 2. Nonetheless, the graph convolution layer presented in this chapter provides a strong foundation for future experiments.

2.5.3 Effect of Depth, Hidden Size and Omega Hyperparameters on Classification Accuracy

In this section we look at how the parameterisation of the MLP used for computing filter weights affects classification accuracy. Specifically, we look at the effect of activation functions, the ω pre-activation scaling hyperparameter for Sine activated MLPs, the hidden size of the MLP and the depth. All experiments are performed on Imagewoof with the SSNN sensor and the same architecture and hyperparameters as used in the main experiments.

MLP Hidden Size: Figure 2.7 shows the performance for both ReLU and Sine activated MLPs for over different hidden sizes. We show that Sine activation significantly outperforms ReLU activation by at least 8% in all cases. We find in the case of ReLU MLPs, scaling the hidden size does not yield significant performance increases. Beyond 36 nodes the computational overhead becomes increasingly prohibitive during training. Sine activated MLPs show a small boost of $\sim 2\%$ when increasing the hidden size from 9 to 18, after which performance plateaus. This provides compelling evidence that Sine activated MLPs learn filters that facilitate the extraction of more discriminative features relative to ReLU. We also show that Sine activated MLPs only need a relatively small hidden size to reach a good accuracy, after which there are diminishing returns.

Omega Scaling Parameter: Motivated by Sitzmann et al. [83] we introduce the ω hyperparameter which applies a scalar multiplier to the pre-activations. This has the effect of increasing the frequencies of the Sine waves by some factor ω, in which higher values may make it easier for the MLP to learn higher frequency functions. We sweep over a range of ω values for a 9 hidden layer MLP (Figure 2.8) and find that values of 4-6 are optimal for our network. Performance is particularly sensitive to this parameter, for example in the case of $\omega = 1$ and $\omega = 10$, we see a 6% and 3% drop in performance respectively.

MLP Depth: Finally, we probe the effect of increasing the number of hidden layers in the MLP (Figure 2.9). In all models we use a hidden size 9 and for the Sine activated model we use an ω value of 6. We find that increasing the depth leads to slight performance increases for both MLP variants. The effect is more pronounced in ReLU activated MLPs which is likely due to

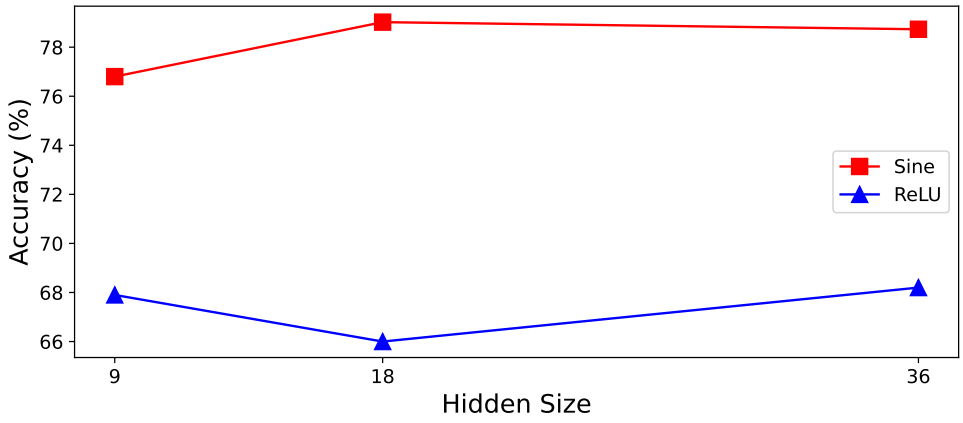

Figure 2.7: The effect of increasing the hidden size for both Sine and ReLU activated MLPs on classification accuracy on Imagewoof. Larger sizes allow for more expressive functions to be modeled, however we show scaling the size does not yield significant performance gains. This suggests that small MLPs should suffice to convolutional filters.

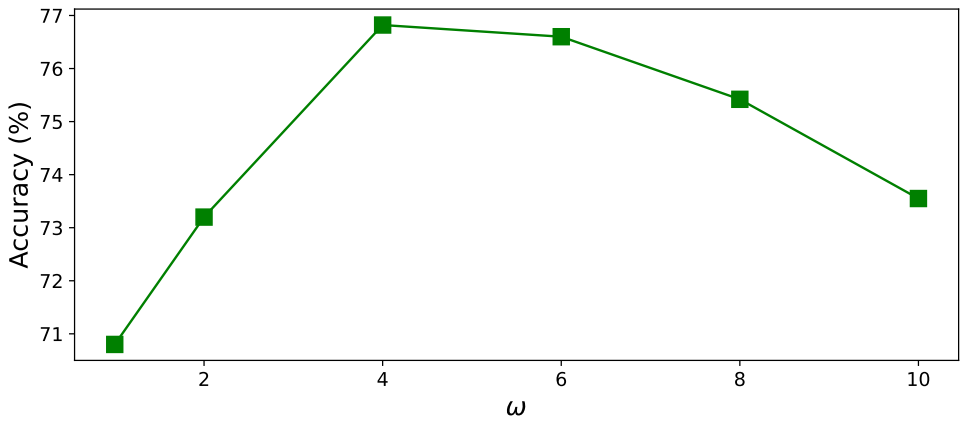

Figure 2.8: The effect of ω, the pre-activation scaling in Sine activated MLPs, on classification accuracy on Imagewoof. We consider an MLP of hidden size 9 and 1 hidden layer in these experiments.

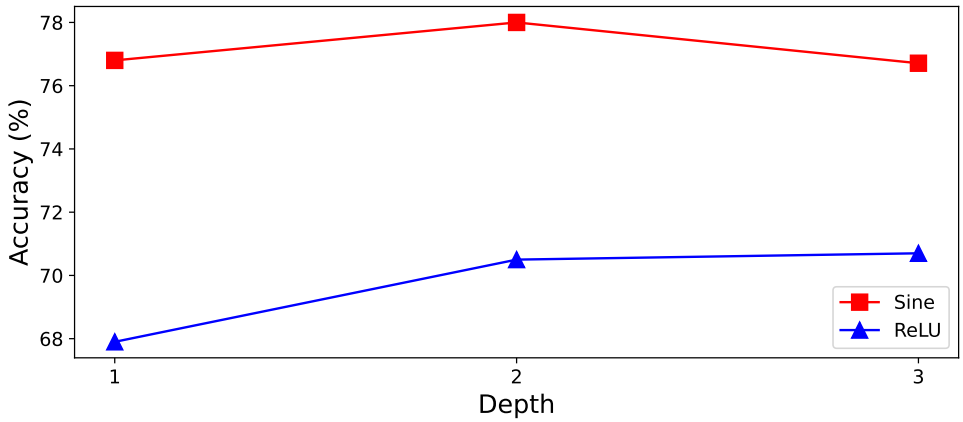

Figure 2.9: The effect of increasing MLP depth on classification accuracy on Imagewoof. Much like scaling the hidden size, increasing the depth of the MLP only marginally improves performance

the additional depth allowing the MLP to model more complex and expressive filter functions. The additional depth does not contribute to a significant performance increase for Sine activated MLPs and even degrades performance when 3 hidden layers are used.

2.6 Limitations

The primary limitation of the work conducted in this chapter is the absence of a fixation mechanism. We circumvented this important functionality by leveraging camera bias in image classification datasets, which typically means objects of interest occupy a sizeable percentage of the full image and are centred in frame, and has been employed as a strategy in other works [26, 32]. Nonetheless, this is a loose approximation at best and likely diminishes the performance of foveated systems. Furthermore, it is possible that the suitability of different coordinate frames changes under the presence of a fixation mechanism, as translation transformations can be partially addressed through translating the sensor. This necessitates the introduction of a fixation mechanism to answer Research Question 1 more completely.

A limitation of the graph convolution method is its memory overhead. Unlike, convolution on a grid, a filter is represented by a unique set of coefficients at each location, as the spatial relationships between pixels is not necessarily the same for two different receptive fields, despite being derived from the same underlying function. This entails an O(n) memory complexity where n is the number of "pixels" in the input, compared to O(1) on grid-structured images. This limitation can be problematic in layers with many filters or when processing many pixels. This overhead is significantly mitigated when adopting depth-wise separable convolution layers [94], which are commonplace in SOTA CNNs [87, 95]. It is also possible to avoid storing the full weight tensor in memory, and instead computing filter weights as part of inference such as

in [70], however this incurs extra computation at inference time.

A final limitation of this study is that only a few sensors and corresponding coordinate frames were compared. We showed that a Cartesian-like coordinate frame was favourable, of which there are several sensors that exhibit this characteristic [22, 23, 25]. It is important to evaluate these sensors, not only to support or refute the benefit of Cartesian-Like representations, but also to ascertain whether a graph convolutional method is strictly necessary.

2.7 Conclusion

In this chapter we looked at the processing of foveated images using convolutional neural networks. We made an observation from the empirical results of prior works that the adoption of a foveated strategy seldom improved the performance of CNNs over a uniform strategy (often diminishing performance) in image classification tasks on natural image datasets. We hypothesized that this trend could be explained by CNNs, in their conventional form, being ill-suited to processing foveated images.

We analysed the fundamental properties of convolution and the ramifications of its application to foveated image data through the lens of geometric deep learning. We identified the geometric transformation of foveated data to a grid-aligned representation as a potential pathology in the application of CNNs to foveated images, that may hamper performance in image classification tasks and is the subject of Research Question 1. Specifically, we speculated that the geometric transform of foveated images to a grid-aligned representations may often impose a sub-optimal coordinate frame in which to apply convolutional processing to visual data, as convolution may no longer capture a useful transformation of visual data that we would like the network to be invariant to.

In order to test this claim we proposed a graph convolutional approach to processing foveated images. This layer generalizes the notion of Convolution to non-grid structured image domains and allows for arbitrary coordinate frames for representing visual data to be used. We use this layer to build a foveated CNN and perform a series of image classification experiments on Imagewoof, contrasting different coordinate frames in absence of other compounding variables. Empirical analysis showed that classification accuracy significantly improved when adopting a Cartesian-like coordinate frame rather than a log-polar coordinate frame. This indicates that the geometric transform of foveated data to a grid-aligned representation can indeed yield a sub-optimal coordinate frame for convolutional processing of foveated data.

Beyond this, our graph convolution layer improves upon previous (Edge-conditioned) graph convolution layers [68, 70, 71] in the context of our work. We drew insights from the field of implicit neural representations to derive a more principled method for representing spatial filters

on a graph. These improvements allow our graph convolution to be a useful operator for building foveated CNNs in its own right, and not just as an exploratory tool for analysing the impact of different coordinate frames.

Our overarching research question (RQ2) asks whether Foveated CNNs can outperform Uniform CNNs in image classification tasks on natural images. We found this not to be the case in this chapter. Importantly, the models presented in this Chapter lack a crucial functionality, the ability to fixate intelligently. In the following chapter we extend this line of work, incorporating a fixation mechanism into the system and address some of the aforementioned limitations.

Chapter 3

Foveated Convolutional Neural Networks

3.1 Introduction

In the previous chapter we looked at the application of convolutional neural networks to foveated images in the context of image classification. We proposed a novel graph convolution layer for processing foveated images and showed that coordinate frame used to represent foveated image data can have a large effect on the classification accuracy of the network. In this chapter,[1] we aim to provide more definitive answers for Research Questions 1 and 2 by addressing the limitations of work conducted in the previous chapter.

We extend upon the previous line of work by introducing a fixation mechanism into the system to center the sensor on salient portions of the input image. In particular, we take inspiration from Spatial Transformer [43] Networks and Polar Transformer Networks [19]. The network leverages differentiable image sampling [43, 96] that allows for end-to-end training of the fixation mechanism using only class labels as supervision.

We compare against a wider array of foveated sensors, in particular those that adopt a Cartesian-Like coordinate frame for representing foveated image data on a grid, as well as biologically implausible methods such as Learning-to-Zoom [44] and Spatial Transformers [43] (Research Question 4). We conduct our main experiments on ImageNet100, a 100 class subset of ImageNet-1k [49]. Our main findings corroborate those of the previous chapter, demonstrating the importance of coordinate frames when applying convolutional processing to foveated images. We show that our foveated graph CNN significantly outperforms all other foveated CNNs, and all other non-foveated methods (including a uniform CNN baseline) with the exception of Learning-to-Zoom [44], promoting the use of graph convolution layers for processing foveated images.

We also make several additional contributions in this chapter. Firstly, we propose a novel

[1] The work in this chapter has been presented, in part, in "Foveation in the era of Deep Learning [48])

foveated sensor which allows for the parameterisation of the radius and sampling resolution of the fovea. We conduct experiments sweeping over a range of different fovea radii, to elucidate the sensitivity of classification accuracy to the sampling layout of the sensor (Research Question 5). Furthermore, several alternative methods for representing spatial filters on graphs are explored for our graph convolutional layer, specifically as linear combinations of known basis functions such as Gaussian Derivatives [97].

3.1.1 Chapter Structure

This chapter is structured as follows:

- Firstly, a general overview of how the ability to fixate is introduced to the foveated CNN described in the previous Chapter is outlined.
- The differentiable GPU accelerated image sampling method is then described along with our newly proposed foveated sensor.
- We then describe the various instantiations for representing spatial filters on a graph as linear combinations of known basis functions.
- Our main experiments comparing different foveated CNNs and non-foveated CNNs are detailed along with results and discussions.
- Experiments pertaining to fovea radius hyperparameters and different methods for representing graph filters is then reported.
- We conclude with a summary of the work undertaken in this chapter, its main findings and its limitations.

3.2 Architecture Overview

Before we outline specific details of the system, we will first provide a high-level overview of the architecture and how it processes visual data. In general, active vision systems are typically thought of as sequential models that process visual data and then make an action to sample new visual data. However, this chapter is primarily concerned with feature extraction from foveated images, not active vision. To evaluate the network's ability to extract diagnostic features from an image, it suffices to consider an architecture that produces a single fixation for a foveated classifier. This allows us to abstract away additional mechanisms, such as memory, from the system. We consider a simple two stage architecture that first localises the object which in turn informs a foveated classifier where to center its gaze. This architecture resembles existing architectures such as region proposal networks and spatial transformers [43, 98].

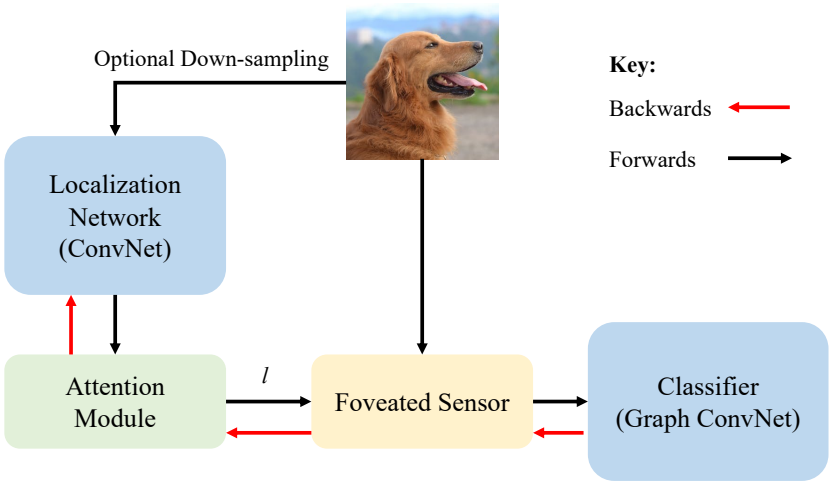

Figure 3.1: Visualisation of the forwards and backwards pass of the foveated CNN with fixations. The localisation network computes a single channel feature map which the attention module uses to compute an (x, y) coordinate where the foveated sensor should centre its gaze. Differentiable sampling is used to backpropagate through the foveated sampling stage to optimize the localisation network using only gradients from the cross-entropy loss.

Specifically, given a uniform resolution input image, a localisation network consisting of a CNN computes a fixation coordinate (x, y). Note that the localisation network does not operate on foveated images. The foveated sensor centres its gaze at this coordinate and samples the full resolution image to produce a foveated image. The foveated image is then fed to a convolutional classifier which outputs class predictions. Both the localisation network and the classifier are trained jointly in an end-to-end fashion with gradient descent by leveraging differentiable image sampling [43, 96]. This also allows the localisation network to be trained without direct supervision of where objects are (Figure 3.1).

3.3 Localisation Network

In order to fully exploit foveated vision, the sensor needs to be aligned with salient objects to maximize the amount of useful visual information they provide to the classifier. We use a localisation network consisting of a CNN and an attention module. Given an RGB input image $I \in \mathbb{R}^{H \times W \times 3}$, where H and W correspond to the height and width respectively, the CNN feature extractor $G(\cdot)$ computes a single channel latent representation $\hat{I} \in \mathbb{R}^{H' \times W' \times C} = G(I)$, where H' and W' are the height and width of the feature map. It is not strictly necessary to adhere to any particular architecture for this feature extractor, however for the purposes of the attention module it should help for the features to retain some sort of retinotopic organisation. That is to

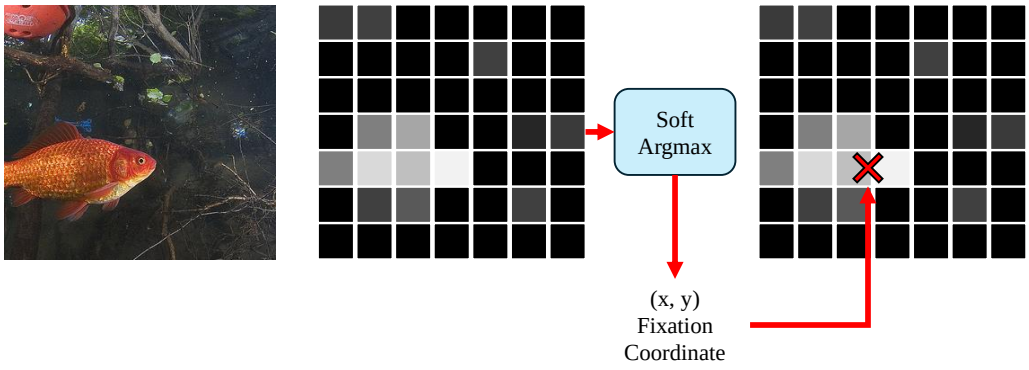

Figure 3.2: Visualisation of the attention module. Left: The input image. Middle: The 1 channel saliency map computed by the localisation network. Right: visualisation of the fixation coordinate computed by the soft argmax operation respective to the saliency map.

say, the spatial positions of features in the output feature map should correlate to their spatial position within the input image. CNNs satisfy this requirement and have the added benefit of translation equivariance making it easy for the localisation network to produce correct fixation coordinates for on object, irrespective of its position in the image.

Exploiting the fact that CNNs tend to learn saliency map like feature representations [99, 100, 101] a simple mechanism for computing fixations would be to take the argmax of the feature maps. This method has the benefit of avoiding regressing the fixation coordinates directly with a fully connected network, which are notoriously difficult to train and frequently collapse [102], however it is non differentiable and prevents training of the localisation network directly from the classification loss. Instead, we apply a spatial softmax activation and associate each neuron $i \in \hat{I}$ with a coordinate (x_i, y_i) which describes each neurons location respective to the original image I. A fixation coordinate l (simplified to one dimensional coordinates for brevity) can then be computed as:

$$l = \sum_{i \in \hat{I}} \text{Softmax}(\hat{I})_i x_i \tag{3.1}$$

This is a soft formulation for producing a fixation coordinate where the value of the feature map is at its greatest, allowing for backpropagation through this module (Figure 3.2). A similar method has been described in [19], which computes the centroid of the feature map which is also differentiable. We initially chose Softmax over the centroid method to explore different temperatures, as well as Gumbel-Softmax [103], and how these affected learning fixation behaviours. In practice we did not find any particular benefit over using the default temperature scaling of 1.0, and similarly found Gumbel-Softmax to be less stable during training. While we maintain the use of Softmax in our module, we do not claim or attempt to verify if it is better or worse than the centroid method described by Esteves et al. [19].

3.4 Differentiable GPU Accelerated Sampling

In this section we outline a simple differentiable GPU accelerated method based on PyTorch [104] for performing foveated sampling with a variety of different strategies. Simply, our sensor S requires a set of Cartesian coordinates, referred to as a tessellation $T \in \mathbb{R}^2$, to serve as the centres of bi-linear kernels. Adjusting the gaze of the sensor is achieved through adding a spatial offset to the coordinates in T, which in this architecture is the fixation coordinate l from the localisation network. The foveated image is given as $F = S(I|T,l)$. Importantly, as shown in [43, 96] it is possible compute $\frac{\delta F}{\delta l}$ through the accumulation of the gradients of each sampling kernels output with respect to l. This permits the learning of l through backpropagation, in turn allowing the localisation network to be trained through the classification loss.

PyTorch provides a grid sample function that can conveniently implement this and handle the backwards pass as well as batch processing. While we use bi-linear kernels, it is possible to use other sampling kernels provided that it is possible to compute $\frac{\delta F}{\delta l}$. Gaussian kernels are a well motivated choice both from a signal processing perspective and due to their connections to biological vision and can fulfil this requirement. The implementation of Gaussian kernels in the pytorch framework becomes more challenging as the spatial support of the kernels varies over the visual field, necessitating a different number of coefficients for each kernel. It is possible to realize this idea in a somewhat efficient way using functionalities such as scatter [2], however we found there was a noticeable overhead. Pilot studies were conducted on classification tasks comparing bilinear kernels and Gaussian kernels both for foveated and uniform sampling, however there was not a noticeable difference in performance. As such, we acknowledge that Gaussian kernels are favorable from a signal processing perspective however for the purpose of this thesis we maintain the use of bilinear kernels due to their convenience.

3.5 Sunflower Foveated Sensor

While not essential to this architecture, we propose a new foveated sampling method that has some convenient properties. Predominantly, this is motivated by the fact that self-similar neural networks (SSNNs) iteratively converge on a sampling layout and this process can be slow. While this sampling layout performed best in our previous chapter, it is inconvenient to use when exploring different hyperparameters such as fovea radius as the iterative procedure has to be run for the new hyperparameters.

Our proposed foveated sampling layout is based on the Fibonacci (or Sunflower) spiral [3] which provides an easy method for creating the densest possible packing of congruent circles in a circle. Of course, in this case we only consider points that serve as the positions of sampling

[2] https://pytorch.org/docs/stable/generated/torch.scatter.html
[3] https://physics.nyu.edu/grierlab/fibonacci3b/node2.html

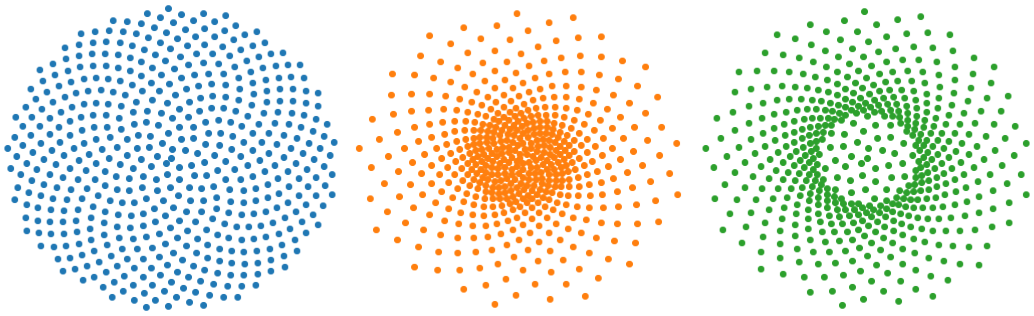

Figure 3.3: Left to Right: Vogel's model of a sunflower capitulum [3], our foveated adaptation (eq. 3.2) where fovea sampling density is well parameterised, and our adaptation where the fovea sampling density is poorly parameterised.

kernels. By default, this method gives rise to approximately uniformly distributed points within a circle, however we need the density of points to vary over the visual field such that it is greatest at the centre and decreases with eccentricity. We achieve this by logarithmically spacing points outside a given radius. Formally the position of the i^{th} sampling kernel in polar coordinates (ρ, θ) is given by:

$$\theta_i = 2\pi i\phi, \qquad \lambda = r^{\frac{1}{d-N}}, \qquad \rho_i = \begin{cases} r\sqrt{\frac{i}{d}}, & \text{if } i < d-1. \\ r\lambda^{i-d}, & \text{otherwise.} \end{cases} \tag{3.2}$$

Where ϕ is the golden ratio, N is the total number of sampling kernels, d is the number of sampling kernels in the fovea, and r is the radius of the fovea.

The sampling density of the fovea can be controlled independently from the size of the fovea; however, setting the number of sampling kernels in the fovea too low will result in an excessively sparse fovea. This is undesirable as the sampling resolution will not be able to resolve details in the fovea (Figure 3.3). In practice, we do not tune this parameter in any of our experiments, instead we search for a value of d such that the sampling resolution in the fovea is approximately the same as that immediately outside the fovea, resulting in a smooth transition between the fovea and the periphery. It is easy to extend the design of our sensor to achieve different sampling densities across the visual field by defining a function $\xi(i)$ to map i to some other value for ρ_i. In the next chapter we explore learning the function $\xi(i)$.

This sampling layout shares many similarities with those produced by a SSNN[1]. The fovea radius can be controlled through a hyperparameter and there is a smooth transition between foveal and peripheral resolution. Similarly, the peripheral resolution decays logarithmically with eccentricity. However, unlike the SSNN, this runs in negligible time making it convenient for experiments involving sampling layout hyperparameters.

3.6 Basis Filters

The original formulation of our graph convolution used a Sine activated MLP to represent a learnable function G that maps edge labels to filter weights. Filter weights are learned as a linear combination of the penultimate activations of the MLP. A parallel can be drawn between this method and parameterising filters as a linear combinations of known basis filters. In the case of the Sine activated MLP the penultimate hidden neurons correspond to a learned basis of Sinusoidal waves where the incoming weights and bias of a neuron correspond to frequency and phase respectively. This shares similarities to a Fourier basis. Accordingly, we derive a new formulation for G which is a linear combination of known basis functions.

There are of course many possible basis spaces that could be used to construct arbitrary filters. Previous works have considered Discrete Cosines [105], Fourier-Bessel functions [106], Gabor Filters [107], Circular Harmonics [108] and Gaussian Derivatives [97] as candidates for learning filters from atomic basis functions.

Predominantly, our aim is to see whether the use of suitable basis filters can achieve comparable or better performance than the Sine activated MLP method. There are two benefits of a basis approach. Firstly, the basis filters can be computed at initialisation, reducing the amount of computation during a forward and backward passes during training, in comparison to an MLP. Secondly, it is possible to leverage certain functionalities from different types of basis. For example, many basis functions such as Gaussian Derivatives, Circular Harmonics, and Fourier Bessel functions, are steerable and can be used to build rotation equivariant convolution layers [108, 109].

For this work, we adopt Gaussian Derivative basis functions for our main experiments. The basis filters can be computed through the multiplication of a Gaussian windowing function G and the Hermite polynomials H where $H_m(x)$ computes the m^{th} order partial derivative along the x axis. Accordingly, a 2-D Gaussian Derivative basis filter can be computed as follows:

$$B(x,y,\sigma,m) = (-1)^{m_x m_y} H_{m_x}\left(\frac{x}{\sigma\sqrt{2}}\right) H_{m_y}\left(\frac{y}{\sigma\sqrt{2}}\right) G(x,y,\sigma) \tag{3.3}$$

Where x and y are the coordinates at which the filter is evaluated, σ controls the size of the filter, and $m = (m_x, m_y)$ defines the order of the partial derivatives in the x and y directions respectively. The Hermite polynomials are defined as:

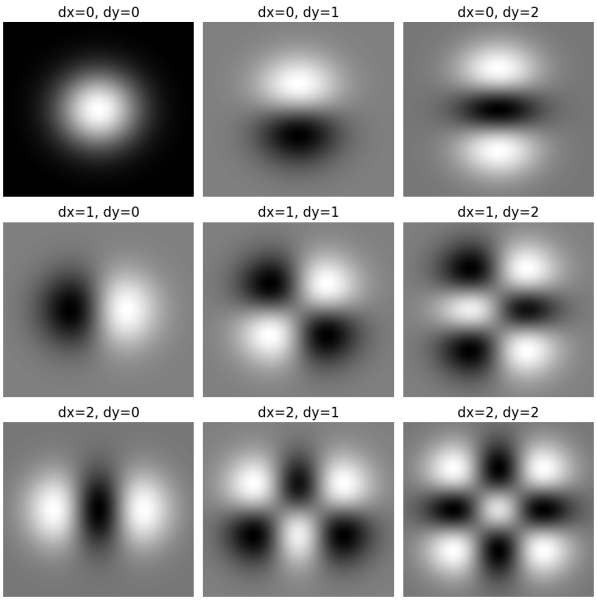

Figure 3.4: Visualisation of Gaussian Derivative Basis Functions. The order of the partial derivative in the y direction increases from left to right, and the x direction increases from top to bottom. Our choice of basis functions for a given layer is determined by the hyperparameter M. The basis includes all Gaussian derivatives where the maximum order of their partial derivatives is $\leq M$

$$\begin{aligned} H_0(x) &= 1 \\ H_1(x) &= 2x \\ H_2(x) &= 4x^2 - 2 \\ H_3(x) &= 8x^3 - 12x \\ H_4(x) &= 16x^4 - 48x^2 + 12 \\ H_5(x) &= 32x^5 - 160x^3 + 120x \end{aligned} \tag{3.4}$$

Additionally, we explore a Cosine basis and a Fourier basis. The Cosine basis is given by:

$$B(x,y,u,v) = cos(\pi ux)cos(\pi vy) \tag{3.5}$$

where x and y are the spatial coordinates where the basis is evaluated, u and v are the frequencies in the x and y directions respectively. We set the maximum frequency of the cosine waves via the hyperparameter M such that we consider all directional frequencies from the set $(u,v) \in \{(0,0),(0,1),...,(M,M)\}$. The Fourier basis in the complex domain is given as:

$$B(x,y,u,v) = e^{2\pi j(ux+vy)} \tag{3.6}$$

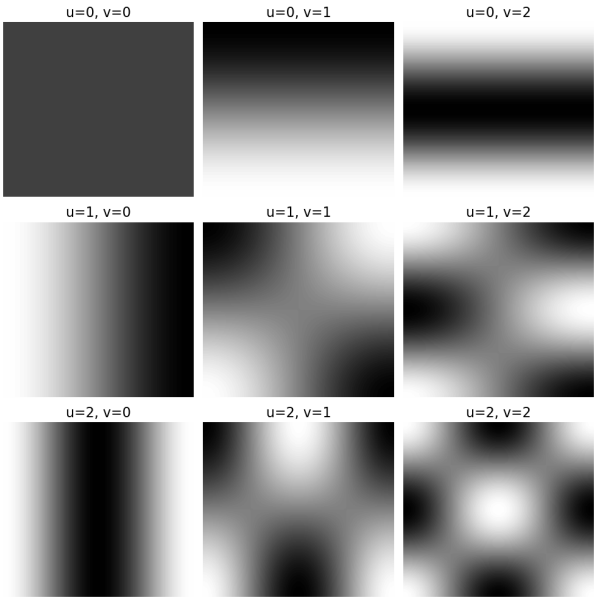

Figure 3.5: Visualisation of a Cosine Basis, u and v refer to the frequency in the horizontal and vertical directions respectively. The Gaussian derivative basis bears resemblance to a Cosine basis multiplied by a Gaussian windowing function.

Again like in the Cosine basis case we we consider all directional frequencies from the set $(u,v) \in \{(0,0),(0,1),...,(M,M)\}$, where M is a hyperparameter. Finally, we separate the complex valued basis functions into their real and imaginary components to avoid the use of complex weights when linearly combining the basis. Note that in the case of the Cosine and Fourier basis, we renormalize the edge labels for each receptive field so that the coordinates lie in the range of (0, 1) rather than (-1, 1) as is the case in the Gaussian Derivative basis or Implicit Neural Filters.

In all cases, it is simple to learn convolution filters via simply learning a linear combination of the basis functions, for example using a linear layer. We can regularize the learned filters by truncating the basis space to only use lower frequency basis filters, which may also have additional benefits with regard to mitigating aliasing.

Our choice for using Gaussian Derivatives in our main experiments was that they were found to perform best during early development phases. Additionally, there was consideration to exploit their steerability and relation to scale-space theory to build locally scale and rotation equivariant convolution layers. Ultimately, this line of work was not pursued as such ideas had already been explored for uniform images. Nonetheless, this remains an interesting avenue for future work which we discuss in more detail in the final chapter.

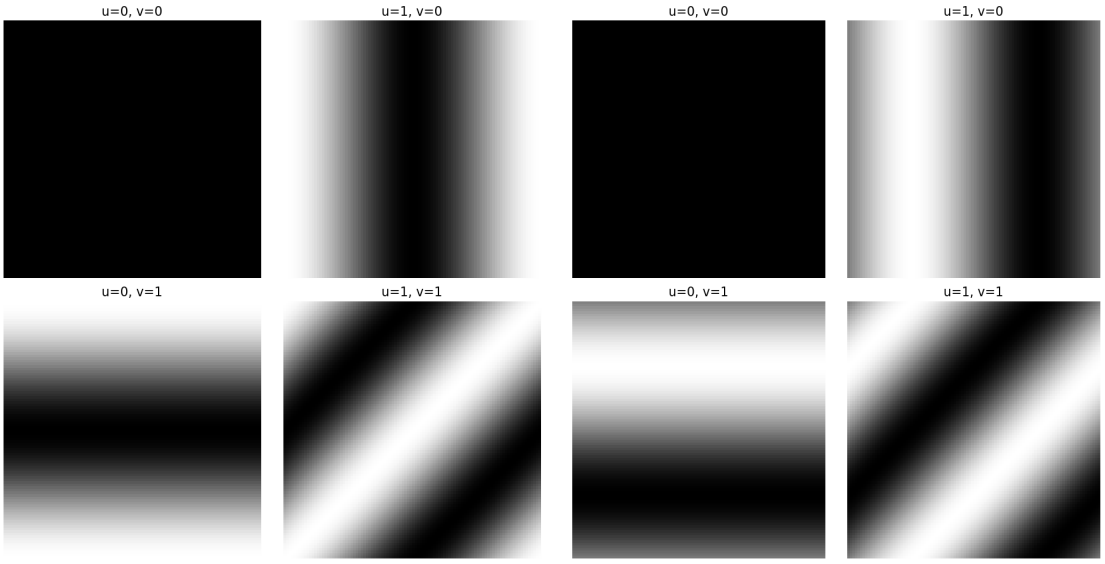

Figure 3.6: Visualisation of the real and imaginary components of a Fourier basis. u and v refer to the horizontal and vertical frequencies respectively. Directional frequencies are given by $\frac{u}{v}$

3.7 ImageNet-100 Classification Experiments

In this section we conduct several experiments on ImageNet-100, a 100-class subset of the ImageNet-1K dataset [49]. We aim to answer three main questions. Firstly, can foveated CNNs with with a fixation mechanism outperform Uniform CNNs in an image classification task (RQ2). Secondly, for foveated sampling layouts with a Cartesian-like coordinate frames, is accuracy still improved with our graph convolutional approach (RQ1)? Finally, can foveated CNNs outperform alternative biologically implausible methods (RQ4) such as Spatial Transformers [43] and Learning-to-Zoom [44]?

3.7.1 Implementation Details

We evaluate various foveated convolutional neural networks, including the Multi-FoV crop method [24, 25, 27], Cartesian Foveal Geometry [21, 22], and the blindspot log-polar method [13] (A visualisation of each sensor is provided in Figure 3.7). Additionally, we assess Spatial Transformer Networks [43], which utilize a localisation network to predict an affine transformation of a uniform sampling grid, enabling zooming and other transformations. Moreover, we report results for learning-to-zoom [44], which dynamically adjusts the sampling grid to densely sample salient regions based on the final convolution feature maps generated by the localisation network. Finally, to provide additional context, we conduct identical experiments using a uniform resolution CNN without a localisation network.

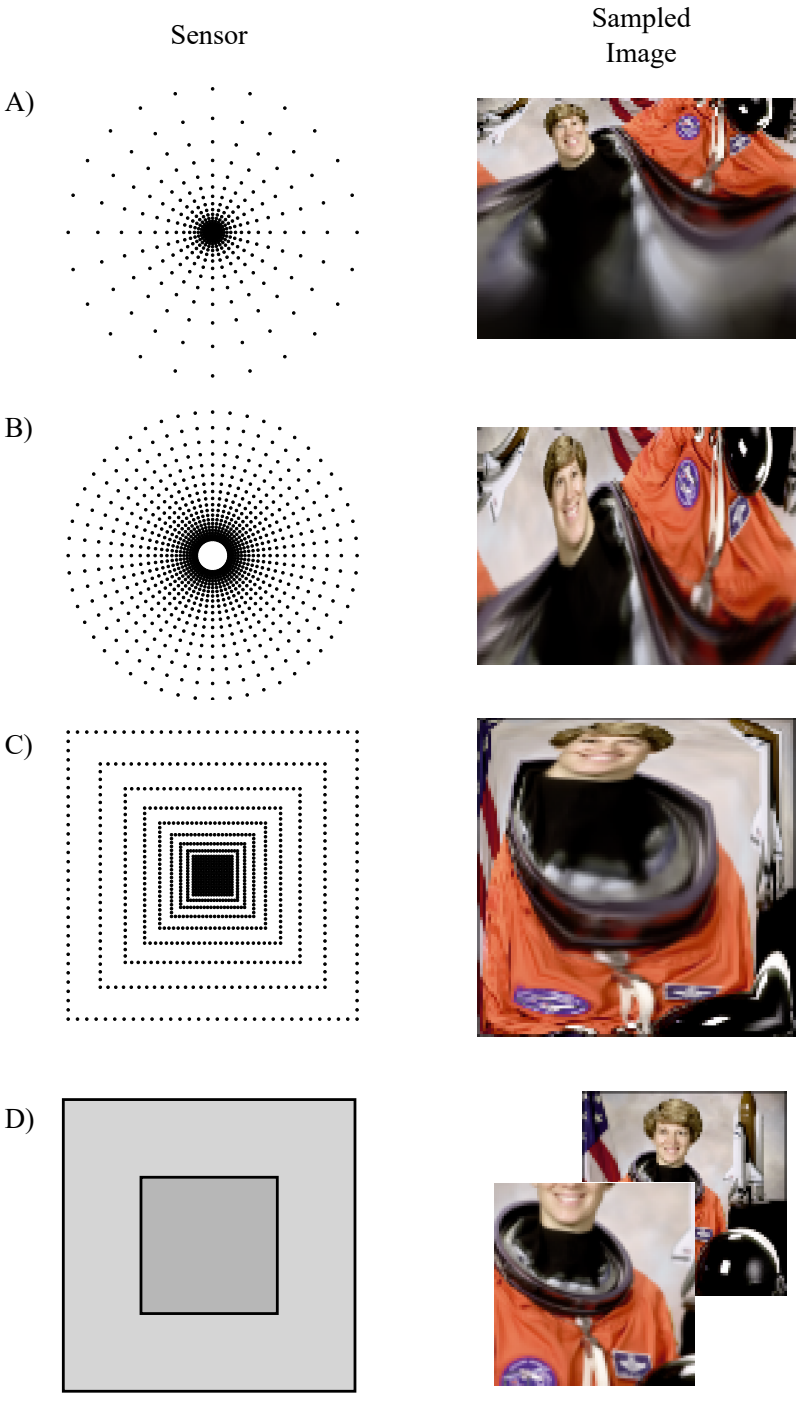

Figure 3.7: A visualisation of different foveated sampling methods and the resultant image. From top to bottom. log-polar (not used in experiments), Blindspot log-polar, Cartesian Foveal Geometry and Multi-FoV crops.

For the sake of fair comparisons we keep the underlying architectures as similar as possible for all methods. For all classifiers we use the "atto" variant of the ConvNeXt [87, 110] architecture which belongs to a family of State-of-the-art CNNs. All classifiers use a sensor with approximately 112^2 pixels except for the full-resolution uniform CNN which uses a 224×224 input. In the case of the 112^2 models we remove the final downsampling stage in the ConvNeXt architecture so that the final feature map has a 7×7 resolution. Similarly, the localisation networks are based on a truncated ConvNeXt atto in which the final convolutional stage is removed (i.e. the last two ConvNeXt blocks are removed) and replaced with a 1x1 convolution layer that outputs a 1 channel feature map. All foveated networks use the attention module presented in section 3.3. In the spatial transformer network the 1x1 convolution is followed by an MLP with 1 hidden layer, ReLU activation and batch normalisation. The MLP regresses the affine matrix used to transform the uniform sampling grid. In the learning-to-zoom model we use the saliency sampler as described in [44].

3.7.2 Training Hyperparameters

Networks are trained with a batch size of 64, and the AdamW [91, 92] optimizer. We perform a linear warm-up on the learning rate for 5 epochs, followed by cosine annealing for 85 epochs [111]. During training, we use TrivialAugment data augmentation [93], followed by resizing the shortest side to 256px and a random resized crop of 224×224 and a random scale variation of (0.08 - 1.0, the Pytorch default). At test time, we resize the shortest side to 256px and perform a 224×224 centre crop. Images are normalized using Imagenet mean and standard deviation. We use a learning rate of 0.004 and a weight decay of 0.005 for the feature extractors. For the localisation network, we use a learning rate of 0.0004 for the final 1×1 convolution (or MLP in case of the full affine spatial transformer) and a learning rate of 0.00004 for the convolution stages as done in [44]. We avoid weight decay on the localisation network which we found to be generally conducive to greater training stability, however we did not extensively test other weight decay values. We found that some methods that use a localisation network were prone to collapse during training; particularly spatial transformers. Therefore, we use weights pre-trained on Imagenet-1K for the localisation networks.

Characterizing the fovea radius as a percentage of the radius of the full sensor, we use the following settings for the log-polar method, we use a blindspot model with a fovea radius of 5%. For our Sunflower sensor, we use a fovea radius of 40% (equivalent to $r = 0.4$). The following sensors have their size described analogously for the case of square foveae. For the Foveal Cartesian Geometry sensor, we use a Fovea radius of 30%. The Multi-FoV sensor has a fovea of size 50%. We evaluate all methods on the test set using the best-performing model checkpoint on the validation set. We include the implementation details that are specific to our graph convolutional ConvNeXt atto in table 3.1. Hyperparameters were optimized through

Foveated Graph ConvNeXt atto configuration	Imagenet 100k Settings
Fovea Radius	40%
Stem - kernel size	16
Stem - sigma	1.0
Stem - max order	4
Blocks - kernel size	49
Blocks - sigma	0.8
Blocks - max order	4
Downsampling - kernel size	4
Downsampling - sigma	0.6
Downsampling - max order	0

Table 3.1: Stem refers to the initial convolution layer in the ConvNeXt architecture, Blocks refers to the configuration of the depthwise convolution layers in the ConvNeXt Blocks. Downsampling refers to the configuration of the downsampling layers that reduce spatial dimensionality between stages in the ConvNeXt architecture. Kernel size is analogous to kernel size in ordinary convolution layers, presented as the total number of spatial elements in the filter. Sigma determines the size of Gaussian derivative basis filters; max order refers to the maximum order of partial derivatives used in the basis.

random hyperparameter search [112] using validation accuracy.

3.8 Results and Discussion

We report top-1 accuracy on the Imagenet-100 test set in Table 3.2, along with the number of parameters and GFLOPs.

Method	**Operator**	**Sensor**	**# Input Pixels**	**# Fixations**	**Params (M)**	**GFLOPs**	**Accuracy (%)**
ConvNeXt	Conv	Uniform	50176	-	3.7	0.55	78.4 ± 0.5
ConvNeXt	Conv	Uniform	12544	-	3.7	0.20	70.0 ± 0.7
Ours (non-attentive)	Graph Conv	Our Sensor	12544	-	3.7	0.20	72.5 ± 0.4
STN	Conv	Uniform	12544	1	4.8	0.32	72.7 ± 1.0
PTN	Conv	Log-Polar	12800	1	4.8	0.33	70.7 ± 0.6
FCG-STN	Conv	FCG	12544	1	4.8	0.33	71.0 ± 0.4
Fov STN	Conv	Multi-FoV Crops	12800	1	4.8	0.33	71.8 ± 0.5
Fov STN (ours)	Graph Conv	Our Sensor	12544	1	4.8	0.32	74.2 ± 1.1
Learning to Zoom	Conv	Deformable Grid	12544	1	4.8	0.32	75.8 ± 1.4

Table 3.2: Top-1 Accuracy on Imagenet100. We split the table into two sections. Top: Passive vision models. Bottom: Active vision models. GFLOPs for graph convolutional models are reported as local filters but are trained with global filters and masking for computational efficiency.

3.8.1 Foveated vs. Uniform CNNs

Comparing the foveated CNNs against uniform CNNs, we show that in all cases classification accuracy is improved by adopting a foveated approach, however this comes at the expense of additional GFLOPs from the localisation network. In some cases, such as in the case of the log-polar sensor and Cartesian Foveal Geometry sensor, this improvement is minor (only +0.7% and 1.0% accuracy respectively). In the case of our graph convolutional method with our newly proposed sensor, as well as the Multi-FoV crop sensor, there is a significant improvement (+4.2% and +1.8% respectively). Independent t-tests show these to be statistically significant: T-stat=5.58, P-value=0.01 and T-stat=3.62, P-value = 0.02 for our sunflower sensor and the Multi-FoV sensor respectively.

Furthermore, we find our graph convolutional method with fixations is able to outperform a uniform CNN with equivalent GFLOPs by 2.5%. Both experiments pertaining to our graph convolution method make a strong case for its use in building foveated CNNs as it not only outperforms a uniform CNN, but all other foveated CNNs built from standard convolution layers. Research Question 2 asked whether foveated CNNs could outperform uniform CNNs in image classification. The results presented here and in Chapter 2 suggest that this is the case, a fixation mechanism is often necessary to achieve this.

This contrasts the recent study by Torabian et al. [20], which found foveated CNNs to perform worse than uniform CNNs even when fixating. We hypothesize that the difference in our findings can be attributed to the different fixation mechanisms. Torabian et al adopt a pretrained DeepGaze architecture [56, 57], which aims to predict where humans would look on an image. There may be a divide between where humans fixate on an image, and where an optimal fixation would be for a given foveated CNN. In contrast, our method learns a fixation mechanism, directly by minimizing the classification loss, and may yield more optimal fixations for foveated CNNs as a result.

3.8.2 Discussion on Coordinate Frames

Our motivation for adopting a graph convolutional approach was driven by the hypothesis that geometric transforms of foveated images to grid representations imposes sub-optimal coordinate frames for convolution that results in an inhibitory inductive bias for image classification. We showed previously that a Cartesian-Like coordinate frame was favoured over a log-polar one. In this chapter we compared against two other foveated sensors that adopt Cartesian-Like coordinate frames for representing foveated images, Cartesian Foveal Geometry (CFG) [22] and Multi-FoV crops [25]. Again we show that Cartesian-Like coordinate frames outperform log-polar coordinate frames. This is minimal in the case of CFG (+0.3%). In the case of our Graph Convolution Method and Multi-FoV crops this improvement is more noticeable (+3.5%

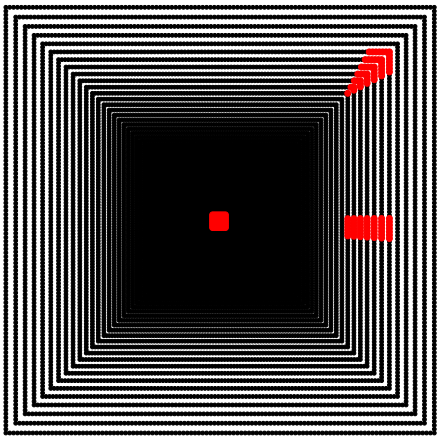

Figure 3.8: Visualisation of the Cartesian Foveal Geometry sampling layout in Cartesian Coordinates. In red, we show the shape of 7×7 filters on various parts of the image. As the filter moves across the field, its shape is transformed from square in the fovea, to kite like in the corners. We can describe the shape of the filter as it moves across the visual field through a projective transform.

and +1.1% respectively). Notably both of these methods share similarities in that with respect to the original image, filters are only scaled and translated, while on the CFG sensor filters undergo projective transformations as they move across the visual field (Figure 3.8).

We hypothesize that the reduced performance of CFG relative to other Cartesian-Like methods is again a result of the coordinate frame imposing a poor inductive bias, or at least one that does not constrain the function space in a meaningful way. While filters undergoing projective transformations across the visual field is not intrinsically problematic, it is unclear of its utility. Berenguel-Baeta et al. discuss a similar problem of filter deformation when applying CNNs to fisheye images, and present a method to adapt the shape of the kernel to mitigate this deformation, showing improved performance in depth estimation and semantic segmentation. This provides some supporting evidence that the projective transformations of filters on CFG images might similarly inhibit performance. In contrast, our graph approach and the Multi-FoV sensor means filters only scale with eccentricity and may explain why these approaches perform better.

We posit that the performance increase of our method relative to Multi-FoV crops can be attributed in part due to the ability of foveal and peripheral features to interact at intermediate stages of processing. In the Multi-FoV crop case, they only interact at the end through spatial pooling. We did consider an alternative approach in which foveal and peripheral regions are concatenated in the channel dimension at input. We found this to perform slightly worse (71.0%) than merging these features after convolutional processing.

3.8.3 Discussion on Biologically Implausible Methods

In Research Question 4, we asked whether foveated CNNs could outperform biologically implausible methods that operate under similar principles to foveated sensing. We consider two alternative methods, Spatial Transformers [43] and Learning-to-Zoom [44] that have the ability to sample parts of a visual scene in greater detail in an adaptive manner, but are not foveated. Spatial Transformers adaptively regress an affine transformation of a sampling grid, permitting behaviours such as zooming, rotating and translating. Learning-to-Zoom adapts the sampling grid intelligently on a per-instance basis to increase sampling resolution at regions that are deemed salient by the localisation network.

We find that out of all foveated CNNs, only our graph CNN was able to outperform a Spatial Transformer (+1.5%). However it could not outperform Learning to Zoom, which achieved 75.8% accuracy, relative to the foveated graph CNN's 74.2%. This is perhaps unsurprising, Learning-to-Zoom predicts an optimal sampling layout on a per-image basis, while the sampling layout of the sensor is fixed in foveated CNNs and only adapted via translations. It could be argued that Learning-to-Zoom is a more expressive generalisation of the principles behind foveation, i.e. the space-variant allocation of visual computation resources across an image.

This raises an important consideration for foveated vision systems. The assumption is that the emergence of foveated vision in biological systems is due to it providing some functional benefit, which may indeed be optimal but only within the constraints of biology. While Learning-to-Zoom is biologically implausible, the lack of biological constraints may allow for an even more powerful mechanism than what is possible in nature. This is seldom stated in the literature concerning foveated vision but it is important to consider. This is not to say that this idea usurps foveated vision in its entirety, as object recognition is a single task in the repertoire of a general vision system. Future work should continue to contrast against these biologically implausible methods in a wider repertoire of visual perception tasks.

3.9 Comparison of Different Basis Functions

In this section we compare our graph convolution layer using different methods for G, the edge conditioned graph convolution filter (results presented in Figure 3.9). For the purposes of these experiments we reuse the isometric architecture in Chapter 2, but with the addition of the localisation network used in this chapter. Our motivation for the choice of this architecture over the ConvNeXt atto is that it allows us to remove additional spatial aggregation mechanisms such as pooling from the architecture. As such it is purely built from our graph convolution operator and 1x1 convolutions. We conduct experiments on the Imagewoof dataset using the same experimental setup and training scheme as in the Imagenet-100 experiments. For additional context we also consider models with no fixation mechanism. For Gaussian Derivatives we consider

derivatives up to a maximum directional derivative of 4 and sigma of 0.8. For the cosine and Fourier basis we limit the maximum frequency to be 3. These settings were the optimal found via random hyperparameter search. For MLP based methods, we use single hidden layer MLPs with 16 hidden neurons.

The initial hypothesis was that all basis functions would perform approximately the same. We found this not to be the case, and there was in fact noticeable differences in performance between them. We found MLPs with Sine activation to perform the best. This corresponds to a learned sinusoidal basis. Gaussian Derivatives also performed well at 80.3% while the Cosine and Fourier bases performed relatively poorly, at 78.1% and 74.3% respectively. This poor performance is interesting considering the similarities between INF with Sine activation. It is not exactly clear why there is such a noticeable performance difference. Moreover, the performance gain from the inclusion of the fixation mechanism differs wildly for different methods. While we leave this for future work we will offer two hypotheses.

Firstly, MLP based methods learn a basis space for representing filters, while other methods have a fixed basis space. As such, we do not have to tune the specific properties of the basis space, and are less reliant on finding good hyperparameters, explaining the Sine activated MLP's improvement in performance over the fixed basis methods.

Secondly, of the fixed basis methods, the Gaussian Derivate method performs best. Unlike the other basis spaces, this one is a product of a Gaussian windowing function and Hermite polynomials. Note that we used a fixed number of pixels to represent a local patch on a graph which does not yield consistent spatial arrangements of points, unlike local patches on a grid which are always of the same shape. The attenuation of distant pixels from the centre of the receptive field via Gaussian windowing, may improve robustness to the specific arrangement of pixels that are being filtered, and explain the improved performance of the Gaussian Derivative method.

3.10 Fovea Radius

Research Question 5 pertains to the parameterisation of the foveated sensor and the sensitivity of classification accuracy to this parameterisation. In particular, we attempt to find whether the optimal parameterisation is solely dependent on the dataset and consistent across all architectures, or whether it is dependent on the architecture as well.

Our proposed sensor allows for the radius of the fovea and its sampling density to be controlled by the hyperparameter r and d respectively. Smaller values for r and higher values for d increase the visual acuity of the fovea at the expense of decreased peripheral resolution. Recall that we define d so that the foveal resolution, and the resolution immediately outside the fovea, are

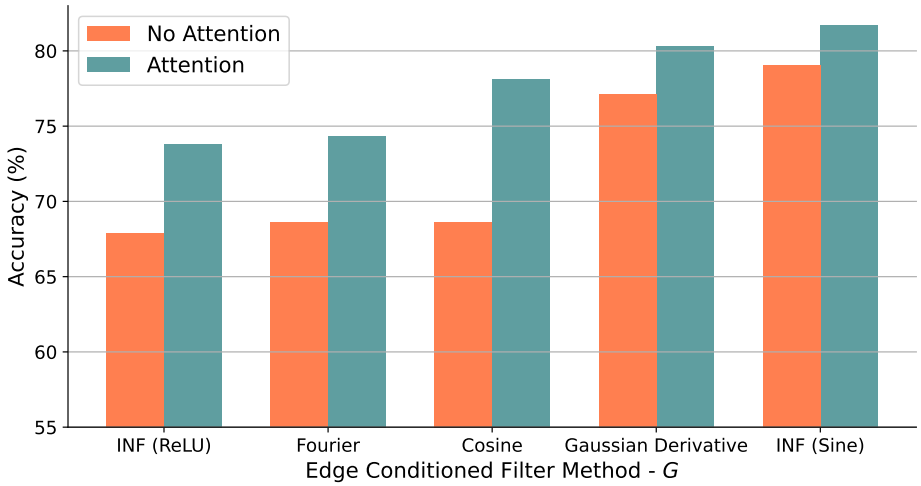

Figure 3.9: Top-1 accuracy on Imagewoof for Foveated Isometric Graph ConvNeXts when using different edge-conditioned filter methods. INF refers to Implicit Neural Filters 2.4.4. For sake of comparison between Chapter 2. we perform experiments with and without a fixation mechanism. All architectures use a fovea radius of 30%.

approximately the same and exhibit a smooth transition between the two regions.

We evaluate three graph convolutional architectures, the ConvNeXt atto with 112^2px sensor and two Isometric ConvNeXt architectures with 112^2px and 56^2px sensors. The internal feature map resolution of the Isometric architectures is 4 times smaller than the sensor resolution owing to the initial convolutional downsampling stage (28^2 and 14^2 respectively). We refer to these architectures as Iso-28 and Iso-14 respectively. Architectures are trained on the Imagewoof dataset using the same hyperparameters and training setup as in section 3.7.2. We sweep over a range of fovea radii and report results averaged over three runs for each method. Results are shown in Figure 3.10.

We find that for all cases, architectures with a fovea radius of 0.9, i.e negligible foveation, exhibit the lowest performance relative to other fovea radii. We also see that extreme amounts of foveation, e.g. where the fovea radius is just 0.1, also leads to less performant architectures. Architectures with larger input resolutions and internal resolutions derive less benefit from using foveated sensors. At their optimal fovea radii values, ConvNeXt atto and Iso-28 only achieve a marginal performance gain of 1.8% and 1.5% respectively, over a fovea radius of 0.9. The benefit of foveation is more pronounced in the Iso-14 architecture, where the optimal fovea radius of 0.2 has a 3.4% increase in recognition accuracy compared to the 0.9 radius variant. Additionally, we see that architectures with a higher input resolution also perform well with relatively large fovea sizes, while in the Iso-14, there is a notable trend towards increased performance with smaller more high-resolution foveae.

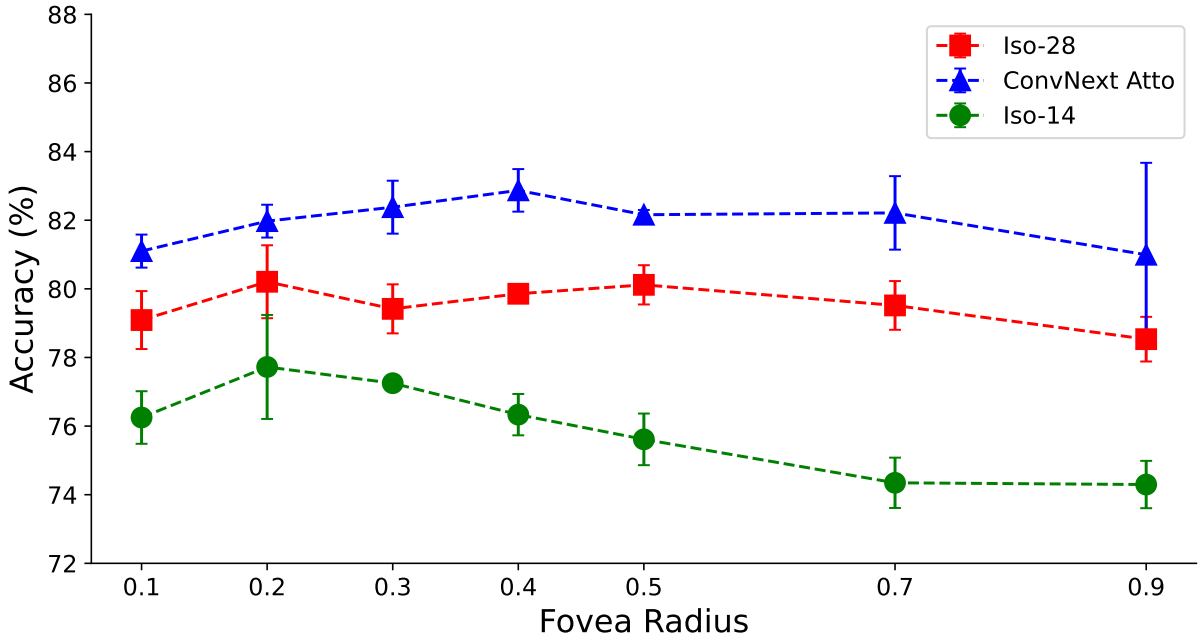

Figure 3.10: Top-1 accuracy on Imagewoof for different fovea radii. We contrast three architectures. ConvNeXt atto operating on a 112^2 foveated sensor, and two Isometric ConvNeXt architectures (depth 8) operating on 112^2 and 56^2 foveated sensors. Each experiment was run three times with mean and standard deviation reported.

Much like resolution scaling, we can see diminished returns in increasing foveation for architectures that already operate at a relatively high-resolution. Additionally, smaller higher resolution foveae come at the expense of decreased peripheral resolution. This will likely be a trade-off in which the loss of useful visual information in the peripheral field of view outweighs the gain in useful visual information gained by a higher resolution fovea. Furthermore, decreasing the fovea radius decreases the area of the visual field in which the system has translation equivariance to visual features. As such systems with a larger radius may be more robust to incorrect fixation predictions. Again this highlights a trade-off between increasing foveation at the expense of other desirable qualities we might get with less or no foveation.

There may be several factors that contribute to the optimal fovea radius. These experiments have presented some preliminary evidence that lower resolution architectures benefit more from foveation, and this is intuitive. Importantly, this shows that the optimal amount of foveation is not solely dependent on characteristics of the dataset and is partially dependent on the subsequent architecture, but perhaps in non-trivial ways. While searching for optimal parameters over this space is not wholly unfeasible, it does introduce an additional level of complexity to the hyperparameter optimisation of foveated vision architectures. In the next chapter we investigate a foveated sensor that can be optimized via backpropagation to alleviate this problem.

3.11 Limitations

The primary limitation of this chapter, particularly with regard to the main experiments were that of scope. While we answered Research Questions 1 and 2, this was only on one dataset and with one architecture. These limitations were primarily born out of what was feasible to complete during this thesis with available computing resources. Nonetheless, in order to better substantiate the findings in this chapter, it is necessary for experiments on a wider range of datasets to be conducted.

While there are many publicly available image classification Datasets there is a degree of difficulty in finding suitable ones to perform the previous experiments on. Datasets such as CIFAR10 [55] and MNIST [54], are too low in resolution to reasonably apply foveated downsampling to. Other datasets such as Caltech Birds, Stanford Dogs and Oxford Pets are comprised of relatively high resolution images but are difficult to use when training neural networks from scratch and typically require pretraining on datasets such as ImageNet1k.

3.12 Conclusion

In this chapter we introduced a fixation mechanism for foveated CNNs that allowed the foveated sensor to be aligned with objects of interest in an image. This was done in order to better test the claims made in the previous chapter and provide more comprehensive answers to Research Questions 1 and 2.

We conducted experiments on Imagenet100, comparing an array of different foveated sensors. We showed conclusively that, on Imagenet100, adopting a foveated strategy yielded consistent improvements in classification accuracy relative to the more conventional Uniform paradigm. This comes with the caveat that the fixation mechanism was necessary to achieve such an improvement and this does incur a computational cost.

In the previous chapter, we showed the pivotal role of the coordinate frame used to represent foveated images had on classification accuracy (RQ 1). Our results in this Chapter further corroborate these findings and that they hold in the presence of a fixation mechanism. In particular, we again showed that Cartesian-like coordinate frames are favourable however may still be suboptimal in some cases such as for the CFG sensor [22]. Our graph convolution operator allowed us to derive a better coordinate frame for representing foveated visual data, as it did not necessitate that the coordinate frame yielded a grid-like pixel structure. This allowed our graph convolutional method to significantly outperform all other foveated CNNs built with standard convolution layers. This presents a compelling case for adopting our graph convolution when processing foveated images.

In Research Question 4 we questioned whether foveated CNNs could outperform biologically implausible methods that adaptively allocated visual processing resources over the visual field. Despite these methods being highly related to the guiding motivations of foveated sensing, comparisons between them are yet to be made in the literature. We showed that our foveated graph CNN could outperform Spatial Transformers, but not Learning to Zoom. As mentioned previously, this does not undermine foveated vision in its entirety, however we argue that it is imperative that future works continue to contrast against such methods in order to properly gauge the utility of foveated sensing in the wider field of computer vision.

Finally, we explored the sensitivity of classification accuracy to the size and sampling resolution of the fovea in a foveated sensor (Research Question 5). Using three different instantiations of a foveated graph convolutional ConvNeXt, we report classification accuracy over a range of different fovea radii. We showed that both very large and very small fovea decreased classification accuracy. Furthermore, we showed that higher resolution architectures generally favoured larger fovea (i.e. less foveation), and that the architecture was also less sensitive to the fovea radius parameter. These findings demonstrate that the layout of a foveated sensor does not have a globally optimal design for all visual data or a given dataset, and is partly dependent on the subsequent architecture.

Chapter 4

Non Convolutional Foveated Vision Architectures

4.1 Introduction

The work presented thus far has been in the context of CNNs. Recently, a variety of non-convolutional vision architectures have been proposed. The most eminent of these is the vision transformer (ViT) [45], which translates the original transformer architecture [113] (designed for natural language processing) to a vision context. At their core, they split input images into visual tokens (small image patches). Spatial features are computed over these tokens using the self-attention mechanism. Later works explore alternatives for self-attention and show that even linear layers [46, 114] suffice for building highly performant vision systems.

These architectures are interesting in the context of foveated vision for several reasons. Firstly, they operate on sequences of visual tokens, not grid-structured images. As such, they are far more amenable to the space-variant structure of foveated images. Secondly, spatial relationships between the tokens are learned (for example, via positional embeddings). In the previous chapters we discussed the importance of representing foveated data in suitable coordinate frames to maximize the performance of the CNN. These architectures circumvent this problem and instead attempt to learn such behaviour via the data itself. Finally, these architectures do not impose translation equivariant behaviour and can potentially learn different computations for different regions of the foveated image.

In this Chapter, we predominantly focus on Research Question 3, which asks whether non-convolutional architectures exhibit improved classification accuracy when operating on foveated images. In particular, we consider two candidate architectures: Vision transformer, which uses self-attention, and ResMLP, which uses linear layers for extracting spatial features across tokens.

We propose a simple variation of the image tokenisation procedure that extracts a sequence of image patches arranged in a foveated fashion. We use this tokenisation method with 'out-of-the-box' ViT and ResMLP architectures and show that they can effectively process foveated arrangements of visual tokens with no extra machinery. We incorporate these systems with the fixation mechanism described previously. We conduct image classification experiments on ImageNet100, comparing against a uniform tokenisation method (i.e. the standard method). Furthermore, we revisit Research Question 4 contrast against Learning-to-Zoom [44] in this new context. Furthermore, we propose an extension of our foveated tokenisation method in which the sampling resolution over the visual field can be learned via backpropagation (RQ6). We show that our method shows emergent foveation and converges on a similar sampling layout to the optimal one found through hyperparameter tuning.

We perform further experiments pertaining to the sensitivity of classification accuracy to the sampling layout, again showing that lower-resolution architectures benefit from higher-resolution fovea 5. Additionally, we explore how properties of the dataset might influence the optimal amount of foveation. We synthesize several toy datasets derived from MNIST, in which digits are randomly scaled based on a uniform scale distribution and placed on a 224×224 blank canvas. Using a simple Multi-Layer Perceptron (MLP) operating on a foveated sensor comprised of only 784 pixels, we show that the optimal degree of foveation is linked to the dynamic range of the scale distribution, with higher resolution foveae becoming increasingly necessary for higher dynamic ranges.

4.1.1 Chapter Structure

This chapter is structured as follows:

- We first cover relevant related work concerning both foveated and uniform non-convolutional architectures.
- We then provide relative background knowledge pertaining to spatial feature extraction using self-attention layers and linear layers.
- Our proposed foveated tokenisation method is then detailed along with an extension that allows its sampling layout to be learned via backpropagation.
- We then outline the overall architectures that are considered in this Chapter.
- Experiments are then detailed, and results and discussions are then presented.
- Finally, we discuss the limitations of this chapter and summarize its main contributions.

4.2 Background

Vision transformers [45] stand as the first significant paradigm shift from CNNs in neural network based vision systems. Transformers [113] were originally used for natural language processing (NLP), and aimed to address problems with recurrent neural networks in modelling long range interactions as well as training parallelism. The primary ingredient to this success was the self-attention mechanism. Briefly, the self-attention mechanism operates on a set of tokens (vector embeddings of parts of input data, e.g. words or image patches) and constructs query, key and value embeddings via linear projections. An outer product between queries and keys describes the attention weights between each token to all other tokens. Softmax activation is applied to these attention weights, which are then used to perform a weighted sum of values to arrive at new representations for each token. In its purest form, a transformer operates on sets and is equivariant to the permutation of said set. As such Transformers do not have any understanding of the order of tokens. Positional information is added to token vectors via positional encodings or learned embeddings to make transformers position-aware [45, 113]. Vision transformers demonstrated extremely good performance with minimal vision inductive priors and proved to be highly scalable architectures. Subsequent works have extended this original design in a variety of ways, such as incorporating hierarchical processing akin to CNNs [115], data-efficient training regimes [116], and local attention [117, 118].

By and large, these architectures all leverage the self-attention operator to model spatial interactions between features. Comparisons can be drawn between convolution and self-attention. Han et al. show that local self-attention resembles depth-wise convolution [119] with dynamic weights. That is to say that the weights used to perform weighted sums of spatial features are a function of the input data itself and not fixed as in convolution. We can draw similarities between self-attention and our graph convolution proposed in earlier chapters in that neither method assumes a grid-structured domain and uses explicitly provided (or learned) spatial information to model spatial interactions between features. In fact, both methods can be viewed as specific flavours of message-passing graph-neural-networks [50, 64].

Vision transformers have inspired several subsequent architectures that use alternatives to the self-attention operation for spatial interactions. ResMLP [46] and MLP-Mixer [114] concurrently proposed a simple linear layer as an alternative to self-attention. In such a formulation, the attention weights can be considered fixed and content unaware, with positional information being implicitly defined by the connections between input and output neurons. Both show competitive performance to transformer architectures and promising scalability. RepMLP [120] proposes an extension to CNNs in which a linear layer operates in parallel to convolution during training. Convolution can be merged into the linear layer at inference, allowing the network to leverage the locality and translation equivariance priors of convolution while also utilizing linear layers to model global spatial interactions. Gao et al. [121] present a generalized view of both

MLP methods and self-attention through their container module, combining content-aware and static spatial interactions. g-MLP [122] employs a spatial gating mechanism similar to squeeze-excitation mechanisms [123] but applied spatially rather than channel-wise. Features are split in the channel dimension, the MLP spatially aggregates information from one split which is then applied as a multiplicative interaction between the other split. Again, these methods show competitive performance to transformer architectures, even exceeding, in some cases, at image recognition. Elsayed et al. [124] propose a locally connected network for vision, which allows for the same locality prior as convolution but allows filter weights to vary across the visual field. Weights are regularized to have a low-rank decomposition into a set of basis weights. Unlike the previously mentioned methods, locally connected layers do not model long-range interactions.

One commonality of these architectures is their relaxation of the translation equivariance prior in CNNs. As such, they can perform space-variant computation, which has shown to be advantageous in problems such as Face Recognition [125]. These qualities may align particularly well with space-variant sampling of foveated vision, allowing different computations to be applied to foveal and peripheral regions, for example. Karpathy et al. [26] showed that separate CNNs applied to foveal and peripheral regions of a Multi-FoV crop sensor learn different weights, suggesting it may be beneficial to incorporate space-variant processing on foveated images. This does require the network to learn translation invariant behaviour, unlike CNNs. As such, these architectures may additionally resonate with the active vision paradigm by solving translation invariance via eye movements and diminishing the dependence on solving translation generalisation through the weights of the neural network.

In the context of foveated vision, there are relatively few architectures that consider non convolutional approaches to processing foveated images. Jonnalagadda et al. [40] proposed a foveated transformer architecture for image classification, however, foveated sampling is applied to the output of a convolutional backbone and not utilized through all stages of the network. In contrast, the work presented in this chapter uses foveated images as input to a neural network. Min et al. propose PerViT [126] which extends self-attention with a novel positional embedding method to segregate tokens into foveal, para-foveal and peripheral regions. While this method shows improved performance on object recognition benchmarks, it does not leverage foveated sampling to reduce computational complexity, and instead views foveation in the context of focusing attention on a particular part of a scene. Mnih et al [28] use a simple MLP, however this work was primarily concerned with actively attending to scenes in a recurrent fashion and not the feature-extraction process. Only toy datasets were considered, and the simplistic MLP may not be feasible to apply to more challenging natural image datasets. Nakada et al. [31] propose a locally connected network for processing foveated images, again however, only simulated visual environments are considered and it remains unclear whether such a design extends to more challenging visual environments.

4.3 Methods and Materials

We consider two alternatives to convolution for extracting spatial features from a foveated image, namely self-attention and linear layers.

Self-attention, originally proposed by Vaswani et al. [113], operates on a set of tokens $x = (x_1, x_2, ..., x_N)$. In the case of vision transformers, a token is a d-dimensional linear projection of an image patch $x_i \in \mathbb{R}^d$. For each token, self-attention applies a linear projection to create query, key and value embeddings, creating corresponding Q, K, and V matrices for the whole token set. Self-attention is then given as:

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{\sqrt{d}})V \tag{4.1}$$

d, in this case, refers to the dimensionality of the projected query embeddings and predominantly subserves easier optimisation. In practice it is useful to run multiple different self-attention operations in parallel, such that each token can attend to the same token in different ways. This variant is known as multi-head attention and can simply be achieved by running multiple self-attention layers in parallel, concatenating the outputs and applying a further linear projection.

The linear layers used in architectures such as ResMLP [46] and MLP-Mixer [114] are comparatively simpler than self-attention. Considering our set of input tokens x as a matrix $X \in \mathbb{R}^{N \times d}$, a weight matrix $W \in \mathbb{R}^{N \times N}$ and bias terms $b \in \mathbb{R}^N$ associated with the linear layer, the linear mixing operation is simply given as:

$$Linear(X) = X^T W + b \tag{4.2}$$

A commonality both with mixer architectures and vision transformers (and many other architectures such as ConvNeXt [87, 110]) is the use of a pointwise feed-forward network. Continuing with the terminology of tokens to describe the processing steps. After spatial features are computed for each token, a single hidden layer MLP processes each token independently. Typically a hidden size of $4\times$ the token's dimensionality is used with GELU activation.

4.4 Patch Based Foveated Sensor

Many transformer-like architectures use an initial convolution layer with non-overlapping filters to tokenize the input image [45]. While it is possible to use the graph convolution layer proposed in Chapter 2, we instead propose a simpler method of foveated sampling in which small patch-sized sampling grids are arranged in a foveated fashion (Figure 4.1). Specifically, we define the

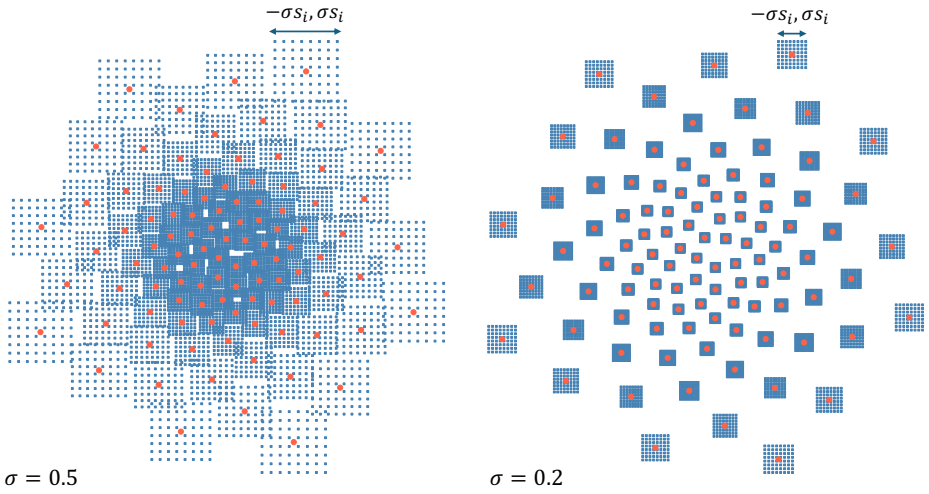

Figure 4.1: Visualisation of the proposed patch sensor. Blue points show the spatial positions of sampling kernels for the entire sensor. Red points show the spatial centres of each patch. These centres are generated by the Sunflower sensor proposed in Section 3.5; however, other methods for generating a foveated arrangement of points are viable.

centre of each patch using the sunflower retina as described in section 3.5. Each centre is the origin for a $p \times q$ grid, which dictates the number of sampling kernels for each patch. the x and y coordinates of the i^{th} grid lie in the range of $(-\sigma s_i, \sigma s_i)$ where σs_i controls the spatial extent of the patch. We use a k-nearest neighbours operation to find the average distance between a patch i and its 2 nearest neighbours to determine s_i. We introduce a scaling term σ for all patches to control the amount of overlap between them. A sigma of 0.5 was chosen qualitatively so as to minimize overlap between patches while still providing good coverage of the pre-sampled image. While we use the sunflower foveated sensor for this work, it is possible to use arbitrary foveated arrangements of pixels to serve as the centres of the patches.

Sampling of the input image I can then be performed via bi-linear sampling using the same method as described in section 3.4. The foveated image $\hat{I}$ is a tensor of shape $\hat{I} \in \mathbb{R}^{N \times C \times pq}$ where N is the number of patches, C is the number of channels in the input image and p and q are the spatial dimensions of the patch. Embeddings for each token can be computed by reshaping the tensor $\hat{I}$ from $N \times C \times pq$ to $N \times Cpq$ and applying a linear projection to a d-dimensional embedding for each patch to arrive at $z \in \mathbb{R}^{N \times d}$, the tokenized representation of I. For the purposes of all spatial interaction methods, we will frame the mechanism in the context of how they operate on these tokens.

4.5 Learnable Sampling Layout

As shown in the previous chapter, the parameterisation of the sampling layout, i.e. how sampling resolution varies across the visual field, can have a significant effect on classification accuracy. Finding the optimal parameterisation was achieved through a hyperparameter search, however this is costly in certain scenarios. Here we explore a foveated sensor which has learnable parameters that control the sampling layout and can be optimized via backpropagation from a downstream loss function. This allows us to jointly optimize the architecture weights as well as the sampling layout, removing the need for hyperparameter optimisation. Additionally, if optimizing this layout to minimize the classification objective does show emergent foveation, it can provide compelling evidence that foveation is beneficial to visual perception tasks such as object recognition. A final note is that while it is technically possible to achieve this with the graph convolutional architecture, it would require recomputing the graph after each gradient step which makes it unwieldy. In contrast the architectures considered in this chapter can work on arbitrary spatial arrangements of patches, and do not explicitly construct a graph , making it significantly easier to realize a learnable topography.

To achieve this we adopt a similar approach to the sunflower sensor proposed in Section 3.5. Recall that the sampling distribution is controlled by a function ζ, which is a piece-wise combination of a linear and exponential function to map the radial coordinate of each point in the sunflower arrangement to a new radial coordinate. Additionally, we discussed that it is trivial to shape the sampling layout of this sensor by simply using a different function for ζ.

Our previously proposed ζ function is not amenable to backpropagation in its current formulation as there is not a soft differentiable way to assign the points in the arrangement as foveal and peripheral points. Previous works have considered alternative methods to learn a sampling layout such as simply optimizing each points coordinates independently [127], or in a separable way for each x, y coordinate of a grid [41]. [44] imposed a Gaussian smoothing function on the weights to mitigate the possibility of the points folding over on themselves. In early experiments we considered both the above methods to learn spatial resolution over the visual field, however we found that allowing too much freedom in positions of patches became challenging to optimize with a large number of patches.

Instead, we seek a formulation for ζ that utilizes a relatively low number of learnable parameters, but has the expressivity required to shape the sampling distribution over the full visual field. We opt for an N^{th} degree polynomial. We make some assumptions on the ideal form of this polynomial that allows us to constrain its coefficients. Firstly, we expect ζ to be a monotonically increasing function, i.e. the sampling density must decrease with eccentricity. We achieve this by ensuring that all coefficients are positive. Secondly, we expect $\zeta(0) = 0$, $\zeta(1) = 1$, ensuring that the centre of the field of view is sampled, and sampling extends to the borders of the uniform

image (note we normalize image coordinates to lie in the range of (-1, 1)). We enforce this constraint by removing the bias term in the polynomial and normalizing all coefficients such that they sum to one. A simple way to achieve this in practice is to apply a Softmax across the polynomial coefficients.

In a similar vein, we learn a polynomial function of eccentricity to determine the scale of the patch. In this case we reintroduce a bias term that allows for a minimum patch scale to be learned. Similarly we enforce that this function is monotonically increasing and that the coefficients sum to a 1 via softmax. We introduce a scaling term s as in the fixed static sensor to provide an upper bound for the scale of the patches. This is applied as a multiplication of each coefficient.

4.6 Architecture

The architectures considered in this chapter broadly follow the same two-stage design used in the previous chapter. We maintain the use of a truncated ConvNeXt atto [87]. Our newly proposed sensor can be aligned using the same method as the previous section, by adding the fixation as an offset to all sampling kernels. Again we use bilinear sampling kernels and exploit differentiable image sampling to train the localization network. The architecture is visualized in Figure 4.2. For the classifiers, we use 49 patch and 196 patch variations of ResMLP and vision transformer. They are all comprised of a stack of 12 vision transformer or ResMLP blocks. For the ResMLP models, the final features are pooled in the spatial dimension and passed to linear classifier. For transformer variants, we append a class token to the input and apply a linear classifier to this class token at the end.

4.7 Experiments

In this section we conduct image classification experiments on ImageNet100. These experiments aim to answer two questions in the context of non-convolutional vision models. Firstly, do foveated models perform better than uniform ones (RQ3). Secondly, can foveated models outperform Learning-to-Zoom, which was highly effective in a convolutional context (RQ4).

4.7.1 Implementation Details

All models are trained for 100 epochs with a batch size of 256, the AdamW[92] optimizer and cross-entropy loss with label smoothing of 0.1. For all models, we use a maximum learning rate of 0.004 and a OneCycle learning rate policy with a warm-up period of 33 epochs. For the learned sensor polynomials we used a slightly higher maximum learning rate of 0.01 (chosen manually). We use weight decay of 0.1 for all parameters except those pertaining to biases,

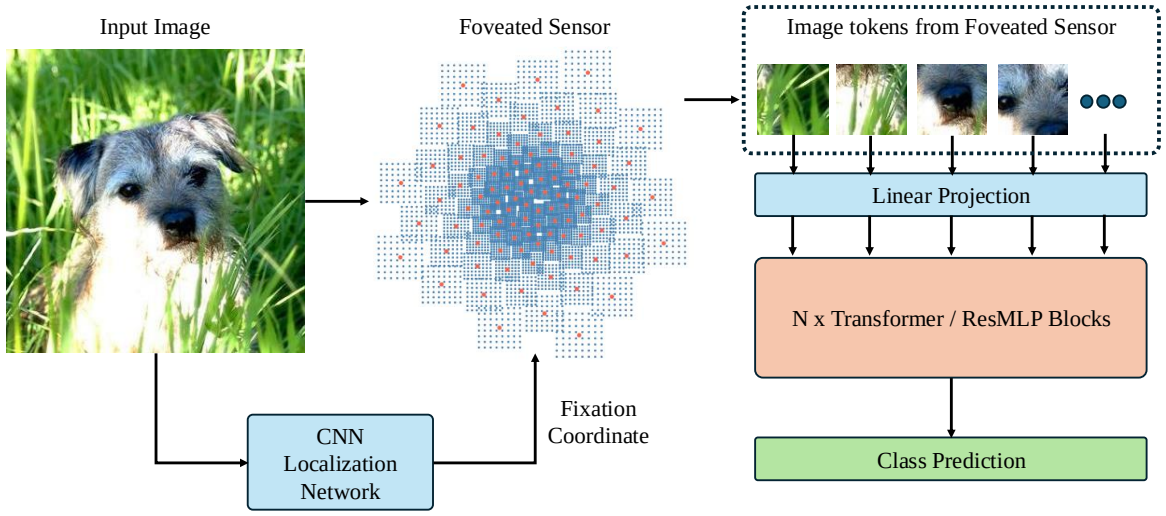

Figure 4.2: Schematic for non-convolutional architectures. The CNN localisation adapts the sensor, in this case a foveated sensor. The sensor extracts image tokens and feeds them to a non-convolutional architecture which then makes a class prediction. The primary difference between all methods evaluated in the next section is the choice of sensor.

normalisation layers, layer-scale, sensor parameters and the final convolution layer in the localisation networks. Due to large model sizes and limited compute we make use of mixed precision training and four gradient accumulation steps. Using no gradient accumulation may lead to slightly different results due to the batch normalisation layers in the localisation networks. For the patch sensor, we use a Fovea radius of 30% for the 49 patch models and 50% for the 196 patch models. Hyperparameters were chosen through random search using a validation set. This was only performed with the ResMLP-196 backbone, with the optimal hyperparameters being used for all other backbones. Results are reported on a held out test set and averaged over 4 training runs from different random seeds for the ResMLP models.

4.7.2 Object Recognition on Imagenet-100

Uniform vs. Foveated Sensing: Table 4.1 shows the classification accuracy on the ImageNet-100 test set for low resolution (49 patch) and high resolution (196 patch) backbones and different sensing methods. We show that foveated models with fixation mechanisms consistently outperform the uniform models with no fixation mechanism. Importantly, this performance improvement could be attributed to the ability to fixate, particularly as these models are not translation invariant. We report further results on uniform models with fixation mechanisms. We show that the fixation mechanism improved classification accuracy for ResMLP models by 0.4% and 1.4% for the 49 and 196 patches respectively. Conversely, classification accuracy decreased for ViT models when incorporating a fixation mechanism. Ultimately, models that used both

Backbone / Method	ResMLP-49		ResMLP-196		ViT-49		ViT-196	
	Acc@1	GFLOPs	Acc@1	GFLOPs	Acc@1	GFLOPs	Acc@1	GFLOPs
Baseline	70.2 ± 1.2	0.70	76.3 ± 0.3	2.95	67.9 ± 0.4	1.26	77.2 ± 0.5	5.56
Uniform *	70.6 ± 0.4	0.82	77.7 ± 0.8	3.07	67.7 ± 0.5	1.38	77.0 ± 0.2	5.68
Learnable Crop *	70.4 ± 0.6	0.82	77.8 ± 0.3	3.07	-	1.38	-	5.68
Foveated Sensor *	72.2 ± 0.8	0.82	79.6 ± 0.2	3.07	70.3 ± 0.7	1.38	77.3 ± 0.2	5.68
Learnable Sensor *	72.3 ± 0.6	0.82	79.2 ± 0.2	3.07	69.8 ± 1.0	1.38	76.8 ± 0.7	5.68
Learning to Zoom *	72.9 ± 0.4	0.82	79.9 ± 0.8	3.07	70.1 ± 0.5	1.38	76.4 ± 0.7	5.68

Table 4.1: Top-1 accuracy (Mean and SD over 3 random seeds) on Imagenet100 under a variety of different image sampling methods and vision backbones. ResMLP-49 indicates a 49 patch resmlp, with other model names analogously defined. Methods with '*' indicate that a separate network adapts how the image is sampled (e.g. a localisation network). Baseline refers to backbones in their conventional form i.e. no fixation mechanism.

foveated sampling and a fixation mechanism performed better than their uniform counterparts. In summary, we find that foveated sensing often allows these non-convolutional models to better classify the Imagenet-100 dataset.

Our results further support the findings in the previous chapters where we showed it was generally favourable to have smaller higher resolution fovea for models operating on fewer number of pixels. We found the optimal Fovea radius for the 49-patch models to be 30% while 50% was found to be better for the 196 patch models. An interesting observation was that the fixation mechanism was found to help uniform ResMLP models but not uniform ViT models. We were unable to ascertain why this was the case. We can speculate that while these models are not translation invariant, they can learn to be (approximately) and there is no strict reason why they should benefit from this behaviour.

Foveated Sensing vs. Learning to Zoom: We repeat experiments with the Learning-to-Zoom method presented by Recasens et al [44] in a non-convolutional context. We find that under the optimal foveated sensor parameterisations, Learning-to-Zoom marginally outperformed foveated sensing methods. For the ResMLP variants, independent t-tests did not show this improvement to be statistically significant (p=0.25 and p=0.57 for 49 patch and 196 patch models respectively). For ViT models, we found that foveated approach marginally outperformed Learning-to-Zoom in the case of 196 patch model. However, while more significant, an independent t-test assigns a p-value of 0.1, and does not meet the standard threshold of 0.05 for statistical significance. Ultimately, these results would suggest that foveated sensing and Learning-to-Zoom are approximately as effective as each other in improving the accuracy of these non-convolutional systems.

This contrasts our previous study on foveated graph convolutional networks, in which we found Learning-to-Zoom to outperform foveated methods. While it is hard to ascertain exactly why this is the case, we can speculate on a few potential reasons. Firstly, in both studies the

	Fixed Sensor					Learned Sensor
Fovea Radius	0.1	0.3	0.5	0.7	0.9	n/a
49 Patch	71.54	72.2	72.1	71.4	70.9	72.3 (+1.2) (+0.1)
196 Patch	78.9	79.0	79.6	78.7	78.3	79.2 (+0.9) (-0.4)

Table 4.2: ImageNet-100 classification accuracy with 49 patch and 196 patch ResMLP-S12 models. We consider two variants of each model, one with a fixed foveated sensor in which we vary the parameterisation by adjusting the radius of the fovea and a learned sensor in which the sampling density over the visual field is optimized jointly with the network weights. For the learned sensor we show the absolute difference in accuracy between the worst (left) and best (right) performing fixed sensor parameterisations.

ImageNet-100 dataset was used suggesting that this difference in observations is due to the architectures. In this chapter, the classification networks are identical for all methods, whereas in the graph convolution study, our foveated architecture used graph convolutions and the Learning-to-Zoom method used ordinary convolution. This could suggest that the graph convolution, or its hyperparameterisation, was still suboptimal for operating on foveated images. Alternatively, it may be possible that weight sharing (in terms of spatial convolution) on foveated images is an overly restrictive inductive bias and as such ResMLP and ViT are more suited to processing foveated image data. In a similar vein, this may indicate that our choice of coordinate frame is still suboptimal in our graph convolution layer. As discussed, ResMLP and ViT learn spatial structure from data and may arrive at better solutions than our hand designed method.

4.7.3 Learnable Sampling Layout

As observed in the previous chapter, the sampling layout can have a significant impact on the performance of the network, however hyperparameter search is inconvenient, particularly in high data settings and large architectures. In this chapter we introduced a learnable sensor in which the sampling layout is learned via gradient descent jointly with neural-network weights and circumvents the need for this costly tuning.

In Table 4.2 we show the results of a sweep over the fovea radius of our fixed foveated sensor for two ResMLP architectures of different spatial resolutions and compare this to the learned sensor. We show that for both models, our learned sensor can outperform the worst parameterisation of the sensor by 0.9% for the 196 patch model and 1.2% for the 49 patch model. We found that the performance of the learned sensor was comparable to that of the optimal found by hyperparameter search.

To further support the claim that the learnable sensor is converging on an optimal layout we compare the learned sampling layout with the fixed layouts. Recall that for both sensors, the coordinates of each patch centre are dictated by a mapping function $\zeta(\cdot)$ that maps the radial

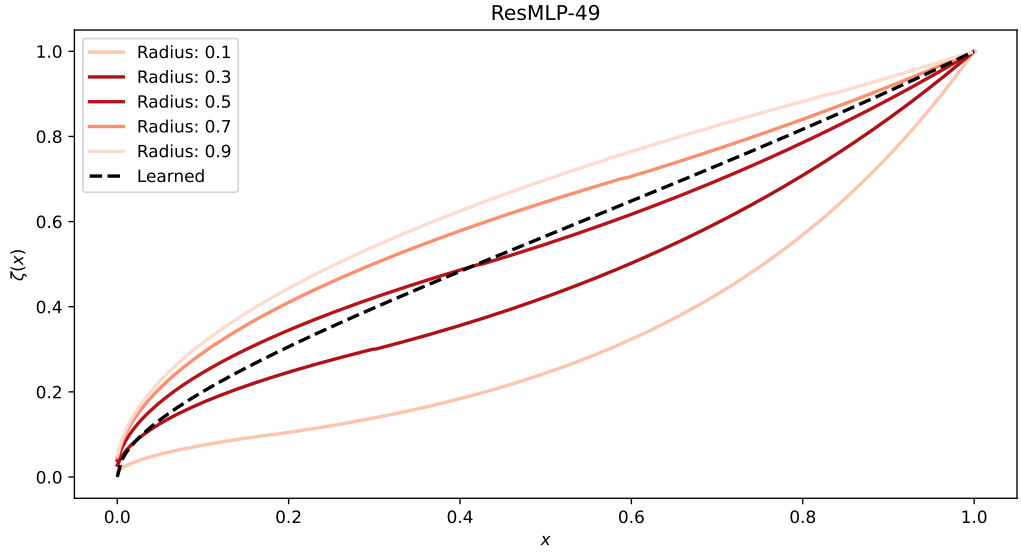

Figure 4.3: Graph of ζ_x for different fovea radii parameterisations of the Sunflower Sensor 3.5 (solid lines) and a learned parameterisation represented by a polynomial (dashed line). Darker lines represent better performing parameterisations on Imagenet100 and 49 Patch ResMLP.

coordinate of a uniform sensor to a new radial coordinate. We plot the ζ functions for different fixed sensor parameterisations as well as the learned sensors in Figures 4.3 and 4.4.

We show that the learned ζ function has a high correlation, particularly in the foveal regions, with the optimal ζ functions found by hyperparameter search. When moving to the peripheral regions the correlation begins to diverge with the learned ζ functions favouring a more linear function in these regions. Nonetheless the learned functions resemble the optimal fixed sensor parameterisation most closely suggesting that it is possible to learn the sampling layout jointly with the neural network and mitigate the reliance on hyperparameter search. Importantly, the fixed sensor ζ function does not cover the full space of possible sampling layouts and it is possible that the optimal layout is not possible in its current formulation. In contrast, the learned sensor uses a polynomial and can comparatively represent a much larger range of possible sampling layouts, particularly as the order of the polynomial increases.

4.7.4 To Crop or Not to Crop

A question one may have with regard to foveated sampling is if its benefits towards visual perception could similarly be realized by a crop. Note that both our fixed formulation and our learned formulation guarantees that sampling covers the full visual field. I.e. it extends to the borders of the pre-sampled image. As such there is no parameterisation of either method that could yield crop like behaviour. To verify that the best parameterisations found either through hyperparameter tuning or through gradient descent are not simply an approximation to an opti-

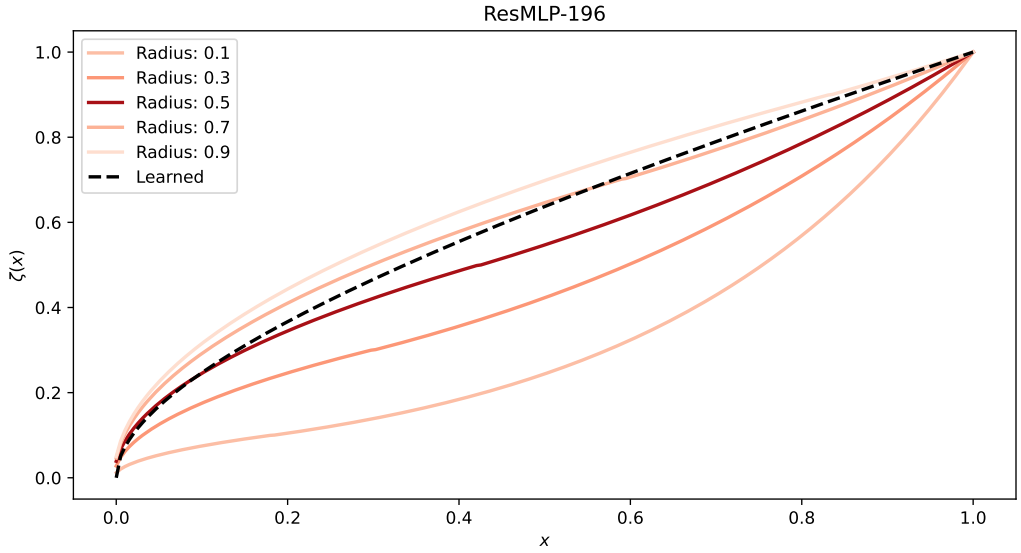

Figure 4.4: Graph of ζ_x for different fovea radii parameterisations of the Sunflower Sensor 3.5 (solid lines) and a learned parameterisation represented by a polynomial (dashed line). Darker lines represent better performing parameterisations on Imagenet100 and 196 Patch ResMLP.

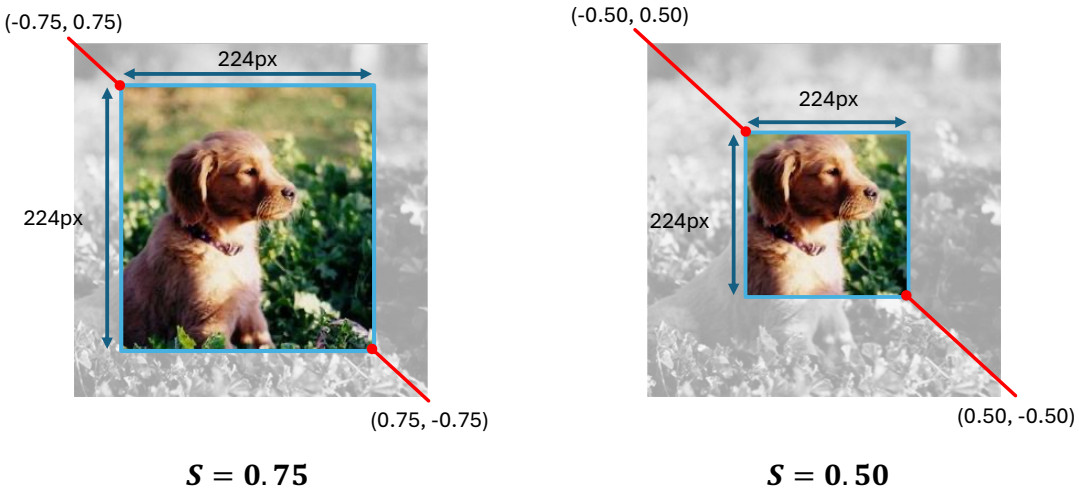

Figure 4.5: Illustration of the region of an input image that is sampled by the learnable crop sensor for different values of S. Note that the cropped region is always resampled to a consistent size (e.g. 224×224). As such, a lower value of S yields a higher resolution sampling of the input image.

mal crop we conduct experiments with a uniform sensor and fixation mechanism and introduce a learnable parameter S that controls the spatial extent of the sampling grid. As S decreases the crop becomes tighter resulting in a higher-resolution sampling of the pre-sampled image (Figure 4.5). We refer to this as the learnable crop method.

As shown in Table 4.1, the learnable crop method does not outperform foveated methods and in fact performs worse than the baseline models with no fixation mechanism in the case of ResMLP-49. Inspecting the parameterisation of the grid over the course of training showed that S would decrease from its initial value of 1.0 resulting in tighter crops, before increasing and converging back to values around 1.0 by the end of training. This suggests that the network found it was generally more optimal to sample the full visual field than only part of the visual field at higher resolution. We speculate that the decrease in performance may be due to the changing sampling grid over the course of training may and a shift in the statistics of the patch embeddings over the course of training. In turn this may necessitate a longer training time for subsequent network weights to converge.

This speculation would similarly hold for the learnable sensor proposed in this chapter, however we show a significant performance improvement over the learnable crop methods for both ResMLP models by approximately 2%. This suggests that foveated sampling may indeed represent a more powerful way to sample visual information than is possible with a uniform arrangement of pixels and that foveation is not simply an approximation of an optimal crop of the input data.

4.8 MNIST Experiments

So far we have shown that foveated sampling can increase the classification accuracy of object recognition systems. However, we have also seen that it is important to use the right amount of foveation in order to achieve good performance and a compelling explanation for the optimal choice remains elusive. In this section we attempt to elucidate some causal factors that explain when foveation is useful. Specifically, in this work we look at the scale distribution of objects in the dataset as a potential driving factor that informs the optimal amount of foveation. For the purposes of this we use MNIST [54] to derive a highly controlled setting to control the scale distribution of objects.

For all networks we use a 3 hidden layer MLP with 128 hidden neurons in each layer and a linear classifier on the end. For foveated sampling we use the foveated patch sensor with 196 patches and average pool each patch value rather than projecting to a higher-dimensionality with a linear layer. Again, the extreme dimensionality reduction of the input data through the sensor poses challenges for classifying very small digits. All networks were trained for 8 epochs, at a learning rate of 0.1 and batch size of 512 and no weight decay. OneCycle [128] learning rate

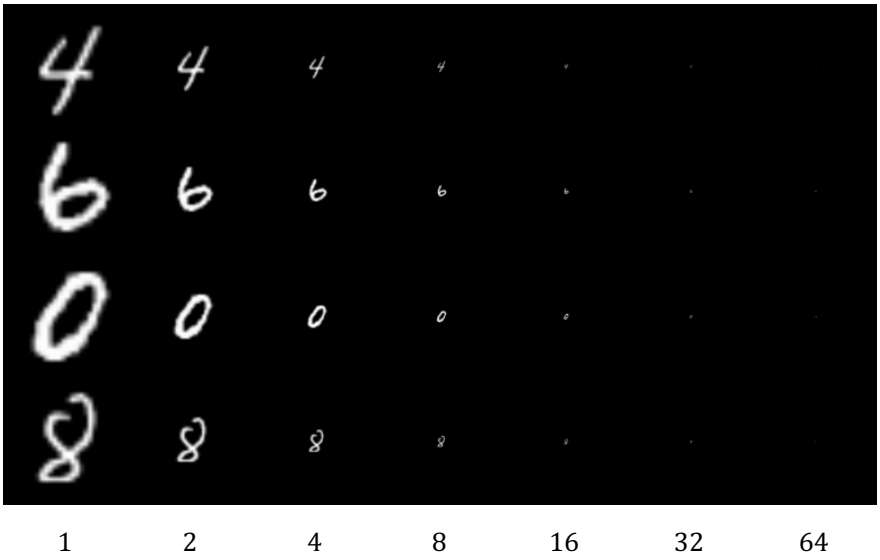

Figure 4.6: Visualisation of MNIST digits with different values for M. As M increases, digits take up a smaller percentage of the full image.

schedule is used with a warm-up period of 1 epoch.

We train foveated classifiers at four different fovea radii (0%, 20%, 50%, 100%), where 0% and 100% correspond to extreme foveation and no foveation respectively. We consider 6 variations of Scaled-MNIST where the digits are randomly scaled between a range of 1 and an upper bound M where M is either (2, 4, 8, 16, 32 or 64). The higher the scale value the smaller percentage of the visual field the digit takes up (Figure 4.6).

In Figure 4.7 we plot classification accuracy as a function of object scale. Specifically, the unscaled digit is said to have a spatial extent of -1 to 1 in both the x and y axes. The scale factor is how many times smaller the spatial extent of the digit is in these axes, i.e. the new spatial extent is given in image coordinates as $\pm$ 1/ (scale factor). For example a scale factor of 2 refers to a spatial extent of -0.5 to 0.5.

Broadly, we show that a foveated sampling is significantly more robust to a high dynamic range of scale variation within the dataset. Considering subfigure F as an example. The foveated sensor with a 0% radius is the only method able to reliably classify digits across the entire scale range. Note that in the case of 0% radius this is similar to a pure log-polar transform with oversampling at the origin. As expected, all methods can reliably classify large objects, however even at 20% radius this accuracy drops off dramatically as objects become smaller and smaller, and for > 20% radius foveae this degrades into no better than random guessing for extremely small digits. While scale equivariance is a guarantee for log-polar images, leveraging this property directly requires a translation equivariant vision system such as a CNN. In these experiments we use an MLP, meaning the ability to classify objects over this scale range is a purely learned behaviour,

and not because of any specific inductive bias.

From subfigure A we can see that the foveated strategy is not better in all cases. With minor scale variation with a scale factor range of 1 - 2, we show that a 0% radius fovea performs worse than all other instantiations in classifying digits, albeit only marginally. A simple explanation for this is that with a fixed pixel budget, increasing the resolution of the fovea, decreases the resolution of the periphery. As such for large objects, the reduction in sampling resolution in the periphery hinders classification accuracy. This provides some explanations as to why in previous studies extreme amounts of foveation were found to hinder performance. Comparing each graph shows that the optimal amount of foveation will be in part dependent on the scale distribution of the objects in dataset. We can also comment on potential train test divides that may influence the optimal amount of foveation. Referring to subfigure C, we can see that over the full scale range, the 0% radius fovea is a good choice for classifying most objects in this scale range well. However, if the test set only contained objects in a scale range of 1-2, we would still favour larger fovea radii, despite being trained on a dataset with higher scale variation.

Some interesting observations can be made on the general shape of the curves. In subfigure A we can see a conspicuous drop off at the limits of each scale range, i.e scales of 1.0 or 2.0. We see that the decline in accuracy at a scale of 1.0 becomes more pronounced as the scale distribution increases for all methods. For example in subfigure A, the accuracy at this range is approximately 97% for all methods, however in subfigure F it is approximately 80%. Similarly we see the same drop off at the end of the spectrum, however this begins to be dominated by the inability to resolve small digits for the methods with larger fovea radii. However, for the 0% radius method this drop still seems to occur albeit slightly. Noticeably, these drops in accuracy at the limits of the scale distribution seem to be invariant to the range of this scale distribution. Concretely, we can hypothesise that regardless of the scale distribution, the classification rate will be lower at the limits of the scale range. Unfortunately we could not isolate the exact cause of this observation and leave this for future work.

4.9 Limitations

Similar to the previous chapter, our experiments are limited in that only the ImageNet100 dataset is considered. To further substantiate the claims made in this chapter, evaluation on other datasets are required, however this was infeasible due to time and computing resources. An additional detail in the context of architectures such as ViT and ResMLP, is that owing to having fewer image specific inductive biases than CNNs, they are comparatively data hungry and are prone to overfitting on smaller datasets. As such, it would be useful to repeat this experimental procedure on far larger datasets as these architectures have been shown to excel in this large data regime.

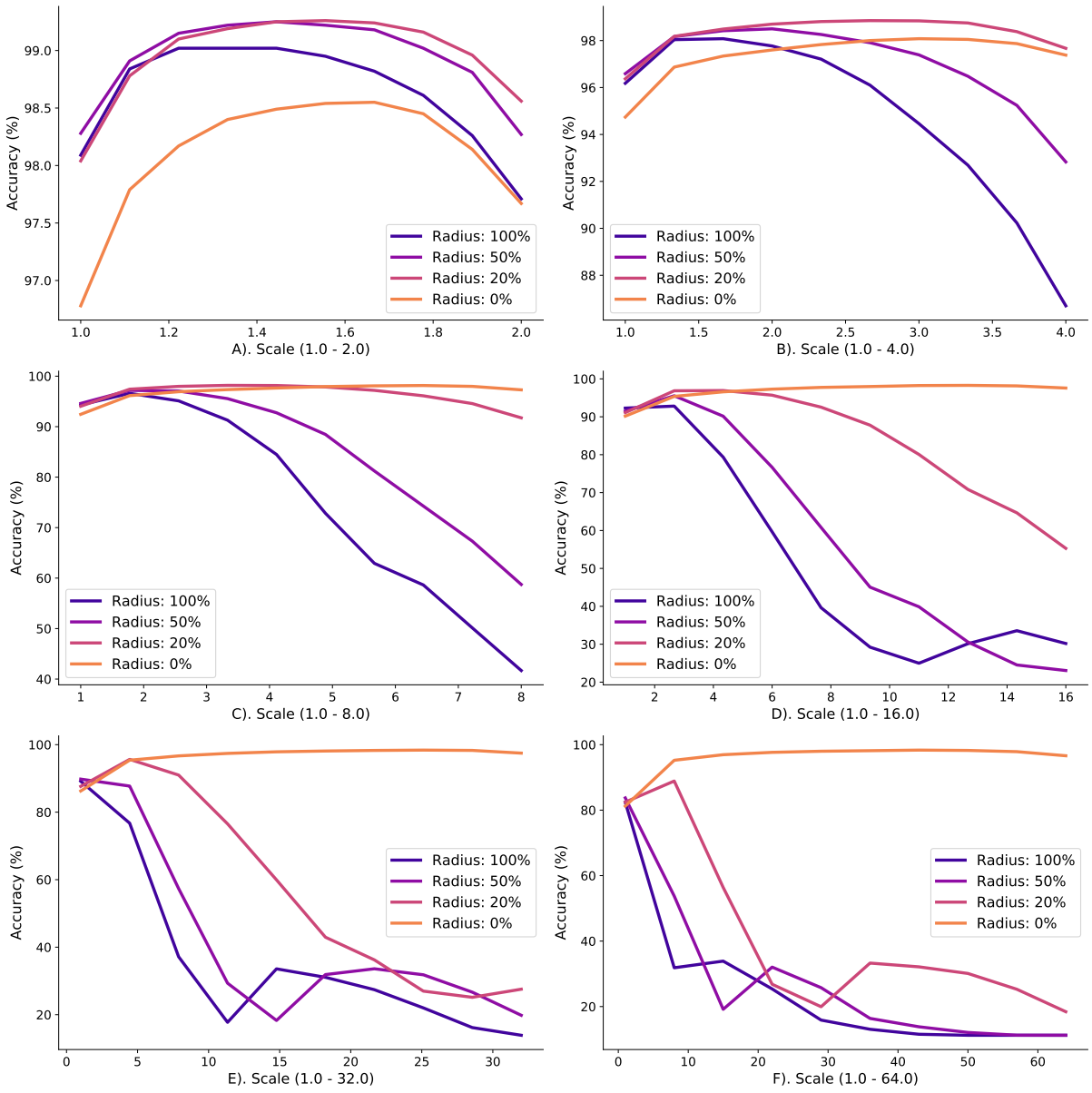

Figure 4.7: Classification Accuracy of Foveated Neural Networks with different fovea radii on the scaled MNIST dataset. Each subfigure plots the classification accuracy at different scales, with each subfigure representing a different scale distribution (or scale range). Characterized by an approximately flat response in classification accuracy, we can show that foveation is helpful in classifying objects that exhibit a high dynamic range of scale variation.

Our MNIST experiments demonstrated the importance of higher resolution foveae as the dynamic range of object scales increases. While MNIST allowed us to synthesize a tightly controlled setting to perform this evaluation, MNIST digits are easily classified by most architectures and do not necessarily confirm their efficacy on more challenge datasets such as those comprised of natural images. Ideally, bringing this controlled setting to more challenging classification tasks would allow for stronger claims to be made with regard to whether this behaviour similarly contributes to the optimal fovea radius parameterisation on datasets such as ImageNet100.

4.10 Conclusion

In this Chapter we explored non-convolutional architectures, namely ViT and ResMLP, for processing foveated images. While they have origins in Natural Language Processing, these architectures have demonstrated great success in vision tasks and set themselves apart from convolutional neural networks in several ways. Firstly, they do not necessitate data exists on a grid-structured domain, and instead operate on sets or sequences of visual tokens. This makes them convenient to apply to foveated images, which do not possess a grid-structured domain in the Cartesian coordinate frame.

Secondly, they do not implicitly assume any spatial structure between visual tokens and instead learn this through data, represented in the form of learned positional embeddings or implicitly through connections between neurons. In the previous chapters we expressed the importance of finding suitable coordinate frames to represent foveated image data, which Convolution layers then use to infer spatial structure of the visual signal. Learning spatial structure via data has the potential to surpass hand designed methods and resonates with Sutton's bitter lesson [129].

Finally, they have the potential to learn space-variant computation over the visual field. There is of course no strict reason why the same computation should be performed for each part of the visual field. This seems particularly intuitive for foveated vision architectures as they have space-variant resolution and translation of visual signals can be addressed through fixating.

Our primary goal of this chapter was to answer Research Question 3, which asks whether a foveated sensing approach is beneficial to these architectures. We proposed a simple method for creating a foveated arrangement of visual tokens that can be used with ViT and ResMLP without specifically modifying them for a foveated context. We conducted experiments on ImageNet100 and showed that such architectures did indeed exhibit improved classification accuracy under this regime. We found that the introduction of a fixation mechanism to these architectures also improved classification accuracy, even with a uniform sensor. We posit that this behaviour is useful for such architectures by improving robustness to translation of objects, which they do not inherently generalize to unlike CNNs.

We revisited the biologically implausible method of learning to zoom (RQ4) and showed that for these architectures they performed on par with a foveated approach, contrasting our previous study with CNNs in which we found Learning-to-Zoom to outperform foveated methods. We hypothesize two potential reasons for this. Firstly, Learning-to-zoom does not use a fixation mechanism to translate the sensor and as such may be less robust to translations. Secondly, this could indicate that our Cartesian-like coordinate frame used in the graph convolution layer could still be improved, and the data driven approach for obtaining spatial structure used in these architectures has converged on a better solution than our hand designed method.

In further experiments we compared the sensitivity of classification accuracy to the sampling layout, by comparing performance across a range of different fovea radii. Again, we observed that lower resolution models favoured higher resolution foveae, and the importance of tuning the fovea radius hyperparameter. Hyperparameter optimisation is costly for these large architectures. In Research Question 6 we asked whether we can learn this parameterisation via back-propagation. We proposed a simple extension of our foveated tokenisation method that described the sampling resolution over the visual field through a learnable polynomial. We showed that this method would converge on a sampling layour similar to the optimal found via hyperparameter search without the need for costly tuning.

In our final experiments we revisited how properties of data influenced the optimal sampling layout of a sensor (RQ5). In particular, we explored the scale distribution of objects. We used MNIST to synthesize several toy datasets where digits were randomly rescaled according to a uniform scale distribution. We showed that as the scale distribution increased, increasingly high resolution foveae were favourable. Conversely, if the scale distribution was small, more uniform sensors were favourable. In experiments with natural images we observed that having too high resolution fovea or too low would decrease classification accuracy. Based on our MNIST experiments, we hypothesize that these observations could be indicative of the underlying scale distribution of objects present in the ImageNet100 dataset.

Chapter 5

Sequential Active Vision Architectures

5.1 Introduction

Thus far we have considered two-stage architectures which first localise salient regions of an image, which a foveated classifier then attends to. An alternative design is that of a sequential approach, in which a single neural network jointly extracts image features that can be used for classification, and for predicting where to look in the next time step.

This design is interesting for two reasons. Firstly, the network can be run over many timesteps, accumulating information from multiple viewpoints. We hypothesise that integrating this information across different views may lead to better classification accuracy, as multiple regions can be viewed with foveal resolution. In Ballard's terminology [36], such a system operates at low resolution (in terms of the number of pixels processed in a single timestep) but exhibits a high virtual resolution.

Secondly, it may be possible to share most of the computation between localisation and classification processes, requiring only minimal extra machinery to implement the fixation mechanism, in contrast to a two-stage approach which utilizes a dedicated CNN for localisation. This becomes particularly interesting in application to video data. In such a scenario, it may be possible to predict fixations for the next frame from the current frame, amortizing the cost of fixating so that it incurs very minimal overhead. While we do not perform any experiments on video data in this thesis, we include these ideas as motivation for pursuing such architectures and are promising avenues for future work. Importantly, if we expect such architectures to work on video data, we should equally expect them to work on image data, and so it provides a reasonable framework for investigations into these systems at relatively nascent stages.

In this chapter we explore sequential systems applied to ImageNet100 and Imagewoof. Ultimately, we aim to show whether sequential architectures derive improvements from attending

to images multiple times, rather than the single-shot approach of two-stage architectures, and what additional mechanisms are needed to achieve this (RQ7). We perform several experiments with various architectures, exploring design considerations for sequential systems. Firstly, we present a simple sequential architecture based on the foveated CNNs analysed in Chapter 3. The architecture comprises a single foveated CNN and predicts fixation for the next time step by applying the attention module presented in section 3.3 to the final convolutional feature maps. Observations are integrated into a single prediction by averaging predictions over all time steps. We perform experiments analysing the ability of such networks to classify the ImageNet100 dataset.

An important mechanism for implementing sequential systems is some form of memory to store previous observations. We present further experiments where we compare more expressive mechanisms, namely self-attention [113] and Legendre Memory Units (LMUs) [130] to ascertain whether they facilitate better classification accuracy.

Furthermore, we examine the behaviour, in terms of classification accuracy, of the aforementioned system as the number of times it is allowed to attend to an Image is increased. Additionally, we revisit Research Question 5 and explore whether the optimal parameterisation of a foveated sensor, in terms of fovea radius, is dependent on the number of times the network attends to an image.

Finally, we propose a proof-of-concept sequential architecture for examining the use of memory not just for the final feature representations but for intermediate feature representations as well. The architecture is a reformulation of vision transformers to a sequential form. In this architecture, we forego a foveated sensor and instead treat a visual token (i.e. image patch) as the output of a uniform sensor with a very restricted field of view. We consider two mechanisms for implementing memory, self-attention and LMUs. We adopt data-agnostic policies for attending to images, a raster scan policy and a random one. These policies, in conjunction with the parallel form of each memory mechanism, allow us to efficiently train these systems over many time steps to examine the behaviour in the limit of attending to all locations in a scene.

5.1.1 Chapter Structure

This chapter is structured as follows:

- We first provide the reader with relevant background knowledge of prior works pertaining to memory in neural networks and sequential vision systems.
- We then present a simple sequential foveated CNN and present experiments showing its ability to classify ImageNet100, contrasting against the two stage approach presented in Chapter 3.

- Following this, we present further experiments analysing the performance of the system when utilizing more expressive mechanisms for implementing memory, as well as the behaviour of such a system as it is allowed to make more time steps and with different fovea radii.
- Finally, we present the proof-of-concept architecture based on sequential vision transformers and perform classification experiments on ImageNet100.
- We conclude with a discussion on the limitations and contributions made in this chapter

5.2 Background and Related Work

5.2.1 Memory in Neural Networks

The problem of processing sequential or temporal data frequently requires some form of memory to provide context for future inputs and is a common problem within machine learning. Recurrent neural networks (RNNs) [131] are an early well-known incarnation of a neural mechanism to achieve this and have been applied to a variety of problems, including NLP [132], video understanding [133] and speech recognition [134]. They run on sequences of data and maintain a hidden state to influence the computation at the current time step based on previous time steps. RNNs presented difficulties for training over long sequences as they could not be parallelized and were prone to vanishing gradient problems. Long Short Term Memory (LSTM) RNNs [135] were proposed to alleviate vanishing gradients by introducing gating mechanisms and a memory cell in the neural circuit. This allows LSTMs to selectively forget and remember information and control the flow of information over long sequences. Gated Recurrent Units (GRUs) [136] operate in a similar fashion but make use of fewer gating mechanisms relative to LSTMs. Neural Turing Machines [137] are a differentiable analogue to Turing machines that allow the network to read and write data to a memory store, meaning they can store and retrieve information over long periods of time. Bahdanau et al. [138] proposes an attention mechanism for a bidirectional RNN to selectively introduce parts of the input sequence into the hidden state for machine translation to overcome the bottleneck of encoding long sequences into a fixed-sized vector.

Self-attention [113] can be seen as a successor to the work of Bahdanau et al. [138] and Graves et al. [137]. Self-attention departs from the more conventional RNN approach for sequence problems and instead caches intermediate representations of information processed in previous time steps. Future inputs can attend to these cached representations giving self-attention the possibility of operating over arbitrarily long sequence lengths given enough compute. Self-attention addressed key limitations of recurrent approaches in mitigating gradient flow problems and training parallelism and allowed for significant advances in NLP over recent years. As we

have already discussed, it has permeated through much of deep learning and become a ubiquitous element in many state-of-the-art systems, not just for sequence problems [45]. One key limitation of self-attention with respect to recurrent neural networks is its complexity over long sequences. Running a single time-step inference in an RNN completes in constant time as it uses a fixed-size hidden state during each time step. In contrast, self-attention has an O(n) complexity where n is the number of previous time steps. This is assuming the self-attention mechanism employs causal masking such that it can only attend to previous time steps.

More recently, a family of mechanisms named state-space models (SSMs) aim to reconcile the constant time complexity of RNNs with the training parallelism and ability to operate over long sequences of self-attention [130, 139, 140, 141, 142]. The key difficulty of bringing parallelism to RNNs is the non-linear dynamics of the memory necessitating each time step is computed consecutively. State space models, in contrast, have linear dynamics with regard to memory and a non-linear hidden state. They permit both a parallel formulation and a recurrent formulation, allowing for efficient training and inference, respectively. They have demonstrated impressive performance over long sequences, comparable to that of self-attention but with significantly reduced inference costs. Works have additionally replaced self-attention in vision transformers with SSMs and demonstrated that they can effectively solve vision tasks as well as or even better than vision transformers [143, 144, 145], however, the parallel nature of visual processing means the recurrent form of these architectures is not used.

Other methods for implementing memory in neural mechanisms include Hopfield networks [146]. They are a form of associative memory that allows a full memory to be reconstructed from a subset of the inputs. In their original formulation, Hopfield networks were limited to binary vectors. Recently, Modern Hopfield networks [147] have been shown to work with continuous vectors and have a greater capacity. Hochreiter et al. [148] show that the update rule to retrieve information from Modern Hopfield networks is equivalent to self-attention and additionally show that Hopfield networks can take on several forms to provide associative memory for neural networks, including forms that can operate like RNNs. While Modern Hopfield networks are not explored in this work we mention them as a potential avenue for future research.

5.2.2 Sequential Vision Systems

Many prior works can be described as sequential vision models, both foveated and non-foveated, that observe and act to accumulate new information in service to completing a task.

Many of these works have made use of an Elman RNN [131]. Mnih et al.'s seminal work [28] uses a Multi-FoV crop sensor, processed by an MLP. The MLP features are fed to an RNN that accumulates information into a hidden state and predicts where to look next at each time step. They train the network with Reinforcement learning due to the hard nature of the fixation and

image sampling mechanism. This study was limited in that it was only applied to an augmented MNIST dataset.

Mnih et al.'s work inspired many subsequent works. Ba et al. [149] extend the system to classifying multiple digits in a single image on the SVHN dataset [150]. Sermanet et al. [27] apply the model to the ImageNet-1k dataset [49], and make use of a GoogLeNet [151] instead of an MLP. Interestingly, when the sensor has a field-of-view that spans the full image, they did not see significant improvements in accuracy when attending to the image multiple times. In the case of a small field-of-view, observations improved accuracy to a greater degree.

Li et al. [152] similarly extend Mnih et al.'s work while foregoing a foveated sensor. Instead they make use of an adaptive crop which can zoom in on salient image regions. An interesting addition to Mnih et al.'s work is the use of a dynamic stopping condition, where the network can decide it has enough information to make a decision. They show that they can achieve the accuracy of a network running for 5 time steps (i.e. 5 observations) in 3.6 time steps on average, saving time and computational resources.

Wang et al. propose Adafocus [153], a hard attention model for video understanding. It can be considered a recurrent extension of the two stage localisation classification architectures adopted in works such as spatial transformers and our method in Chapter 3. Each frame is processed by a lightweight CNN to find the most informative region of a frame, which is then cropped and passed to a classifier. Rather than using an RNN, the localiser and classifier use a GRU [136] to leverage temporal information. The network is trained using reinforcement learning, and they show that their method outperforms previous methods, such as 3D-CNNs, in terms of FLOPs and accuracy on the Something Something dataset [154]. Adafocus requires an impractical three-stage pretraining process to make it feasible to train the system with reinforcement learning. In follow-up work, Wang et al. propose AdaFocus V2 [155], which provides an end-to-end training scheme leveraging differentiable sampling as used in [43, 96] and our work. They show improved performance over AdaFocus on a variety of video understanding datasets.

RNNs [130, 131, 135, 136]are characterized by a fixed size hidden state that has an O(1) space complexity with respect to the number of observations. Similarly, they exhibit O(1) time complexity. Such qualities are useful in keeping inference cost low, or at least consistent, when applied to sequential problems over many time steps. However, RNNs present challenging training dynamics, such as vanishing gradients, due to the Non-linear memory state. Self-attention [113] avoids these problems and has been utilized in many recent works concerning sequential vision systems, however it has an O(n) time and space complexity with respect to the number of observations made.

Jonnalagadda et al. [40] apply foveated pooling to the output of a ResNet [156]. Foveal features are maintained in memory allowing future observations to attend to previous foveal features

through self-attention. Attention weights for peripheral features are used to inform where to look next. Previous attention maps (modulated by a decay parameters) are subtracted from the current attention maps to inhibit attention to already attended locations. GliTr, proposed by Rangrej et al [157], is a visual attention model based purely on transformers for video understanding. Each frame is sampled with a restricted field-of-view (i.e. a small crop of the image) reducing computational overhead, and fed to a transformer. The transformer jointly classifies and predicts a fixation for the next frame and stores frame embeddings and uses self-attention to integrate information across frames.

Works under the umbrella of active visual exploration have utilized transformers to sequentially attend to images with a sensor with a restricted field-of-view. Jha et al propose SimGlim [158], which is a pure transformer based encoder-decoder architecture. The encoder operates on a sequence of partial observations (patches of the full image). The decoder aims to reconstruct the full image in an autoencoder fashion. Simultaneously, a fully connected network aims to predict the reconstruction loss for the full image. The region where the reconstruction loss is greatest is used as the next fixation location (i.e. the next patch that should be sampled). Pardyl et al. [159] propose a similar architecture but rather than using reconstruction loss they use Shannon-Entropy on the self-attention weights to find which unsampled patches the network is most uncertain about, circumventing the need for an additional loss. Both these works additionally experiment with a foveated approach based on Multi-FoV crops. Although not in all cases, foveated approaches were often found to perform worse than a simple crop based method.

Olszewski et al. [160] extend on Pardyl et al.'s work and propose a modification to the transformer architecture to make the encoder more computationally efficient. Motivated by mitigating the O(n) complexity of self-attention, they limit self-attention between patches to later stages in the network, and employ independent feed-forward processing per observation in earlier stages. They show that this method can both reduce FLOPs and improve accuracy in classification and reconstruction tasks compared to the method proposed by Pardyl et al. They provide ablation studies that show that decreasing the amount of self-attention in the network does degrade performance, however this only becomes noticeable when most of the self-attention is removed.

An interesting property of the models proposed by Pardyl et al. and Olszewski et al. is that intermediate layers of the feature extractors use self-attention to attend to previous features. In contrast, many of the works discussed here first extract spatial features, and then temporal features. This may be useful behaviour as it allows intermediate features to be contextualized by previous observations. However, this behaviour is expensive, hence Olszewski et al.'s proposal to remove this behaviour in earlier network layers.

The works relating to self-attention store previous observations as a sequence of tokens. A few

works operate on similar principles but store the tokens in a spatial feature map corresponding to the spatial location where the observation was made. In such a scenario the memory size is constant. However, the memory size is typically in relation to the number discrete observations that could be made on an image, and may not be practically more efficient than storing as a sequence of visual tokens.

For example, Seifi et al. present several works where they propose the use of spatial memory maps. Features from observations are stored in these spatial memory maps and features between observations are integrated in various ways such as self-attention or fully connected layers [161] and convolutional layers [162]. Elsayed et al. propose Saccader [163] in an attempt to address the difficulties of training recurrent hard attention models with reinforcement learning such as those relating to Mnih et al.'s work [28]. In parallel, they compute observations across a full input image and store these in a feature map. An attention policy is trained on this feature map to iteratively select observations from this feature map which are passed to a classifier with predictions averaged over time. Importantly, the attention module uses all observations to select the next observation and so is not strictly sequential. They show that their model, unlike RNN approaches, is able to generalize to longer sequences than it was trained on and steadily improves in accuracy as it is allowed to make more observations.

In this chapter we make the following contributions to this area of research. Firstly, we conduct our studies with foveated sensors that have a large field-of-view that spans the entire image when fixating at the centre. Most prior works have adopted small crop sensors or foveated sensors with a very small field-of-view. Two exceptions are Lukanov et al. [21] and Sermanet et al. [27] Our work differentiates itself from these prior works by comparing a wider range of foveated sensors, and being end-to-end differentiable trained only through backpropagating the classification loss. We also contrast the performance of these models against a two-stage approach to contextualize their performance against other methods. Furthermore, no studies have attempted to contrast different methods for implementing memory, we provide experiments for self-attention, LMUs and averaging, mechanisms. We also explore sensor parameterisation in relation to these sequential models. Finally, we explore memory in intermediate layers. Ozlewski et al. [160] and Pardyl et al. [159] propose models that have this behaviour. We make the contribution of exploring the use of LMUs instead of self-attention to achieve the same behaviour in O(1) time complexity.

5.3 A Simple Sequential Model

In this section, we outline a simple sequential architecture based on a foveated CNN (Figure 5.1). Our aim is to explore the simplest possible architecture in the absence of extra machinery and examine its ability to perform image classification tasks in a sequential manner.

It consists of a singular foveated CNN, based on ConvNeXt atto. At time step 0, the network

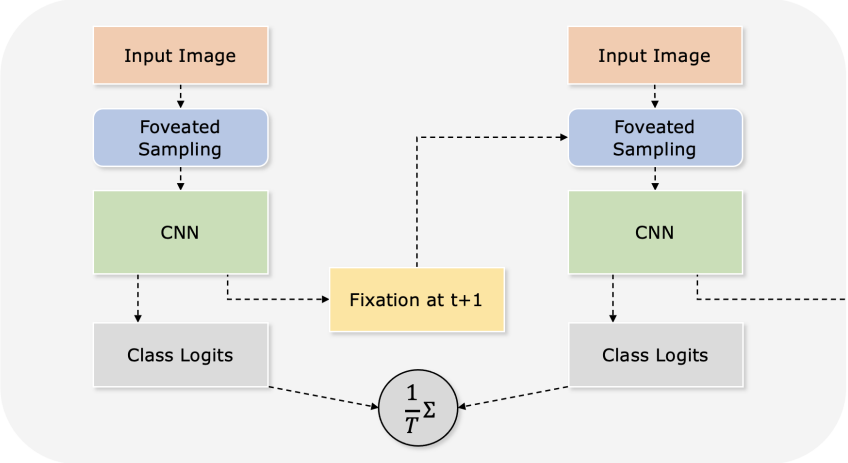

Figure 5.1: Overview of the simple sequential model. At each time step, a foveated CNN makes a class prediction followed and a fixation location for the next time step. This fixation is used to adjust the sensor at the next time step. After T iterations the class predictions are averaged to arrive at a final class prediction.

receives an initial fixation at the centre of the image. The image is then sampled with a foveated sensor at this location and processed by the CNN, producing a set of feature maps. These feature maps are then fed to the attention module as used in the two-stage architecture. To briefly reiterate, a 1x1 convolution is applied to produce a single channel feature map followed by Softmax activation. An elementwise product between each pixel and its corresponding (x, y) location is then computed and summed over, giving the predicted fixation coordinate. This fixation is then used in the next time step to inform the sensor where to centre its gaze. Lukanov et al. describe a similar architecture but instead apply argmax over the final feature maps. Our approach differentiates itself in that it is end-to-end differentiable, facilitating the possibility of adding dedicated network layers for fixation behaviour.

At each time step, the final feature maps (i.e. those that are used as input to the attention module) are also fed to a memory module. For the purposes of our first experiments, we consider a simple mechanism that simply averages over previous observations. Concretely, we apply global average pooling to the feature maps yielding a single feature vector. These are stored in memory over all time steps. The final feature representation is then computed by averaging over these vectors and subsequently fed to a linear classifier. While this is not particularly expensive in terms of memory, an efficient implementation is to apply temporal averaging after the linear classifier to the class logits and maintain a running average.

5.4 Classifying ImageNet100 with Sequential Foveated CNNs

In this section we perform several experiments using our sequential model with different foveated CNN backbones, applied to ImageNet100 and aim to answer two main questions. Firstly, does

Method	Operator	Sensor	# Input Pixels	# Fixations	Params (M)	GFLOPs	Accuracy (%)
ConvNeXt [110]	Conv	Uniform	50176	-	3.7	0.55	78.4 ± 0.5
ConvNeXt	Conv	Uniform	12544	-	3.7	0.20	70.0 ± 0.7
Ours (non-attentive)	Graph Conv	Our Sensor	12544	-	3.7	0.20	72.5 ± 0.4
STN [43]	Conv	Uniform	12544	1	4.8	0.32	72.7 ± 1.0
PTN [19]	Conv	Log-Polar	12800	1	4.8	0.33	70.7 ± 0.6
FCG-STN	Conv	CFG [22]	12544	1	4.8	0.33	71.0 ± 0.4
Fov STN	Conv	Multi-FoV Crops	12800	1	4.8	0.33	71.8 ± 0.5
Fov STN (ours)	Graph Conv	Our Sensor	12544	1	4.8	0.32	74.2 ± 1.1
Learning to Zoom [44]	Conv	Deformable Grid	12544	1	4.8	0.32	75.8 ± 1.4
Sequential	Conv	CFG [22]	12800	2	3.7	0.41	70.2 ± 0.8
Sequential	Conv	Log-Polar	12800	2	3.7	0.41	70.4 ± 0.5
Sequential	Conv	Multi-FoV Crops	12800	2	3.7	0.41	72.8 ± 1.3
Sequential (Ours)	Graph Conv	Our Sensor	12544	2	3.7	0.41	73.8 ± 0.6
Sequential (Ours)	Graph Conv	Our Sensor	12544	3	3.7	0.61	76.5 ± 0.4

Table 5.1: Top-1 Accuracy on the Imagenet100 test set. We split the table into three sections. Top: non-attentive models. Middle: Spatial Transformer like models. Bottom: Sequential Models.

a sequential model improve as it is allowed to attend to an image more times? Secondly, can sequential models perform as well as a two-stage architecture in a similar number of FLOPs, despite not using a dedicated network for fixation behaviour.

5.4.1 Implementation Details

We conduct our main experiments using a graph convolutional ConvNeXt atto as described in Chapter 3. It uses the sunflower foveated sensor described in section 3.5 with 112^2 pixels and a fovea radius of 0.4 and analyse its performance when allowed to attend to an image for 1, 2 and 3 timesteps. Additionally, we perform experiments with sequential foveated CNNs operating on a log-polar sensor, the Cartesian Foveal Geometry sensor, and the Multi-FoV crop sensor operating for 2 time steps and contrast against their corresponding two-stage counterparts. Again, these CNNs are based on the ConvNeXt atto architecture and use sensors with approximately 112^2 pixels. We use the same hyperparameters as used in section 3.7.2 for all architectures. Results are reported on a held out test set, averaged over three separate training runs.

5.4.2 Classification Accuracy vs. Time Steps

We present results on ImageNet100 in Table 5.1. for the foveated graph convolutional sequential model. 1, 2, and 3 time step models achieve classification accuracies of 72.5%, 73.8% and 76.5%. This steady increase in classification accuracy supports the hypothesis that image classification improves when the system is allowed to integrate information from multiple observations when making a final prediction. We speculate that this trend is partly due to viewing multiple regions of the image at Foveal resolution, coinciding with Ballard's idea of these systems having a high internal resolution [36].

An important caveat is that this architecture is Markovian in nature, as the fixation for the next

time step is dependent only on the current observation and maintains no memory of where it has already looked meaning there is the potential for the system to re-attend to the same locations. Nonetheless we observe a dramatic increase in performance from 2 to 3 timesteps (+2.7%). We speculate on two potential reasons for this. Firstly, the model initially fixates on the centre of the image and may not be be sufficiently aligned with objects of interest in order to make a robust classification. The contribution of this initial fixation is comparatively lower in the three step model and could explain this dramatic increase in performance. A second consideration is that viewing the image at similar but nonetheless different fixation locations provides functionality similar to that of test-time augmentation which has shown to improve robustness in image classification [164].

5.4.3 Contrasting Against a Two Stage Approach

We report results in Table 5.1 comparing two-stage foveated CNNs against their sequential counterparts. In general we observe a decrease in classification accuracy when adopting a sequential design. In most cases this decrease is relatively small. For example, in case the sequential graph CNN a 0.4% decrease in classification accuracy is observed. Interestingly, we observe improved performance for the Multi-FoV CNN in a sequential regime. This method is distinct from the other foveated CNNs in that the low resolution region is a uniform downsampling of the full field-of-view. The application of the CNN to this low resolution region is translation equivariant. We speculate that the improved performance of this method, and conversely the decreased performance for the other methods is that this translation equivariant behaviour aids in the ability to localise salient objects. Nonetheless, we still find the sequential graph CNN approach to outperform this method.

While the sequential approaches exhibit lower classification accuracy in general, and a slightly higher FLOP count, the ability to attend to the image is only at the expense of minimal extra parameters from the convolution layer in the attention module (321 parameters). As we discussed in the introduction, these architectures could be particularly interesting when applied to video data, if the fixation process could be distributed across multiple frames, with each frame being attended to only once. We have shown that even this very simple sequential design mostly suffices for achieving comparable performance to an architecture with a dedicated network for attending to scenes on images. Architectures such as AdaFocus [153] have demonstrated the feasibility of this design on video data when using image crops. A promising line of future work is to explore similar systems using foveated sensing.

5.5 Memory in Sequential Foveated CNNs

The sequential architecture covered in the previous section used an averaging mechanism to maintain a memory of previous observations that could then be used for classification. In this section we explore the use of more sophisticated mechanisms for implementing memory. In particular, we consider self-attention [113] and Legendre Memory Units (LMUs) [130]. Self-attention has been a highly effective mechanism for integrating information over long sequences. While powerful, self-attention is computationally expensive as it requires caching previous observations and has a O(n) complexity per timestep where *n* is the number of observations. In line with the goal of foveated sensing to improve computational efficiency, we also consider Legendre Memory Units. LMUs have been shown to have excellent memory capacity in contrast to GRUs [136], LSTMs [135] and other State of the Art RNNs. Unlike self-attention, previous observations aren't cached, but instead maintained in a hidden memory state and operate in O(1) time per observation. We take the sequential model from the previous model, but replace the averaging mechanism with two consecutive 'transformer' blocks, with self-attention or LMUs for combining information between timesteps.

5.5.1 Implementation Details

We use the same foveated graph CNN backbone and experimental set up as in the previous section along with the same hyperparameters. We utilize self-attention in two consecutive transformer blocks with causal masking and a class token. For the self-attention layers, we use multi-head self-attention, with 8 heads and a dimensionality of 40 for the query, value and key projection layers. The feedforward network in the transformer block expands the dimensionality by a factor of 4, applies GELU activation, and then projects the dimensionality back to 320. After the transformer blocks, the class token is fed to a linear classifier. We did not perform any hyperparameter tuning for the self-attention layers.

In the LMU variant, we replace self-attention in the transformer blocks with LMUs. Each LMU uses a hidden size of 320 and a memory size of 256. The memory size corresponds to the number of legendre basis polynomials that the observation history is projected onto. The theta parameter controls the time window, i.e. the number of observations, to be stored in memory. We consider a 3 time step architecture in this case and use a theta value of 3 seconds, where delta t is 1 second. We use the final time step representation after the LMU blocks as input to a linear classifier.

Additionally, for both methods we explore the use of positional encoding to allow the fixation location to be encoded with the observations. We use sinusoidal positional encodings as described by Dosovitskiy et al. [45] to encode a fixation coordinate as a 320 dimensional vector which is added to the image features after global average pooling of the final feature maps. Our

	Averaging	Self-Attention	Legendre Memory Unit
w/ Pos Enc	-	74.9%	75.0%
w/o Pos Enc	**76.5%**	74.7%	75.2%

Table 5.2: Results on the ImageNet100 test set. We omit Positional Encoding for the Averaging method as it has no mechanism to make use of this information.

motivation is to allow the memory layers to incorporate knowledge of the spatial relationships between two different observations.

5.5.2 Results on ImageNet100

In Table 5.2 we show the accuracy of the different methods on ImageNet100. Surprisingly, despite utilizing two additional transformer like blocks, neither the self-attention nor the LMU methods outperformed a simple averaging approach. We found the LMU without positional encoding to perform best, at 75.2% accuracy, out of the more complex mechanisms. In general, the performance difference between the more complex mechanisms is relatively small. The averaging mechanism was substantially better, by approximately 1.5%. We did not observe significant improvements by incorporating positional encodings, and in fact marginally decreased performance for the LMU.

We can speculate on many potential reasons for the above results. Simple explanations such as more hyperparameter tuning, longer training, and the need for more training data may all apply here. An alternative hypothesis is that averaging suffices for combining information in this image classification task. A reasonable avenue for future work is to explore these mechanisms on other visual tasks. One possibility is that of image captioning, which requires the system to reason about multiple parts of a visual scene. For example, Xu et al [165] apply a non-foveated sequential architecture to image captioning tasks and adopt an LSTM for memory. Repeating these experiments in such a setting may better elucidate any potential benefits of these more complex mechanisms.

5.6 Optimal Sensor Layout in a Sequential Context

In this section we investigate the optimal parameterisation of the fovea radius in the Sunflower sensor for a sequential system. There are two reasons why one might expect the optimal sensor parameterisation to change in a sequential context.

Firstly, smaller higher resolution foveae allocate more visual processing resources to a smaller region of the field of view, at the expense of peripheral processing resources. It may be beneficial for these systems to operate over many time steps to increase the area that is sampled at foveal

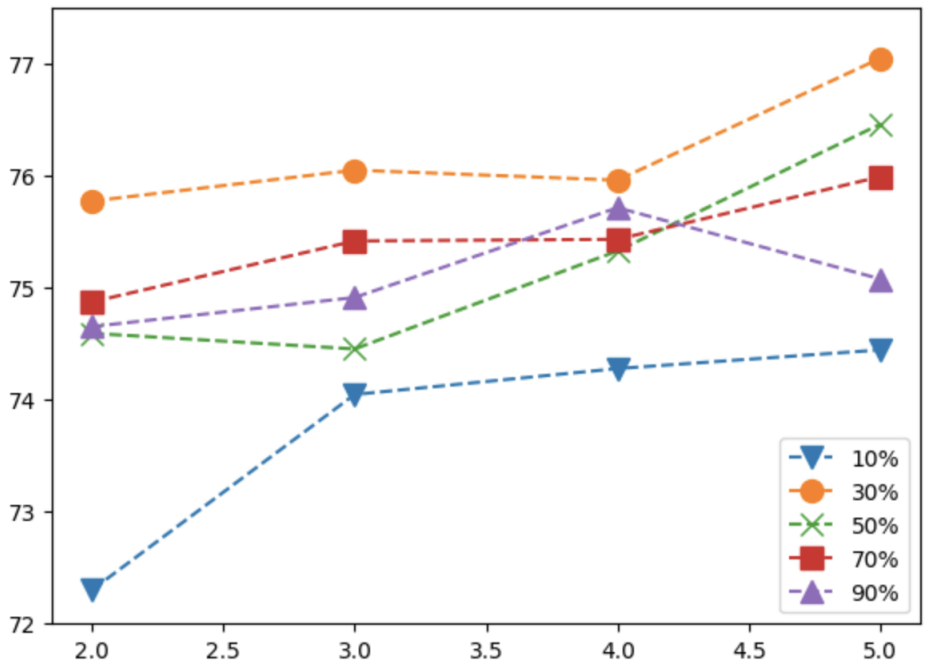

Figure 5.2: Classification accuracy on the Imagewoof test set for different fovea radii and number of timesteps. The optimal fovea radius is largely invariant to the number of time steps the network is allowed to perform.

resolution. Secondly, fixation predictions are made from foveated images (as opposed to uniform ones in the two-stage architecture). It may become increasingly difficult for sensors with lower peripheral resolutions to localise objects in these regions of the field-of-view.

We conduct image classification experiments on Imagewoof[90], and compare how Fovea radii affects classification accuracy of a sequential graph CNN when allowed to attend for a given number of time steps. Beyond this, we also use these experiments to further verify whether multiple observations of an image improve classification accuracy.

5.6.1 Implementation Details

We use the Isometric Graph ConvNext presented in Chapter 2 as the CNN backbone for the model and the Sunflower sensor proposed in section 3.5. We use the simple sequential model design with the averaging mechanism described earlier in this chapter. The training scheme and hyperparameters is the same as in section 3.7.2, except that we use a batch size of 64, and 4 steps of gradient accumulation, to yield an effective batch size of 256. We consider the following range of fovea radii: 0.1, 0.3, 0.5, 0.7 and 0.9. We evaluate architectures that run for up to 5 time steps including the initial fixation at the centre of the image.

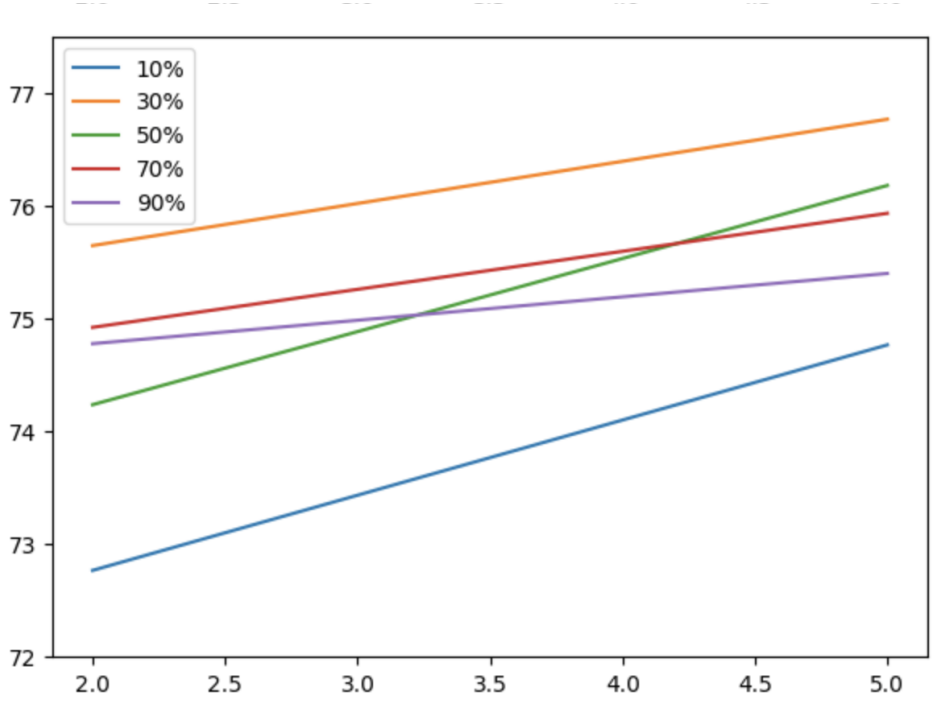

Figure 5.3: Lines of best fit for Figure 5.2. In general steeper lines can be observed for small fovea radii, suggesting that they benefit more greatly from attending to images multiple times.

5.6.2 Results on Imagewoof

Figure 5.2 shows the performance of each run on the Imagewoof test set, each result averaged over three training runs. In general we see, with the exception of some slight variation, the optimal fovea radius is largely invariant to the number of timesteps the network is allowed to perform. Again we find that both a large fovea radius (i.e. 90%) and a very small fovea radius (i.e. 10%) typically exhibit worse performance and find the optimal amount to be 30%. This correlates with the optimal radius found in the previous study which was around 20%.

In Figure 5.3 we show lines of best fit for each radius. We show that smaller fovea radii, in general, have steeper lines of best fit, suggesting they improve more quickly with more fixations than larger fovea radii. This explanation seems intuitive as larger fovea radii result in a more uniform sampling of the visual field. In such cases the sampled visual information differs mainly in translation, but does not acquire much new information as the periphery and the fovea have approximately the same resolution. It is unlikely that these lines of best fit can be extrapolated very far as we should expect diminishing returns on more fixations. Furthermore, running the network for many fixations does incur significant overhead, and as such we are mainly concerned with the behaviour of the network at a low number of timesteps as this is when the network is still computationally efficient. Ultimately, these experiments show that we can largely discount the number of timesteps as a factor for the optimal sensor parameterisation. A limitation of this study was to only compare this trend when using averaging to aggregate information over

timesteps. Future work should assess these trends with other mechanisms such as self-attention and recurrent networks to see if the same observation still holds.

5.7 Memory for Intermediate Layers

One question we can ask about sequential models is where is memory needed. The aforementioned models only integrate information from different fixations at the very end of the network. However, it may be the case that intermediate layers of the feature extractor could benefit from the ability to utilize intermediate representations from previous time steps as part of the calculation for the intermediate representations of the current time step.

We know that to some degree it is useful to have memory in intermediate layers as we can view a vision transformer as a sequential vision model which operates with a restricted field of view (i.e. the size of an image patch). A problem with adopting self-attention in all intermediate layers of a network is that it is computationally expensive. In concurrent work, Olszewski et al. [160] highlight this problem and propose to remove self-attention for attending to tokens from previous time steps from earlier network layers to improve computational efficiency. They show on classification tasks, accuracy is robust to the removal of self-attention in earlier layers, but degrades more rapidly when removed in later layers.

Our work is similar in spirit and complementary to Olszewski et al.'s work, but differentiates itself in that rather than exploring the removal of self-attention, we explore their replacement with Legendre Memory Units (LMUs). Our primary aim is to explore the possibility of a computationally efficient way to implement memory in intermediate network layers. As mentioned previously, LMUs run with constant time complexity, rather than O(n) of self-attention. As such they have the potential to be far more computationally efficient, particularly over long sequences. We propose a toy architecture based on a reformulation of a vision transformer. We conduct experiments on ImageNet100 contrasting classification accuracy when using LMUs or self-attention. We make some omissions in this work. Firstly, we do not use a foveated sensor, instead opting to treat a single visual token as the output of a sensor which performs a small crop of an image. We also do not use a learned fixation policy, as this allows us to exploit the parallelism of both self-attention and LMUs to evaluate the behaviour of both mechanisms over long sequences, which in this case is exhaustively attending to all regions of an image. At the end of this section we propose how these omissions could be reintroduced in future work.

5.7.1 Transformers as Sequential Vision Models

The architectures considered in this section are based on vision transformers. They consist of an initial convolution stage, with non overlapping filters. While the sensor is not foveated in this case, it bears resemblance to foveated sensors as it samples only a small region of the full

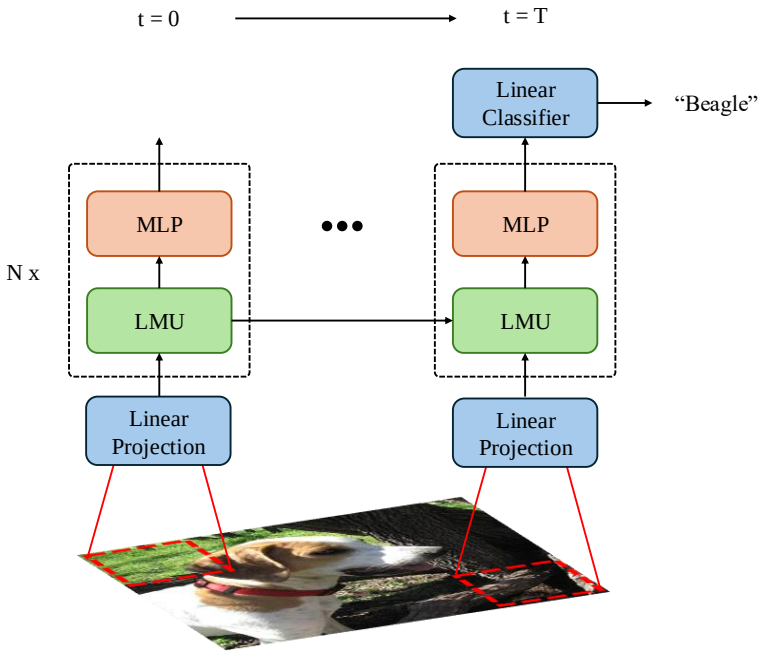

Figure 5.4: Schematic overview of our sequential model based on a reformulation of vision transformers. The red region indicates the small cropped region that is viewed on a given time step. This patch is flattened and linearly projected, before being processed by N transformer blocks ($N = 12$ in our experiments). Intermediate representations from previous time steps are used in future time steps through the form of a memory module (we use an LMU as an example).

image at high resolution. The output of the convolution layers is T visual tokens which are linear projection of image patches. We can interpret this as a sequence of observations after exhaustively attending to all regions of an image with a small crop sensor.

After convolution, a stack of N transformer blocks are then applied to the tokens. A single block consists of a memory layer (either self-attention or an LMU) followed by an MLP, in line with a ViT block [45]. In the case of the self-attention variant, we employ causal masking. This ensures that tokens can only attend to previously seen tokens which is necessary for a sequential interpretation of a vision transformer. Legendre Memory Units implicitly interpret tokens in a sequential manner and do not need Causal Masking to be explicitly applied. We visualize the sequential interpretation of our model in Figure 5.4.

5.7.2 Fixation Policies

We aim to explore the behaviour of each memory mechanism over long sequences, particularly in the limit of exhaustively attending to all regions of the image with small crops. To make this computationally feasible, we can exploit the parallelism of both self-attention and LMUs, and the initial convolution stage, so that all time steps can be computed in parallel. However, the process of making a decision on where to look next from previous observations, as done

by the previous sequential architectures, is a non-linear process, with no obvious parallelizable implementation. This necessitates that we train these architectures sequentially, and cannot leverage their parallel forms. We circumvent this problem by adopting data agnostic fixation policies.

In particular we consider a random policy, and a raster scan policy (Figure 5.5). The output of the convolution stage is a sequence of d-dimensional vectors of length T. In the random policy visual tokens, prior to their processing with transformer blocks, are randomly permuted. This is equivalent to randomly attending to the image. This permutation is applied per sample. In the raster scan policy, the sequence is unchanged, and is arranged left to right, top to bottom.

Furthermore, for the random policy we consider two different positional embedding schemes, a temporal one and a spatial one. In both cases, we learn T positional embeddings that are added to the tokens. In the temporal approach this is applied after permuting the sequence, and allows the positional embeddings to denote the order in which visual tokens are received. In the spatial approach, the embeddings are added prior to permuting the sequence, and denote the spatial position of the visual token. In the raster scan policy we also apply learned positional embeddings.

The random policy is representative of a worst case scenario, in which the fixation policy exhibits no intelligent behaviour nor can any underlying dynamics in the fixation behaviour be leveraged by the network. While the raster scan policy is similarly unintelligent, it does have consistent dynamics in how it traverses an image. It may be possible for the networks to leverage these more predictable dynamics and may be more representative of a learned policy.

5.7.3 Implementation Details

The tokenisation stage is a convolution layer with a filter size of 16x16, a stride of 16 and 384 output channels. This applied to a 224×224 RGB image, yielding 196 tokens ($T = 196$). We use $N = 12$ transformer blocks. For the self-attention variant we use Multi-head self-attention. We use 8 heads, each with a dimensionality of 48. For the LMU, we use a hidden size of 384, and a memory size of 256. Theta controls the sliding window history of the layer, we use a theta of 196 as the sequence length is 196. The MLP in each transformer has a hidden size of 1536 and uses GELU activation. The output size of the MLP has a dimension of 384. After processing all tokens, we take the final token in the sequence and apply a linear classifier to it to get a final class prediction. We adopt the same hyperparameters and training scheme as in section 4.6, and results are reported on a held-out test set, averaged over three training runs.

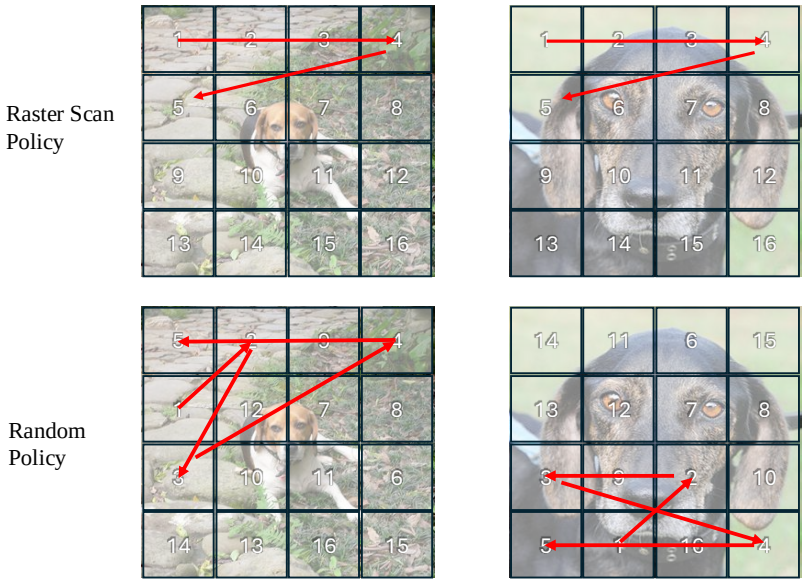

Figure 5.5: Visualisation of fixation patterns for each policy for two different samples. Each square represents a cropped region viewed at a given time step. The raster scan policy scans left to right, top to bottom and is the same for all samples. The random policy randomly attends to the image and is different for each sample.

	Identity	Self-Attention (No Causal Mask)	Self Attention (Causal Mask)	Legendre Memory Unit
Raster Scan Policy	37.7%	74.7%	74.2%	70.9%
Random Policy (Temporal PE)	n/a	n/a	67.4%	66.2%
Random Policy (Spatial PE)	n/a	n/a	72.2%	66.6%

Table 5.3: Classification Accuracy on Imagenet100 test set when using different methods for aggregating visual information from different fixations. Identity refers to the case when no aggregation mechanism is used (i.e. the layer is replaced by an identity transform). We omit results for self-attention with no Causal mask and identity for random policies as they are equivalent to the raster scan policy

5.7.4 Results on ImageNet100

We report results for each method in table 5.3. Interestingly, we find that applying a causal mask in vision transformers does not significantly affect performance, only reducing accuracy by 0.5% and 2.5% for the raster scan policy, and random policy with spatial positional embeddings respectively. The difference in performance between the random policy and the raster scan policy suggests that the latter might naturally favour our dataset. It is likely that the most important tokens for image classification are ones near the centre of the image as that is likely to align with the location of objects in the image owing to camera bias. A raster scan policy might provide these important tokens with enough context to lead to accurate classification which may not always be the case in a random policy.

When using temporal positional embeddings the performance decrease becomes noticeable (-7.3%). It is clear that in such networks it is important to provide information about how tokens are spatially related to one another. This is unsurprising as temporal position embeddings do not provide the network with much meaningful extra information under a random policy. Ultimately, we can show that when framing a sequential network as a causal vision transformer, we can nearly match the performance of a classical vision transformer with no causal mask.

We find that the Legendre Memory Unit network performs comparatively worse than self-attention. Again, the raster scan policy performs best at 70.9% accuracy, with random policies performing approximately 3.5% worse. Nonetheless, these networks still perform far better than the identity method (only 37.7%), which can be considered a bag of visual tokens, suggesting this method is still allowing each token to aggregate meaningful level of information from other tokens to inform classification. Importantly, while these networks performs worse than self-attention, they can be run recurrently, meaning the processing of each new token in a sequence completes in O(1) time. Comparatively, self-attention completes in O(n) time where n is the number of seen tokens. This improvement in complexity may mean that the LMU networks are favourable in practice, especially if run for many timesteps.

We find the LMU to be agnostic to any positional encoding information. As part of its natural formulation, LMUs implicitly interpret tokens as a temporal sequence regardless of any positional encodings. This is unlike self-attention which don't make any implicit assumptions about the relationships between tokens, as fundamentally its an operation applied to a set. The lack of performance gain when providing spatial positional encodings suggests that the LMU does not learn to make use of this information to any reasonable degree and may still fundamentally treat tokens as a temporal sequence which we have shown in the case of self-attention to hamper performance.

While these models are not representative of the foveated models explored previously in this thesis, we have shown that LMUs can be utilized as memory mechanisms in intermediate layers

and perform competitively with self-attention while operating in O(1) time complexity. Recently, many recurrent alternatives to self-attention, such as Mamba [140], have been proposed that may be able to further improve over LMUs. In the following chapter we discuss some potential extensions to this architecture that could be explored in further work.

5.8 Limitations

The primary limitation of this chapter were that for foveated models, we were only able to explore sequential models for a low number of time steps. This was born of computational and time constraints. A difficulty in training these sequential models is that they must be trained sequentially and cannot be parallelized over time. This is due to the fact that future observations are dependent on previous ones. In our final experiments, we did explore these models over many timesteps, but did so without a learned fixation policy. Future work could look to addressing training parallelism of such models or training regimes that can make the training of these models more feasible over many time steps.

With regards to model design, our sequential models were limited in that they did not incorporate inhibition of return and as such run the risk of re-attending to the same portions of the image multiple times. At early stages, we experimented with applying an LMU to the output of the localisation networks attention module, however we were unable to successfully train these models. We speculate that it may be important to utilize memory prior to the computation of a single (x,y) fixation coordinate. This would allow inhibition to take into account visual features, rather than being unaware of them and only having a sequence of (x,y) coordinates to influence the next fixation. We discuss inhibition of return more greatly in the future work section of the following chapter.

5.9 Conclusion

In this Chapter we explored several vision models that operate in a sequential fashion. At a given timestep, the input image is sampled at a certain fixation point. Features are extracted from this sampling and used to make a prediction as to where to look next. Features are integrated over all timesteps into a unified representation that are then used as input to a classifier. Our motivation for exploring such systems was two-fold. Firstly it allows them to attend to images multiple times, increasing their internal resolution which we hypothesised could increase classification accuracy. Secondly, it may be possible to share much of the computation between fixation processes and classification processes. This becomes particularly interesting for future work on their application to video data, as it may be possible to amortize the cost of fixation across frames and consequently incur very minimal extra computation.

We asked whether a simple sequential model, where predictions are averaged over all timesteps, could outperform the two-stage approach from the previous chapters. We found performance of sequential models to be generally lower but still comparable at approximately the same number of FLOPs. Importantly, the sequential model achieves similar performance at the expense of no extra parameters beyond the attention module. In contrast the two-approach requires a dedicated CNN to guide fixations. We speculate on two reasons for reduced performance. Firstly, the reduction in sampling resolution in the periphery may make it challenging for the network to attend objects that exist in this part of the field-of-view. Secondly, using a single CNN for both fixation and classification may require a compromise when learning optimal filters for the corresponding processes. This compromise is not present in the two-stage approach as there is a dedicated network each process.

Our choice of averaging predictions was so that integration of information from different observations could be performed in a parameter free way, subserving fairer comparisons between the two-stage and sequential approach. We further tested whether classification accuracy can be improved over averaging with more sophisticated mechanisms (i.e self-attention and LMUs). Interestingly we found that performance did not improve when switching to these learnable mechanisms, and in fact degraded classification accuracy. We can relate the averaging approach to that of global average pooling in CNNs, which is commonplace in most state-of-the-art CNN architectures. We speculate that on image classification tasks, this paradigm is largely sufficient for integrating spatial information into a single vector for classification. A reasonable line of future work is to reevaluate these mechanisms on tasks that might require better knowledge of spatial structure such as Image Captioning or Segmentation.

Finally, we explored a proof of concept sequential model that used memory in intermediate network layers rather than just at the end. We made the observation that when using causal masks in vision transformers, they can be viewed as sequential vision models that use a small crop sensor and self-attention for memory. Self-attention has an O(n) complexity where n is the number of previous timesteps. The primary aim of this work was to ascertain whether LMUs, which have an O(1) complexity, could replace self-attention as a more computationally efficient alternative. We compared these two mechanisms under two different data agnostic fixation policies (raster scan and random) which exhaustively explore the full image. We found that LMUs performed worse in general (-5.6% in the worst case scenario), but nonetheless significantly improved over having no memory mechanism in intermediate layers. This deficit may be justifiable if the O(1) complexity can provide meaningful computational savings for a given problem. Furthermore, many O(1) self-attention alternatives, such as Mamba, have been proposed in recent years that may close this gap further.

Chapter 6

Conclusion

6.1 Contributions

In this section we will revisit the research questions outlined at the start of this thesis and highlight the main corresponding contributions made in answering them.

1. ***To what degree does the geometric transformation of foveated data to a grid-aligned representation play a role in the accuracy of foveated convolutional neural networks?***

   Previous studies from Torabian et al. [20] and Ozimek et al. [32] showed that foveated CNNs did not outperform uniform CNNs. We analyzed the implications of applying CNNs to foveated images to identify potential pathologies that may explain these observations. Through the lens of geometric deep learning [51] we identified the geometric transform of foveated image data to a grid-aligned representation as a potential factor. Specifically, we argued that these geometric transforms imposed sub-optimal equivariance properties for convolution layers and may yield unhelpful or potentially inhibitory inductive biases for image classification.

   We tested this hypothesis through a novel graph convolution layer, which generalized convolution to non-grid aligned image data, and allowed for coordinate frames to be freely specified. We compared foveated CNNs utilizing a Cartesian-like coordinate frame and a log-polar coordinate frame and showed the latter to yield a significantly more accurate foveated CNN. This supports the hypothesis that the coordinate frame used to represent foveated data plays a crucial role in the classification accuracy of foveated CNNs.

2. ***Can foveated CNNs outperform uniform CNNs in image classification for a given pixel budget?***

   In Chapters 2 and 3, we looked at the application of CNNs to foveated images. Across

a range of different foveated sensors, we showed that in the presence of a fixation mechanism to guide the sensor, foveated CNNs exhibited better classification accuracy than uniform CNNs when operating on the same number of input pixels. This contrasts a similar previous study from Torabian et al. [20]. We speculate that this difference is partly due to our fixation mechanism being optimized directly to solve the task rather than mimicking human fixations.

In many cases, the improvement in classification accuracy relative to a uniform CNN was small. When adopting our graph convolution layer we found this improvement to be significantly more pronounced, performing the best out of all foveated CNNs. This highlights that our graph convolution layer has utility beyond an explorational tool for our first research question and can serve as a promising convolutional operator for processing foveated images in its own right.

3. ***Does adopting foveated vision in non-convolutional architectures such as vision transformers and mixer architectures lead to better classification accuracy than a uniform sensing approach?***

   In Chapter 4, we looked at processing foveated images with non-convolutional architectures, namely ViT and ResMLP. We speculated that these architectures are highly suitable for processing foveated images for two reasons. Firstly, they circumvent the coordinate frame problem associated with CNNs as they do not use this geometric structure to inform how to extract spatial features. Secondly, they can perform space-variant computation, which may resonate with the space-variant nature of foveated images.

   We proposed a novel foveated sensor that extracted a foveated arrangement of visual tokens processed by a Vision Transformer or ResMLP. Like CNNs, we observed better classification accuracy for foveated systems over uniform ones on ImageNet100. Furthermore, we synthesized a toy dataset by randomly rescaling MNIST digits (based on a scale distribution) and placing them on a blank canvas. We showed that as the scale distribution foveated sensing was necessary to accurately classify digits across this scale distribution while a uniform approach failed. Although this toy dataset is not representative of more visually complex scenes, we speculate from these results that the benefit of foveated sensing will become more apparent for visual tasks that require accurately recognizing objects that exhibit a high dynamic range of scale variation.

4. ***Is a foveated sampling strategy more useful for image classification than biologically implausible methods such as learning-to-zoom [44] in terms of classification accuracy?***

   An important consideration is that the emergence of foveated vision in biological vision systems may be a product of biological constraints. These constraints may not apply to

computer vision systems. We contrasted against learning-to-zoom and spatial transformers, two biologically implausible approaches that exhibit similar functionality to foveated sensors in sampling select portions of the visual field at greater resolution. We showed that for CNN-based models, foveated sensing exhibited better classification accuracy than spatial transformers but not learning-to-zoom. For non-convolutional models, we found the performance between the two methods to be approximately the same. While this does not allow for a decisive claim about one approach's superiority to another, it highlights the importance of contextualizing work concerning foveated vision systems with alternative approaches that operate on similar principles.

5. ***Is the optimal parameterisation of a foveated sensor (in terms of classification accuracy of the system) dependent on the architecture, data or both?***

   Using sensors that allowed for the radius and sampling resolution to be controlled through hyperparameters, we performed several experiments analyzing the sensitivity of classification accuracy to these hyperparameters. We showed that both convolutional and non-convolutional architectures that operated on fewer pixels generally favoured smaller and higher resolution foveae. In Chapter 5, we explored the use of sequential vision architectures. We showed that the optimal fovea radius was mostly agnostic to the number of times the network was allowed to attend to an image. Furthermore, we showed that sequential architectures with smaller and higher resolution foveae derived more benefit from attending to an image multiple times.

   A common trend could be observed in the relationship between classification accuracy and fovea radius. Both very small high-resolution foveae (i.e. extreme foveation) and very large fovea (i.e. an approximately uniform sensor) exhibited the worst classification accuracy over the range of possible parameterisations. In Chapter 4 we provided a potential explanation for this trend through our experiments on the Scaled MNIST dataset. We showed that when all digits in the dataset were approximately the same size, lower amounts of foveation were preferred, and excessive foveation degraded performance. Conversely, as the scale variation between digits increased, higher resolution Fovea were necessary to classify digits over the full-scale range, while more uniform sensors could not classify all but the very largest digits. We speculate that the trends observed between fovea radius and classification accuracy in other experiments are partly reflective of the underlying scale distribution.

6. ***Can we learn an optimal parameterisation of a sensor jointly with deep neural network weights through backpropagation?***

   We showed the importance of optimizing the fovea radius parameter to achieve good classification accuracy with foveated vision models. However, this is a costly procedure with

large models and datasets. In Chapter 4, we proposed an extension of our foveated sensor that allowed its sampling layout to be optimized directly with gradient descent by back-propagating classification loss gradients from the classifier. We showed that this sensor converged on a similar layout to one found via hyperparameter search and yielded a similarly accurate classifier. This entirely mitigated the need to tune this parameter through hyperparameter search and saved a significant amount of time as a result.

7. ***For Sequential Vision architectures, does integrating visual information from multiple observations improve classification accuracy and what mechanisms are needed to achieve this?***

   The majority of our work with active vision models used a two-stage architecture with a dedicated CNN to perform fixations. In Chapter 5 we looked at a sequential design where a single foveated neural network jointly classifies and performs fixations, and can perform this behaviour sequentially over many time steps.

   We showed that applying a very lightweight attention module to the final convolutional feature maps of a foveated classifier allowed it to fixate on regions of a scene and achieve a similar performance to the two-stage architecture but with significantly fewer parameters. We showed that, for a simple sequential model that averaged its predictions over all time steps, classification accuracy steadily improved the more times they were allowed to attend to an image. In subsequent experiments, we explored more sophisticated mechanisms beyond averaging, namely self-attention and Legendre Memory Units. Surprisingly, we did not find these to improve performance relative to the averaging approach.

   In our final experiments, we explored the use of self-attention and Legendre Memory Units for implementing memory in intermediate network layers. We drew a comparison that vision transformers, when using a causal mask, can be interpreted as sequential vision models that use a small crop sensor. We created two variations of a vision transformer, one with self-attention and causal masking, and one with Legendre Memory Units for integrating information from previous observations. We showed for the self-attention variant that performance was comparable to that of a conventional vision transformer. We also showed the importance of spatial positional embeddings so that the network could use the spatial relations between observations. We found the Legendre Memory Unit variant to be slightly worse than self-attention but crucially achieves this behaviour in O(1) time complexity rather than self-attention's O(N) complexity. We showed in this scenario that the more sophisticated mechanisms significantly improved the classification accuracy relative to a simpler averaging approach.

6.1.1 Summary

The aim of this thesis was to build and assess a range of foveated computer vision systems and ascertain whether such a strategy can also provide important functional benefits for computer vision systems based on deep neural networks. We have shown through various image classification experiments that adopting foveated sampling in neural networks can improve their ability to classify images. In particular, we found foveated sampling to be increasingly beneficial when the number of pixels in the sensor decreased and when there was greater variation in the scale of objects that needed to be classified. In the following section, we provide several avenues for future work that can extend this line of research.

6.2 Future Work

6.2.1 Image Classification Datasets for Foveated Vision Systems

In this thesis, we used image classification datasets to evaluate the efficacy of different methods. It should be noted that image classification datasets, particularly those such as ImageNet-1k [49], are highly curated in that objects of interest typically occupy a large portion of the total image. Notably, architectures such as CoCa [166] achieve over 90% accuracy on ImageNet-1k at a resolution of 288×288.

If we interpret the 1.2 million ganglion cells in the human vision system as RGB pixels, this is a factor 14 decrease in the number of pixels the system has to process. Yet, CoCa still performs incredibly well on ImageNet. This may indicate that the visual perception problems that biological systems have to solve require significantly higher resolution processing than those required to solve such classification datasets.

A reasonable line of work could simply employ foveated vision in such architectures to reduce their computational footprint as much as possible. It would also be very interesting to explore the application of foveated sensing for tasks that necessitate a uniform resolution that is computationally intractable for architectures such as CoCa.

We alluded to one possible scenario in Chapter 4 in our MNIST experiments. This dataset required classifying digits that exhibited a wide range of scale variation, from very small to very large. This necessitated that the network could resolve fine details to classify small digits and a large field-of-view to classify larger digits accurately. When constraining the number of pixels in the sensor to 784, we found a uniform approach failed while a foveated approach was successful. A limitation of this study was the complexity of the visual perception task, as MNIST can be readily classified to a high degree of accuracy with very simple neural networks.

Accordingly, we suggest that a prudent line of future work is to devise a dataset that exhibits the

same degree of scale variation in objects that need to be classified, but in much more visually complex scenes. Ideally, this dataset would exhibit the same visual complexity as ImageNet and be similarly large in scale. This would no doubt be challenging to collect. A reasonable stepping stone could be through leveraging virtual environments to generate data. For example, Li et al. [167] use Unity3D[1] to generate virtual data for traffic sign detection algorithms used in automated driving systems. This would allow large quantities of visual data to be collected quickly while allowing for control over the scale of objects and their distance from the camera in a semi-realistic visual scene.

6.2.2 Beyond Image Classification

In this thesis, we have used image classification as a representative indicator of the performance of foveated vision systems. In Chapter 5, we explored sequential vision models and speculated on their application to vision problems that were also temporal in nature. The two-stage architectures presented in Chapters 3 and 4 used a dedicated CNN for fixating, while the sequential networks shared most of this behaviour with the classifier and used a lightweight attention module, which had a negligible contribution to the overall amount of computation. While repeatedly attending to a single image with a sequential model is still computationally expensive, on temporal vision problems, it may be possible to distribute this behaviour across frames, processing each once and amortizing the cost of fixating. We suggest a reasonable line of future work is to ascertain whether it is possible to distribute fixations across frames, for which we will suggest a few potential candidate tasks.

The simplest approach is to apply the sequential models to video classification tasks. Many popular benchmark datasets may be suitable for this task, such as Youtube-8m [168], UCF101 [169] and Kinetics [170]. This could also be extended to more challenging tasks such as video captioning and video question answering. Furthermore, this may serve as a fruitful testing ground to re-examine the different memory mechanisms tested in Chapter 5 in a more challenging visual perception task.

An alternative approach is to test these models in environments with an agent that must complete tasks based on visual input. Such an agent must process visual input and make actions, of which attending could be a subset. Such a scenario is very representative of Active Vision in general and potentially much more similar to the environments in which biological systems operate. These environments could be virtual, for example, VizDoom [171] or Deepmind Lab [172], which would provide a convenient testing ground to explore such systems in these early stages. Nonetheless, the same principles could be extended to real-world scenarios. For example, researchers at Deepmind use the ALOHA platform[173], which aims to perform robotic manipulation in various real-world contexts, such as shoelace tying. They used a a CNN and

[1] https://unity.com

diffusion to predict robotic actions from camera input. It seems reasonable that some tasks, such as threading a needle, could benefit from a high-resolution fovea.

6.2.3 Architectural Design and Training Schemes

Feature Extraction From Foveated Images: We considered both convolutional and non-convolutional architectures for processing foveated images. For CNNs we found our graph convolution to be conducive to a more accurate model, however it is contingent on finding a good coordinate frame in which to represent the visual data. It is difficult to know what a good coordinate frame is apriori. Non-convolutional architectures such as Transformers circumvent this problem and learn it from the data. However, unlike CNNs, which typically use small local filters when extracting spatial features, these architectures use a global context, which is computationally expensive and goes against one of the motivating principles of foveated vision.

A reasonable approach to addressing the limitations of each approach is to use a locally connected network. Input and output neurons can be connected via their spatial proximity in Cartesian coordinates, as we performed with our graph convolution layer. Crucially, however, weights are not shared between these connections, unlike convolution. Local connectivity facilitates greater computational efficiency relative to the global context of architectures such as ResMLP. In removing weight sharing, we remove the equivariance inductive prior present in convolution layers and, consequently, the onus of finding a good coordinate frame for representing the visual data.

Inhibition of Return In this work, we adopted a very simple attention mechanism which consists of computing the soft argmax of a feature map and does not incorporate knowledge of where the system has already attended. A well-known phenomenon concerning this behaviour is Inhibition of Return [174]. This decreases the likelihood of attending to a portion of a scene that has recently been attended to and seems intuitive behaviour for such systems in mitigating redundant computation, particularly when operating on static scenes. Several approaches have been proposed, which could be applied to the sequential systems presented in this thesis and evaluated in the same experimental setup.

We would like to highlight that this behaviour may require careful consideration for its implementation depending on the context in which the system is operating. For example, on temporal data, inhibition of return must account for the fact that the scene can evolve over time. Furthermore, it must be able to account for the fact that the viewpoint (beyond the fixation) can change, for example, in the case of an agent operating in a 3D world.

Owing to this apparent complexity, we suggest that the easiest approach to accounting for this behaviour might simply be to leverage sufficiently expressive neural network layers and data and learn this behaviour. A suitable study to verify that this approach is viable is to contrast

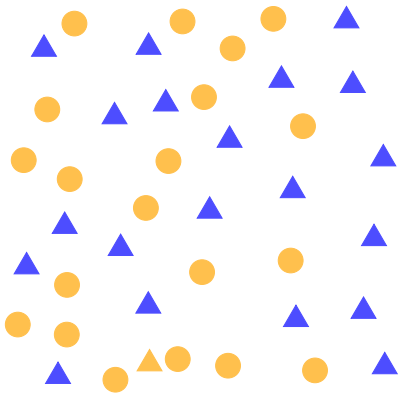

Figure 6.1: A visual search task where the system must find the orange triangle among blue triangles and orange circles.

against such a system with the hand-designed approaches proposed in works such as [40, 162]. Image classification can serve as a testing ground before moving to complex scenarios.

Conditional Fixations: Another interesting extension to fixations mechanisms is to allow it to be conditioned on inputs beyond visual stimuli. Visual question answering would be a useful framework to begin exploring these ideas. An NLP model, such as a transformer, can encode a question. The attention module can condition its behaviour on the output of this model. Cross-modal attention between the hidden states of the attention module and the NLP model could be employed here.

We should expect fixations to be different based on the question asked. For example, if we ask what object is in the bottom left of the image, the fixation mechanism should behave differently than if we ask what object is in the bottom right. It seems intuitive that a general vision system should be able to condition its behaviours on its current goals rather than using the same behaviour irrespective of them.

Training Active Vision Models: A significant challenge for active vision models is training the fixation mechanism, as it is typically hard and not amenable to differentiation. In this work we used differentiable image sampling owing to its convenience. An important detail is that the gradients are highly local as they are the accumulation of subgradients from many bilinear sampling kernels. As such, they lead to poor gradient propagation and may impede learning. Jiang et al. [175] et al. propose linearized multi-sampling to alleviate this problem, but ultimately, it remains unclear how suitable this approach will be across a wider range of tasks.

There are several alternatives to differentiable image sampling that could be used. Reinforcement learning has been frequently used in prior works [27, 28, 163] but comes with many associated challenges, such as training stability and balancing exploration vs exploitation. Other approaches have used heuristics such as Shannon-entropy or argmax of attention maps [40, 159]. However, these do not directly leverage gradient descent to train the fixation mechanism. Both an interesting and important line of future work is to ascertain the limitations of each of these approaches. A reasonable first step in this direction could be to evaluate different methods on their ability to perform visual search tasks, such as finding an object in the presence of other districts (Figure 6.1). This could initially be performed on simple synthetic datasets that can be generated algorithmically and could incorporate conditional attention processes discussed earlier.

6.2.4 Biologically Inspired Models

We did not seek to emulate any particular facets of biological vision systems beyond foveated sampling in this work. Future work could attempt to build a more biologically plausible foveated vision system by incorporating structures known to exist in the visual cortex. It may be the case that the structures and processes found in the visual cortex are specialized specifically for foveated sensing, and might explain why a reductionist approach (e.g. simply applying CNNs to foveated images) does not perform very well. In the following paragraph, we will suggest some initial steps.

Visual cortex layers are known to process a region of the visual field at multiple scales and multiple orientations. Gaussian derivatives have also been used as models of cortical neurons [176, 177]. It is simple to extend our graph convolution layer to a multi-scale and orientation basis. In particular, the steerability of Gaussian derivatives could be used to compute responses at different orientations efficiently. There is no known mechanism for weight sharing in the brain. We previously discussed using locally connected networks for processing foveated images, which do not use weight sharing. We suggest that a biologically plausible model should use an independent set of weights for combining responses to basis filters rather than sharing weights between them.

Hawkins [178] presents a theory that challenges the classical view of how information is processed in the cortex, which could also serve as biological inspiration for foveated neural networks. Briefly, they suggest that cortical columns found in the neocortex all independently build a model of the world. Each column receives input from a region of the field of view as well as sensorimotor input. They suggest that these columns process sequences of visual and sensorimotor inputs. Columns collectively vote on what is being perceived and how to act next.

This bears resemblance to our sequential vision transformer presented in section 5.7. We can

interpret this model as a cortical column. Visual input is a small patch of the input image, while the positional encoding could be considered a sensorimotor input, and this information is processed as a sequence through self-attention or LMUs. An interesting line of work could be to consider many of these models operating in parallel, with their receptive fields arranged in a foveated fashion with voting on perception and actions being performed by applying a transformer to the outputs of the columns, for example.

This is a simplification of Hawkins' proposal, as it omits various additional concepts, such as sparse representations and continuous learning. In the context of computer vision, such a model can be viewed as a combination of our sequential vision transformer and the Olszewski et al.'s model[160], which are well motivated in regard to reducing computational complexity of active vision models. As such, this line of work is interesting in supporting alternative theories of how the brain operates and due to its potential utility in computer vision.

Bibliography

[1] Simon Clippingdale and Roland Wilson. Self-similar neural networks based on a kohonen learning rule. *Neural Networks*, 9(5):747–763, 1996.

[2] Elian Malkin, Arturo Deza, and Tomaso Poggio. Cuda-optimized real-time rendering of a foveated visual system. *arXiv preprint arXiv:2012.08655*, 2020.

[3] Helmut Vogel. A better way to construct the sunflower head. *Bellman Prize in Mathematical Biosciences*, 44:179–189, 1979.

[4] Tom Baden, Thomas Euler, and Philipp Berens. Understanding the retinal basis of vision across species. *Nature Reviews Neuroscience*, 21(1):5–20, 2020.

[5] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

[6] Eric L. Schwartz, Douglas N. Greve, and Giorgio Bonmassar. Space-variant active vision: Definition, overview and examples. *Neural Networks*, 8(7-8):1297–1308, 1995.

[7] Samuel G Solomon. Retinal ganglion cells and the magnocellular, parvocellular, and koniocellular subcortical visual pathways from the eye to the brain. In *Handbook of Clinical Neurology*, volume 178, pages 31–50. Elsevier, 2021.

[8] Eric L Schwartz. Spatial mapping in the primate sensory projection: analytic structure and relevance to perception. *Biological cybernetics*, 25(4):181–194, 1977.

[9] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[10] Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025, 1999.

[11] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[12] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Franziska Geiger, et al. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv*, page 407007, 2018.

[13] V Javier Traver and Alexandre Bernardino. A review of log-polar imaging for visual perception in robotics. *Robotics and Autonomous Systems*, 58(4):378–398, 2010.

[14] Marc Bolduc and Martin D Levine. A real-time foveated sensor with overlapping receptive fields. *Real-Time Imaging*, 3(3):195–212, 1997.

[15] Eric L Schwartz. Cortical anatomy, size invariance, and spatial frequency analysis. *Perception*, 10(4):455–468, 1981.

[16] Patrick Cavanagh. Size invariance: reply to schwartz. *Perception*, 10(4):469–474, 1981.

[17] Jinpyo Kim, Wooekun Jung, Hyungmo Kim, and Jaejin Lee. Cycnn: A rotation invariant cnn using polar mapping and cylindrical convolution layers. *arXiv preprint arXiv:2007.10588*, 2020.

[18] Leendert A Remmelzwaal, Amit Kumar Mishra, and George FR Ellis. Human eye inspired log-polar pre-processing for neural networks. In *2020 International SAUPEC/RobMech/PRASA Conference*, pages 1–6. IEEE, 2020.

[19] Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. Polar transformer networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[20] Parsa Torabian, Ronak Pradeep, Jeff Orchard, and Bryan Tripp. Comparison of foveated downsampling techniques in image recognition. *Journal of Computational Vision and Imaging Systems*, 6(1):1–3, 2020.

[21] Hristofor Lukanov, Peter König, and Gordon Pipa. Biologically inspired deep learning model for efficient foveal-peripheral vision. *Frontiers in Computational Neuroscience*, 15:746204, 2021.

[22] José Martínez and Leopoldo Altamirano Robles. A new foveal cartesian geometry approach used for object tracking. *SPPRA*, 6:133–139, 2006.

[23] Binxu Wang, David Mayo, Arturo Deza, Andrei Barbu, and Colin Conwell. On the use of cortical magnification and saccades as biological proxies for data augmentation. *arXiv preprint arXiv:2112.07173*, 2021.

[24] Peter D Scott and Cesar Bandera. Hierarchical multiresolution data structures and algorithms for foveal vision systems. In *1990 IEEE International Conference on Systems, Man, and Cybernetics Conference Proceedings*, pages 832–834. IEEE, 1990.

[25] Cesar Bandera. *Foveal machine vision systems*. State University of New York at Buffalo, 1990.

[26] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[27] Pierre Sermanet, Andrea Frome, and Esteban Real. Attention for fine-grained categorization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.

[28] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2204–2212, 2014.

[29] Ethan William Albert Harris, Mahesan Niranjan, and Jonathon Hare. Foveated convolutions: improving spatial transformer networks by modelling the retina. In *Shared Visual Representations in Human and Machine Intelligence: 2019 NeurIPS Workshop*, 2019.

[30] Sumitha Balasuriya. *A computational model of space-variant vision based on a self-organised artificial retina tessellation*. PhD thesis, University of Glasgow, 2006.

[31] Masaki Nakada, Tao Zhou, Honglin Chen, Tomer Weiss, and Demetri Terzopoulos. Deep learning of biomimetic sensorimotor control for biomechanical human animation. *ACM Transactions on Graphics (TOG)*, 37(4):1–15, 2018.

[32] Piotr Ozimek, Nina Hristozova, Lorinc Balog, and Jan Paul Siebert. A space-variant visual pathway model for data efficient deep learning. *Frontiers in cellular neuroscience*, 13:36, 2019.

[33] Nina Hristozova, Piotr Ozimek, and Jan Paul Siebert. Efficient egocentric visual perception combining eye-tracking, a software retina and deep learning. *arXiv preprint arXiv:1809.01633*, 2018.

[34] John I Yellott Jr. Spectral consequences of photoreceptor sampling in the rhesus retina. *Science*, 221(4608):382–385, 1983.

[35] Yiannis Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. In *Proc. of first ICCV*, pages 552–573, 1987.

[36] Dana H Ballard. Animate vision. *Artificial Intelligence*, 48(1):57–86, 1991.

[37] John K Tsotsos, Scan M Culhane, Winky Yan Kei Wai, Yuzhong Lai, Neal Davis, and Fernando Nuflo. Modeling visual attention via selective tuning. *Artificial intelligence*, 78(1-2):507–545, 1995.

[38] Bilal Alsallakh, Narine Kokhlikyan, Vivek Miglani, Jun Yuan, and Orion Reblitz-Richardson. Mind the pad–cnns can develop blind spots. *arXiv preprint arXiv:2010.02178*, 2020.

[39] Bilal Alsallakh, Vivek Miglani, Narine Kokhlikyan, David Adkins, and Orion Reblitz-Richardson. Are convolutional networks inherently foveated? In *SVRHM 2021 Workshop@ NeurIPS*, 2021.

[40] Aditya Jonnalagadda, William Yang Wang, B.S. Manjunath, and Miguel Eckstein. Foveater: Foveated transformer for image classification, 2023.

[41] Chittesh Thavamani, Mengtian Li, Nicolas Cebron, and Deva Ramanan. Fovea: Foveated image magnification for autonomous navigation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15539–15548, 2021.

[42] Chittesh Thavamani, Mengtian Li, Francesco Ferroni, and Deva Ramanan. Learning to zoom and unzoom. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5086–5095, 2023.

[43] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

[44] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: a saliency-based sampling layer for neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 51–66, 2018.

[45] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[46] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):5314–5321, 2022.

[47] George Killick, Gerardo Aragon-Camarasa, and J. Paul Siebert. Monte-carlo convolutions on foveated images. In Giovanni Maria Farinella, Petia Radeva, and Kadi Bouatouch, editors, *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2022, Volume 4: VISAPP, Online Streaming, February 6-8, 2022*, pages 444–451. SCITEPRESS, 2022.

[48] George Killick, Paul Henderson, Jan Paul Siebert, and Gerardo Aragon-Camarasa. Foveation in the era of deep learning. In *34th British Machine Vision Conference 2023, BMVC 2023, Aberdeen, UK, November 20-24, 2023*, pages 703–706. BMVA Press, 2023.

[49] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[50] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velickovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478, 2021.

[51] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[52] Marta Amorim, Frederico Bortoloti, Patrick Marques Ciarelli, Elias de Oliveira, and Alberto Ferreira de Souza. Analysing rotation-invariance of a log-polar transformation in convolutional neural networks. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2018.

[53] Ghassan Dabane, Laurent U Perrinet, and Emmanuel Daucé. What you see is what you transform: Foveated spatial transformers as a bio-inspired attention mechanism. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[54] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[55] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[56] Matthias Kümmerer, Lucas Theis, and Matthias Bethge. Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet. *arXiv preprint arXiv:1411.1045*, 2014.

[57] Matthias Kümmerer, Thomas SA Wallis, and Matthias Bethge. Deepgaze ii: Reading fixations from deep features trained on object recognition. *arXiv preprint arXiv:1610.01563*, 2016.

[58] Matthias Kümmerer, Matthias Bethge, and Thomas SA Wallis. Deepgaze iii: Modeling free-viewing human scanpaths with deep learning. *Journal of Vision*, 22(5):7–7, 2022.

[59] Hyungsik Jung and Youngrock Oh. Towards better explanations of class activation mapping. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1336–1344, 2021.

[60] Xin Li, Zequn Jie, Wei Wang, Changsong Liu, Jimei Yang, Xiaohui Shen, Zhe Lin, Qiang Chen, Shuicheng Yan, and Jiashi Feng. Foveanet: Perspective-aware urban scene parsing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 784–792, 2017.

[61] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[62] V Javier Traver and Roberto Paredes. Study of convolutional neural networks for global parametric motion estimation on log-polar imagery. *IEEE Access*, 8:149122–149132, 2020.

[63] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019.

[64] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[65] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[66] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[67] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn:

Convolution on x-transformed points. *Advances in neural information processing systems*, 31, 2018.

[68] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3693–3702, 2017.

[69] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2589–2597, 2018.

[70] Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics (TOG)*, 37(6):1–12, 2018.

[71] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 9621–9630, 2019.

[72] Alexandre Boulch. Convpoint: Continuous convolutions for point cloud processing. *Computers & Graphics*, 88:24–34, 2020.

[73] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019.

[74] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3173–3182, 2021.

[75] Richard S Wallace, Ping-Wen Ong, Benjamin B Bederson, and Eric L Schwartz. Space variant image processing. *International Journal of Computer Vision*, 13(1):71–90, 1994.

[76] Eric L Schwartz. Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding. *Vision research*, 20(8):645–669, 1980.

[77] L Sumitha Balasuriya and J Paul Siebert. Hierarchical feature extraction using a self-organised retinal receptive field sampling tessellation. *Neural Information Processing-Letters & Reviews*, 10(4-6):83–95, 2006.

[78] John G Daugman. Two-dimensional spectral analysis of cortical receptive field profiles. *Vision research*, 20(10):847–856, 1980.

[79] John G Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2(7):1160–1169, 1985.

[80] S Marĉelja. Mathematical description of the responses of simple cortical cells. *JOSA*, 70(11):1297–1300, 1980.

[81] Sumitha Balasuriya and Paul Siebert. A biologically inspired computational vision front-end based on a self-organised pseudo-randomly tessellated artificial retina. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 5, pages 3069–3074. IEEE, 2005.

[82] Matteo Tiezzi, Simone Marullo, Alessandro Betti, Enrico Meloni, Lapo Faggi, Marco Gori, and Stefano Melacci. Foveated neural computation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 19–35. Springer, 2022.

[83] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

[84] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[85] Nuri Benbarka, Timon Höfer, Andreas Zell, et al. Seeing implicit neural representations as fourier series. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2041–2050, 2022.

[86] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

[87] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext V2: co-designing and scaling convnets with masked autoencoders. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 16133–16142. IEEE, 2023.

[88] Asher Trockman and J Zico Kolter. Patches are all you need? *arXiv preprint arXiv:2201.09792*, 2022.

[89] Mark Sandler, Jonathan Baccash, Andrey Zhmoginov, and Andrew Howard. Non-discriminative data or weak model? on the relative importance of data and model resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[90] fastai. Imagenette and imagewoof.

[91] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[92] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[93] Samuel G Müller and Frank Hutter. Trivialaugment: Tuning-free yet state-of-the-art data augmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 774–782, 2021.

[94] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[95] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[96] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *International conference on machine learning*, pages 1462–1471. PMLR, 2015.

[97] Jorn-Henrik Jacobsen, Jan Van Gemert, Zhongyu Lou, and Arnold WM Smeulders. Structured receptive fields in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2610–2619, 2016.

[98] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[99] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization, 2015.

[100] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via

gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, October 2019.

[101] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[102] Pola Schwöbel, Frederik Rahbæk Warburg, Martin Jørgensen, Kristoffer Hougaard Madsen, and Søren Hauberg. Probabilistic spatial transformer networks. In *Uncertainty in Artificial Intelligence*, pages 1749–1759. PMLR, 2022.

[103] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

[104] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[105] Matej Ulicny, Vladimir A Krylov, and Rozenn Dahyot. Harmonic convolutional networks based on discrete cosine transform. *Pattern Recognition*, 129:108707, 2022.

[106] Qiang Qiu, Xiuyuan Cheng, Guillermo Sapiro, et al. Dcfnet: Deep neural network with decomposed convolutional filters. In *International Conference on Machine Learning*, pages 4198–4207. PMLR, 2018.

[107] Shangzhen Luan, Chen Chen, Baochang Zhang, Jungong Han, and Jianzhuang Liu. Gabor convolutional networks. *IEEE Transactions on Image Processing*, 27(9):4357–4366, 2018.

[108] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5028–5037, 2017.

[109] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. *Advances in Neural Information Processing Systems*, 31, 2018.

[110] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 11966–11976. IEEE, 2022.

[111] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts.

In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[112] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[113] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[114] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.

[115] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[116] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[117] Xuran Pan, Tianzhu Ye, Zhuofan Xia, Shiji Song, and Gao Huang. Slide-transformer: Hierarchical vision transformer with local self-attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2082–2091, 2023.

[118] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting spatial attention design in vision transformers. *CoRR*, abs/2104.13840, 2021.

[119] Qi Han, Zejia Fan, Qi Dai, Lei Sun, Ming-Ming Cheng, Jiaying Liu, and Jingdong Wang. Demystifying local vision transformer: Sparse connectivity, weight sharing, and dynamic weight. *CoRR*, abs/2106.04263, 2021.

[120] Xiaohan Ding, Chunlong Xia, Xiangyu Zhang, Xiaojie Chu, Jungong Han, and Guiguang Ding. Repmlp: Re-parameterizing convolutions into fully-connected layers for image recognition. *arXiv preprint arXiv:2105.01883*, 2021.

[121] Peng Gao, Jiasen Lu, Hongsheng Li, Roozbeh Mottaghi, and Aniruddha Kembhavi. Container: Context aggregation network. *arXiv preprint arXiv:2106.01401*, 2021.

[122] Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. *Advances in neural information processing systems*, 34:9204–9215, 2021.

[123] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[124] Gamaleldin Elsayed, Prajit Ramachandran, Jonathon Shlens, and Simon Kornblith. Revisiting spatial invariance with low-rank local connectivity. In *International Conference on Machine Learning*, pages 2868–2879. PMLR, 2020.

[125] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[126] Juhong Min, Yucheng Zhao, Chong Luo, and Minsu Cho. Peripheral vision transformer. *Advances in Neural Information Processing Systems*, 35:32097–32111, 2022.

[127] Brian Cheung, Eric Weiss, and Bruno A. Olshausen. Emergence of foveal image sampling from learning to attend in visual scenes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[128] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019.

[129] Richard Sutton. The bitter lesson. *Incomplete Ideas (blog)*, 13(1):38, 2019.

[130] Aaron Voelker, Ivana Kajić, and Chris Eliasmith. Legendre memory units: Continuous-time representation in recurrent neural networks. *Advances in neural information processing systems*, 32, 2019.

[131] Jeffrey L. Elman. Finding structure in time. *Cogn. Sci.*, 14(2):179–211, 1990.

[132] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014.

[133] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini

Venugopalan, Trevor Darrell, and Kate Saenko. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 2625–2634. IEEE Computer Society, 2015.

[134] Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition. *CoRR*, abs/1412.5567, 2014.

[135] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[136] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[137] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[138] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[139] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

[140] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[141] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

[142] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *CoRR*, abs/2111.00396, 2021.

[143] Qihang Fan, Huaibo Huang, Mingrui Chen, Hongmin Liu, and Ran He. Rmt: Retentive networks meet vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5641–5651, 2024.

[144] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang

Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*, 2024.

[145] Yuchen Duan, Weiyun Wang, Zhe Chen, Xizhou Zhu, Lewei Lu, Tong Lu, Yu Qiao, Hongsheng Li, Jifeng Dai, and Wenhai Wang. Vision-rwkv: Efficient and scalable visual perception with rwkv-like architectures. *arXiv preprint arXiv:2403.02308*, 2024.

[146] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[147] Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

[148] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

[149] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[150] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4. Granada, 2011.

[151] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[152] Zhichao Li, Yi Yang, Xiao Liu, Feng Zhou, Shilei Wen, and Wei Xu. Dynamic computational time for visual attention. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1199–1209, 2017.

[153] Yulin Wang, Yang Yue, Yuanze Lin, Haojun Jiang, Zihang Lai, Victor Kulikov, Nikita Orlov, Humphrey Shi, and Gao Huang. Adafocus v2: End-to-end training of spatial dynamic networks for video recognition. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20030–20040. IEEE, 2022.

[154] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter Yianilos, Moritz

Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. *CoRR*, abs/1706.04261, 2017.

[155] Yulin Wang, Yang Yue, Yuanze Lin, Haojun Jiang, Zihang Lai, Victor Kulikov, Nikita Orlov, Humphrey Shi, and Gao Huang. Adafocus V2: end-to-end training of spatial dynamic networks for video recognition. *CoRR*, abs/2112.14238, 2021.

[156] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[157] Samrudhdhi B Rangrej, Kevin J Liang, Tal Hassner, and James J Clark. Glitr: Glimpse transformers with spatiotemporal consistency for online action prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3413–3423, 2023.

[158] Abhishek Jha, Soroush Seifi, and Tinne Tuytelaars. Simglim: Simplifying glimpse based active visual reconstruction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 269–278, 2023.

[159] Adam Pardyl, Grzegorz Rypeść, Grzegorz Kurzejamski, Bartosz Zieliński, and Tomasz Trzciński. Active visual exploration based on attention-map entropy. *arXiv preprint arXiv:2303.06457*, 2023.

[160] Jan Olszewski, Dawid Rymarczyk, Piotr Wójcik, Mateusz Pach, and Bartosz Zieliński. Token recycling for efficient sequential inference with vision transformers. *arXiv preprint arXiv:2311.15335*, 2023.

[161] Soroush Seifi, Abhishek Jha, and Tinne Tuytelaars. Glimpse-attend-and-explore: Self-attention for active visual exploration. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 16117–16126. IEEE, 2021.

[162] Soroush Seifi and Tinne Tuytelaars. Attend and segment: Attention guided active semantic segmentation. In *European Conference on Computer Vision*, pages 305–321. Springer, 2020.

[163] Gamaleldin Elsayed, Simon Kornblith, and Quoc V Le. Saccader: Improving accuracy of hard attention models for vision. *Advances in Neural Information Processing Systems*, 32, 2019.

[164] Masanari Kimura. Understanding test-time augmentation. In *International Conference on Neural Information Processing*, pages 558–569. Springer, 2021.

[165] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2048–2057. JMLR.org, 2015.

[166] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

[167] Xuan Li, Kunfeng Wang, Yonglin Tian, Lan Yan, Fang Deng, and Fei-Yue Wang. The paralleleye dataset: A large collection of virtual images for traffic vision research. *IEEE Transactions on Intelligent Transportation Systems*, 20(6):2072–2084, 2018.

[168] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *CoRR*, abs/1609.08675, 2016.

[169] Yuxin Peng, Yunzhen Zhao, and Junchao Zhang. Two-stream collaborative learning with spatial-temporal attention for video classification. *CoRR*, abs/1711.03273, 2017.

[170] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *CoRR*, abs/1902.06162, 2019.

[171] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games*, pages 341–348, Santorini, Greece, Sep 2016. IEEE. The Best Paper Award.

[172] Charles Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Shane Legg, and Stig Petersen. Deepmind lab. *CoRR*, abs/1612.03801, 2016.

[173] Jorge Aldaco, Travis Armstrong, Robert Baruch, Jeff Bingham, Sanky Chan, Kenneth Draper, Debidatta Dwibedi, Chelsea Finn, Pete Florence, Spencer Goodrich, et al. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation. *arXiv preprint arXiv:2405.02292*, 2024.

[174] Raymond M Klein. Inhibition of return. *Trends in cognitive sciences*, 4(4):138–147, 2000.

[175] Wei Jiang, Weiwei Sun, Andrea Tagliasacchi, Eduard Trulls, and Kwang Moo Yi. Linearized multi-sampling for differentiable image transformation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2988–2997, 2019.

[176] Tony Lindeberg. Scale-space: A framework for handling image structures at multiple scales. 1996.

[177] Jan J Koenderink and Andrea J van Doorn. Representation of local geometry in the visual system. *Biological cybernetics*, 55(6):367–375, 1987.

[178] Jeff Hawkins. *A thousand brains: A new theory of intelligence*. Basic Books, 2021.