Contour Integration in Artificial Neural Networks

by

Salman Khan

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
System Design Engineering

Waterloo, Ontario, Canada, 2021

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Neil Bruce Associate Professor, School of Computer Science, University of Guelph
Supervisor:	Bryan Tripp Associate Professor, Department of System Design Engineering, University of Waterloo
Supervisor:	Alexander Wong Professor, Department of System Design Engineering, University of Waterloo
Internal Member:	Chris Eliasmith Professor, Department of System Design Engineering and Department of Philosophy, University of Waterloo
Internal Member:	Graham Taylor Adjunct Associate Professor, Department of System Design Engineering, University of Waterloo, and, Professor, School of Engineering, University of Guelph,

Statement of Contributions

I am the sole author of Chapters 1, 2, 3 and 6.

Parts of Chapter 4 and Chapter 5 have been previously published in Khan et al. [2020]. Salman Khan designed the contour integration model. Salman Khan, Alexander Wong and Bryan Tripp conceived the experiments and the analyses that were carried out. Salman Khan conducted all the experiments and analyses under the supervision of Bryan Tripp and Alexander Wong. Salman Khan wrote the manuscript. Bryan Tripp designed Figure 4.1. The remaining figures were designed by Salman Khan or were taken from other sources as referenced in their captions. Salman Khan, Alexander Wong and Bryan Tripp reviewed the manuscript.

Code for all the model, datasets, experiments and analyses is available at `https://github.com/salkhan23/contour_integration_pytorch`. The code for extracting contours from natural images was conceived and written by Bryan Tripp. Additions to this algorithm as well as the remaining code were written by Salman Khan.

Abstract

Under difficult viewing conditions, the brain's visual system uses a variety of modulatory techniques to supplement its core feedforward signal. One such technique is contour integration, whereby contextual stimuli from outside the classically defined receptive fields of neurons can affect their responses. It manifests in the primary visual (V1) cortex, a low layer of the visual cortex, and can selectively enhance smooth contours. Several mechanistic models, that can account for many of its neurophysiological properties, have been proposed in the literature. However, there has been limited exploration of the learning of biologically realistic contour integration circuits or of the role of contour integration in the processing of natural images.

In this thesis, I present a biologically-inspired model of contour integration embedded in a task-driven artificial neural network. The model can relate the low-level neural phenomenon of contour integration to the high-level goals of its encompassing system. It uses intra-area lateral connections and an internal architecture inspired by the V1 cortex. Its parameters are learnt from optimizing performance on high-level tasks rather than being fixed at initialization. When trained to detect contours in a background of random edges, a task commonly used to examine contour integration in the brain, the model learns to integrate contours in a manner consistent with the brain. This is validated by comparing the model with observed data at the behavioral, neurophysiological and neuroanatomical levels.

The model is also used to explore the role of contour integration in the perception of natural scenes. I investigate which natural image tasks benefit from contour integration, how it affects their performances and the consistency of trained models with properties of contour integration from more extensively studied artificial stimuli. Specifically, the model was trained on two natural image tasks: detection of all edges and the ability to distinguish if two points lie on the same or different contours. In natural images, the model was found to enhance weaker contours and demonstrated many properties that were similar to when it was trained on synthetic stimuli. Moreover, the features it learnt were robust and generalized well to test data from outside the distribution of training data. The results provide new evidence that contour integration can improve visual perception and complex scene understanding.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors, Bryan Tripp and Alexander Wong. There invaluable guidance, insightful questions and continuous support over the years have helped me develop as a researcher and complete this dissertation.

I would also like to thank all current and past members of the the Bio-Robotics, Artificial Intelligence and Neuroscience (BRAIN) Lab at the University of Waterloo. Thank you all for the invaluable discussions and for providing a great atmosphere for doing research.

I would also like to thank my parents for their unwavering support during this whole process. Finally, I would like to thank my wife, Sadia Sabieh, and our children, Zahra, Zayd and Noor, for putting up with my late hours and for their unconditional love, support and patience during this journey.

Dedication

This is dedicated to Sadia Sabieh, Zahra Salman, Zayd Salman and Noor Salman. Without their love and support this would not have been possible.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial intelligence	45
ANNs	Artificial neural networks	46
BCE	Binary cross entropy loss function	88
BIPED	Barcelona images for perceptual edge detection dataset	106
BN	Batch normalization layer	48
BP	Back propagation learning algorithm	50
BPTT	Back propagation through time learning algorithm	54
CI	Contour integration block	66
CNNs	Convolutional neural networks	47
cRF	classical receptive field	21
CurrDI	current based divisive inhibition model variant	94
DNNs	Deep neural networks	45
e-cRF	extra-classical receptive field	21
FWHM	Full-width-half-maximum	114
GRU	Gated recurrent unit network	56
hGRU	Horizontal gated recurrent unit network	61
IGM	Inverted Gaussian masked sparsity constraint	90
IoU	Intersection over union	72
IT	Inferior temporal cortex	11
LGN	Lateral Geniculate Nucleus	8
LSTM	Long short term memory network	55
M-cells	Magnocellular retinal ganglion cells	8
ML	Machine Learning	45
MT	Medial temporal cortex	13
P-cells	Parvocellular retinal ganglion cells	8
RCD	Relative co-linear distance	76, 117
RCNNM	Recurrent convolutional neural network model variant	96
rCNNs	Recurrent convolutional neural networks	59

ReLU	Rectified linear unit activation function	47
RF	Receptive field	7
RNNs	Recurrent neural networks	52
RPCM	relaxed positivity constraint model variant	92
SD	Standard deviation	73
SOTA	state of the art	58
V1	Primary visual cortex	14
V2	Secondary visual cortex	11
V4	Visual area 4	12

Chapter 1

Introduction

In the brain, the ventral visual system is responsible for detecting objects in our complex natural environments. It is a deep hierarchical system that processes input in multiple stages. In lower layers, simple features, such as edges, bars and gratings, are extracted over spatially confined areas. Next, as the signal propagates through a series of linear-nonlinear transformations [Poirazi et al., 2003], these features are grouped together into more and more complex features. Finally, in the culminating layer of the ventral visual stream, the Inferior Temporal Cortex, inputs are rendered in a format from which object identity can be consistently decoded across a variety of input variations [DiCarlo et al., 2012]. This deep feedforward stream of information has inspired many recent successes in artificial visual systems. In particular, Deep Neural Networks (DNNs) have recently surpassed human level performances in a variety of complex visual tasks including object classification [He et al., 2016, Huang et al., 2017, Xie et al., 2017, Tan and Le, 2019] and object detection and segmentation [He et al., 2017, Redmon and Farhadi, 2018, Liu et al., 2016, Ronneberger et al., 2015].

Despite these many successes, many functional limitations of DNNs have also been discovered, such as being easily fooled by adversarial stimuli [Goodfellow et al., 2014, Nguyen et al., 2015], poor generalization outside training data distribution [Geirhos et al., 2018, Hendrycks and Dietterich, 2019, Serre, 2019], and poor data sample efficiency [Lake et al., 2015]. The ventral visual stream is much more robust and does not suffer from these limitations. One possible explanation for this robustness may be additional mechanisms that exist in the brain but not yet in DNNs.

In addition to feedforward connections, there exists a plethora of long range intra-area lateral and inter-area feedback connections in every cortical area of the brain. This suggests that the flow of information may not be strictly feedforward and might involve some form of recurrence, especially when dealing with complex stimuli [Lamme and Roelfsema, 2000, Gilbert and Li, 2013, van Bergen and Kriegeskorte, 2020]. The existence of these additional connections have been known about for a long time [Kravitz et al., 2013, Felleman and Van Essen, 1991, Markov et al., 2014]. However, due to their complex modulatory nature (as opposed to the external input driven nature of feedforward connections), multiple parallel routes and the diverse functionality they are thought to perform, their precise roles are not completely understood. Moreover, as they typically involve simultaneous activity in multiple brain areas and limitations in our ability to measure brain activity, analyzing them in the brain is challenging.

In contrast, the internals of DNNs are much more accessible. Isolating and modeling these connections within DNNs could potentially shed more light on their roles and mechanisms. Recent work along this line has shown some successes. In particular, incorporating recurrence, especially biologically inspired forms, into convolutional neural networks has been shown to outperform parameter-matched feedforward models in some classification tasks with occluded or additive white noise corrupted objects [Spoerer et al., 2017, Rajaei et al., 2019]. Moreover, compared to feedforward models, these types of model show better alignment between their internal representations and those of multiple brain cortices [Nayebi et al., 2018, Kar et al., 2019] and have better sample efficiency when trained with limited data [Linsley et al., 2020b]. Despite these early successes, further exploration is needed to better understand how specific types of neurophysiological recurrent connections affect representation and robust behaviour.

In this thesis, I expand on this approach by focusing on the role of recurrent lateral connections in V1 cortex and how it relates to contour integration. Contour integration [Field et al., 1993, Li et al., 2006, Hess et al., 2014, Roelfsema, 2006] is a phenomenon that occurs in V1 cortex, the lowest layer of the ventral visual stream, where stimuli from outside a neuron's classical receptive field (cRF) modulate its feedforward responses. In particular, responses are enhanced if a preferred stimulus within the cRF is part of a larger contour. Under difficult viewing conditions, the ventral visual stream uses contour integration to *pop out* smooth natural contours from the background.

Contour integration is thought to be mediated by intra-area lateral and higher-layer feedback connections. However, the exact mechanism employed by the brain remains unknown. Past computational models, [Li, 1998, Piëch et al., 2013, Ursino and La Cara, 2004a, Hu and Niebur, 2017], have tested potential mechanisms using targeted synthetic stimuli and have replicated many of its neurophysiological properties. However, apart from some recent work, [Linsley et al., 2018, 2020b], there has been little exploration of the role of contour integration in the perception of natural scenes. Moreover, most of these models use fixed pre-determined lateral connection patterns between component neurons and do not address how they may be learnt.

The aim of this work is two-fold, to see if brain-like contour integration can be learned in DNNs and whether modeling contour integration inside these models can further our understanding of the neural phenomenon. In particular, the role it plays in the perception of natural scenes, the over-arching goal of the ventral visual stream.

1.1 Outline

Chapter 2 is a background chapter that focuses on aspects of neuroscience that are relevant to contour integration. First, the brain's visual cortex, its ventral visual pathway and the primary visual (V1) cortex are reviewed. V1 elements and connections that are pertinent to contour integration are described. Next, several observed behavioural and neurophysiological properties of contour integration as well as the mechanistic theories of how it may be realized by the brain are described. Finally, several existing computational models of contour integration are discussed and compared.

Chapter 3 is another background chapter that focuses on artificial neural networks. Two types of networks, convolutional neural networks and recurrent neural networks are described. Next, combinations of these two types of networks that have recently been proposed as biologically plausible models are presented. Finally, precautions that should be taken into account when equating ANN models to biological systems are highlighted.

In Chapter 4, first, the proposed contour integration model is detailed. Second, the model is trained on a dataset consisting of synthetic stimuli, similar to those that have been used to explore the properties of contour integration in the brain. Its performance is compared with a feedforward convolutional neural network with matching capacity. Third, trained networks are tested for consistency with several observed properties of contour integration at the behavioural and neurophysiological levels. Fourth, learnt lateral connections of the trained model are analyzed and compared with lateral connections in the V1 cortex as well as with those used in existing computational models. Fifth, a detailed sensitivity analysis of the model's parameters is conducted. Sixth, different permutations of the model are compared with the proposed model. Finally, differences between the model and another recently proposed ANN model based on lateral connections in the V1 cortex [Linsley et al., 2018] are highlighted.

In Chapter 5 the model is used to explore the role of contour integration in the perception of natural images. First, the model is evaluated on its ability to detect contours in an image. Second, trained models are analyzed to see if they are better at detecting weak edges, as has been hypothesized as one of the potential advantages of contour integration [Li, 1998, Piëch et al., 2013]. Third, the model is tested on a novel task that involves tracing smooth contours to determine if two points in an image are connected by the same contour. Fourth, trained models are analyzed for consistency with properties of contour integration observed when trained with synthetic stimuli.

Chapter 6 concludes this thesis with a summary of the main contributions and highlights some potential future directions.

Chapter 2

Computational neuroscience of vision

Our ability to unconsciously and effortlessly recognize objects in our visual environment and perform multiple tasks with them belies the enormity of this task. As a testament to this, the visual processing system occupies $\approx 55\%$ of the cerebral cortex of the macaque monkey brain [Felleman and Van Essen, 1991, Kruger et al., 2012], one of our closest living relatives whose brain have been extensively studied [Kandel et al., 2013][1]. Computational neuroscience is a field that tries to understand how the brain works by building models of systems and processes in the brain. Building replica models is a proven technique to develop a deeper understanding compared to just passively observing them. The approach can highlight intricate details that are important for correct functionality and can also be used to quantitatively compare alternative hypotheses. Finally, it can even be used to explore and develop new insights into the workings of complex, seemingly opaque systems [Cichy and Kaiser, 2019]. It is the approach I use in investigating the role of low-level contour integration in the high-level functions of the visual system.

This background chapter is divided into three main sections. In the first section, I describe the visual cortex, the computational system of the brain that is responsible for processing visual signals from our eyes. Next, I describe the ventral visual stream, a subsystem within the visual cortex which is responsible for deciphering what objects are in our visual environment. I describe its architecture as well as its processing of object shape. Object shape and form are one of the main visual attributes that neurons in the brain are specialized in handling. Next, I describe the primary visual cortex, a region of the visual cortex where effects of the neural phenomena of contour integration have been

[1]In humans this number is closer to $\approx 33\%$ [Kupers and Ptito, 2014]. However, this number is arguable; it is hard to prescribe fixed numbers as many cortices are thought to be multi-modal.

well studied[2]. I review its architecture, connections and its components neurons, especially those that are involved with contour integration.

In the second section, I focus on contour integration, the main focus of this thesis. I describe many of the behavioural and neurophysiological properties that have been discovered about it. Next, I review the different mechanistic theories that have been proposed on how the brain actually performs contour integration.

In the third and final section of this chapter, I describe and compare some of the existing computational models of contour integration.

2.1 Neurons

Neurons are the information processing cells of the brain. All large systems of the brain as well as transmission lines to and from them are composed of neurons. The morphology and biophysical processes of neurons specialize them for the rapid conduction of electrical signals. Many types of neurons exist in the brain and in the peripheral nervous system. However, neurons generally have three main parts: dendrites, a cell body and an axon. Dendrites form tree-like branching structures that collect signals from many upstream neurons. These signals accumulate in the cell body and if they cross a certain threshold, a short electrical pulse or action potential is generated on the axon. Axons are the main transmission conduits by which information is conveyed over large distances. As such their lengths range from 0.1mm to 1m [Kandel et al., 2013]. Very long axons are coated with insulating myelin sheaths to improve their conduction speeds. Along axons, myelin sheaths are regularly interrupted by nodes of Ranvier where action potentials are regenerated to help them propagate over large distances [Kandel et al., 2013].

Neurons communicate with each other using action potentials. All neurons are surrounded by an insulating cell membrane. These cell membranes contain many ion channels and pumps that regulate the flow of charged ions in to and out of neurons. At rest, neurons maintain excessive negative charge inside the cell compared to the surrounding extracellular fluid. The difference in charge on the inside and outside of a cell membrane is referred to as the membrane potential of the cell. At rest, there is a balance of charges

[2]The effects of contour integration have also been observed in other cortices of the ventral visual stream, including V2 [Chen et al., 2014] and V4 [Chen et al., 2017]. The exact mechanism of contour integration is not yet fully understood. There is an on-going debate on whether V1 lateral connections or feedback from higher layers are the main conduit. However, it is clear that responses of V1 neurons are affected by contours that extend outside their small, individual receptive fields [Li et al., 2006].

flowing into and out of the cell and the resultant membrane potential is referred to as the resting membrane potential. Changes in the membrane potential causes ion channels to open and close. During signaling, the membrane potential of a neuron changes drastically and rapidly. As the potential on the inside increases above a certain threshold, voltage gated channels in the cell membrane flood the inside of the neuron with positive charge. This generates a large positive potential on the inside of the cell. This action potential propagates through the axon of a cell. After the action potential is transmitted the neuron is quickly repolarized to its resting potential by ion channels.

Neurons communicate with each other at specialized sites known as synapses. At these sites, information from a sending or presynaptic neuron is chemically conducted over a gap or synaptic cleft to a receiving or postsynaptic neuron. Synapses are typically located on specialized terminals on the axons of presynaptic neurons and can connect with the dendrites, soma or axons of postsynaptic neurons. An action potential sent down an axon releases chemical neurotransmitters into the synaptic cleft. These neurotransmitters bind to receptor molecules in the postsynaptic neuron which control its ion channels and changes their permeability. A synapse can have either a depolarizing excitatory or shunting inhibitory influence on the postsynaptic neuron based on the type of neurotransmitter and the postsynaptic receptor molecules.

Generated action potentials can vary in their durations, amplitudes and shapes. These variations can occur between action potentials generated by different neurons as well as those generated by the same neuron. However, action potentials are commonly stereotyped as identical events [Dayan and Abbott, 2001]. A neuron can generate multiple spikes to a given stimulus. These spike trains vary over time and also different trials. There are many ways to characterize the spike trains of neurons. For a review, see Dayan and Abbott [2001]. A frequently used description is the time-dependent trial-averaged firing rates of neurons. The higher the firing rate (up to a maximum firing rate) of a neuron the more the input stimuli matches the features the neuron is selective for.

2.1.1 Receptive field

The region of the sensory field that generates responses from a neuron is referred to as its receptive field (RF). For a neuron in the visual system, the RF is defined as the area of visual field that elicits a response from it. Not all stimuli in the RF of a neuron causes it to fire action potentials. Neurons are selective for particular signals in their RFs as described in the following sections.

2.2 The visual system

2.2.1 Pre-cortical processing

The processing of visual inputs starts in the retina of the eye. The retina contains photo-receptive cells that transduce light into electrical signals. There are two main types of photo-receptive cells: rods and cones. Rods are sensitive to low light intensity but cannot detect colours. Cones, on the other hand, are sensitive to different bands of light wavelengths (colours) but require higher light intensity to respond. There are three different types of cones: red cones respond to long wavelengths of light, green cones respond to low-to-medium wavelengths and blue cones respond to short wavelengths. Visual space is not equally covered by rods and cones. The fovea, a small region in the retina that is responsible for central vision, is densely populated with cones. The distribution of cones drops with distance away from the fovea. Rods on the other hand, are almost entirely absent from the fovea, but densely cover other parts of the retina.

Outputs of photo-receptive cells are mostly received by bipolar cells. These bipolar cells link the inner and outer layers of the retina. Many types of bipolar cells (> 10) have been discovered [Euler et al., 2014]. Different types of bipolar cells collect and transform information from photo-receptive cells in different ways. Moreover, their responses are modulated by a variety of horizontal and amacrine cells, predominately through inhibitory lateral connections.

Outputs of bipolar and amacrine calls are received by ganglion cells. There are two main types of ganglion cells: magnocellular (M-cells) and parvocellular (P-cells). P-cells are sensitive to colours and represent mainly foveal vision, while M-cells can detect changes in low light intensities. P-cells have higher spatial resolution compared to M-cells. M-cells on the other hand, are more sensitive to temporal changes in light patterns and have faster response times [Maunsell et al., 1999]. Hence, they are responsible for detecting movement in the visual environment.

Ganglion cells are sensitive to light in a small region of visual space. Their RFs have two distinct regions; a central area that responds to a certain light intensity or colour and a surrounding area that is sensitive to light of the opposite intensity or colour. Their RFs are often described as center-surround RFs. This makes ganglion cells sensitive to spatial contrast changes, such as dots and edges. The different sizes of the central parts of their RF gives them different spatial selectivities and makes them sensitive to edges of varying thickness. Ganglion neurons are the only information carrying cells that project out of the eyes. Most of them send their outputs to the Lateral Geniculate Nucleus (LGN) of the

Thalamus in the mid-brain region.[3]

The LGN is the first region of the brain to receive extracted visual information. LGN neurons have similar selectivities as Ganglion cells. The majority of LGN cells send their outputs to the visual cortex where most of the processing of visual information takes place.

2.2.2 The visual cortex

The visual cortex, the brain's vision processing center, is the largest sensory system in the brain. Anatomically, it is located on the cerebral cortex, a thick folded sheet of neural tissue that forms the outer layer of the brain. Starting in the lower back of the brain (occipital lobe), it extends anterolaterally into the inferior temporal lobe (behind the ear) and also anterosuperiorly into the parietal lobe (upper back of the brain)[4]. Visual information enters the visual cortex principally through the primary visual (V1) cortex. From here, it proceeds along two processing pathways in parallel [Goodale et al., 1994, Mishkin et al., 1983]; the ventral and dorsal pathways. The ventral ("What") pathway, is in charge of perceiving objects in the environment while the dorsal ("Where") pathway is responsible for localizing objects in space and in extracting information relevant to interacting with them.

Each pathway is a highly structured hierarchical system composed of many functionally distinct cortical areas [Kruger et al., 2012, Kandel et al., 2013]. Component areas form one or more internal representations of the outside world from which properties they are interested in can be easily extracted. In lower areas, neurons extract simple features over small spatially confined visual regions. Neurons in these areas tend to be retinotopically arranged; neighboring neurons respond to stimuli in nearby visual space. Higher in the hierarchy, the complexity of features as well as the size and degree of overlap of spatial regions of interest of neurons increases. Internal representation becomes less spatially defined and more aligned with pathway functions.

Each cortical area is highly structured. A prominent organizing motif of cortical areas is functional proximity; neurons that respond to similar stimuli over overlapping spatial

[3]There are also other pathways from the retina into the brain, including from the retinas to the superior colliculus which controls eye movement and from the retina to the Pretectum, which control the pupils and the amount of light let into the eyes. However, these other pathways mostly send control signals that do not take part in the deciphering of visual information.

[4]The brain in split into two hemispheres. Visual cortical areas are duplicated in both hemispheres. Each hemisphere processes slightly more than half the contralateral (opposite side) visual field from both eyes. In the descriptions above and what follows, I describe only the parts on a single hemisphere, but the description is valid for both.

areas are grouped together in cortical columns [Mountcastle et al., 1957]. These *vertical* columns run perpendicular to the cortical surface. Each cortical area contains columns relevant to its functionality. Hence, cortical columns are sometimes considered the basic functional units of cortical areas. Neurons in a cortical column generally receive inputs from a common source in lower areas and send outputs to a common target in higher areas.

Another prominent organizing principle of cortical areas is their division into distinct *horizontal* layers[5]. Layers are numbered from 1 to 6. Starting at the outer surface of the cortex, these layers run parallel to the surface and their index increases with depth. Layers are differentiated based on the types of neurons they contain and their dominant connectivity patterns. Typically, afferent axons from neurons in lower areas terminate in layer 4. Consequently, layer 4 is referred to as the feedforward input layer. Layers above are collectively referred to as superficial or supra-granular layers while layers below are referred to as deep or infra-granular layers. Layers 2 and 3 are often combined together and are collectively referred to as layer 2/3. The complex internal circuitry of a visual area includes many inter- and intra-layer excitatory and inhibitory connections Kandel et al. [2013].

Cortical areas communicate with each other using feedforward and feedback connections. The densest connections tend to be between adjacent areas [Felleman and Van Essen, 1991]. However, cortical areas also directly connect with far away cortical areas that do not follow sequentially in the hierarchy. Moreover, cortical areas also connect to other sub-cortical areas. In turn, these sub-cortical areas connect back to the visual cortex at different levels of the hierarchy. In general, feedforward inputs tend to be visual input driven and feedback inputs tend to modulate the feedforward information flow. Inter-area connections are mostly excitatory [Chalupa and Werner, 2004]. However, their net effect is not always excitatory as some of these connections target inhibitory inter-neurons. Inhibitory neurons modulate feedforward neurons and reduce their activity.

The visual cortex also purportedly processes different attributes of visual information separately [Livingstone and Hubel, 1988]. Cortical areas V1 and V2 have identifiable sub-divisions that contain clusters of neurons specialized in shape (typically referring to edges and contours of objects), colour, direction of motion and disparity processing. These specialized sub-divisions preferentially connect with sub-divisions in other cortical areas that also specialize in the same visual aspects [Kandel et al., 2013]. Deeper in the hierarchy, visual pathways diverge and cortical areas specialize in features relevant to their

[5]Layers of visual cortical areas are different from layers in artificial neural networks (see Chapter 3). Layers in artificial neural networks typically correspond to a single feed-forward step whereas layers of a cortical area refer to partitions within it. Multiple feedforward and local recurrent steps occur in each cortical area, both across and within layers.

encompassing pathways. These information streams get routed to pathways they are most useful for; colour and form to the ventral pathway while motion and disparity to the dorsal pathway [Kandel et al., 2013, Kruger et al., 2012].

Although these information streams are generally considered separate from each other, there are many areas where they may potentially interact. In lower areas, neurons in specialized sub-regions connect with other neurons in neighboring sub-regions. While deeper in the hierarchy, areas that are considered exclusively to be part of one visual pathway send secondary connections to cortical areas that are part of the opposite pathway. Finally, neurons that jointly encode multiple visual attributes, such as colour and form [Garg et al., 2019] and motion and disparity [Grunewald and Skoumbourdis, 2004] have also been discovered.

2.2.3 The ventral visual pathway

The ventral visual pathway of a macaque monkey brain is shown in Figure 2.1A[6]. The central feedforward path traverses the V1, V2, V4 cortices and finally the inferior temporal (IT) cortex (see Figure 2.1B). Below, I briefly discuss the processing of edges and object shapes as it relates to first-pass 'core' object recognition [DiCarlo et al., 2012]. Contour integration, the focus of this research, is a neural phenomenon associated with edge processing. The ventral pathway also uses other visual attributes, such as colour, depth, and texture to detect objects. For a more detailed description see Kandel et al. [2013], Kruger et al. [2012], Orban [2008].

Visual signals from the LGN enter the visual cortex through the V1 cortex. As it is part of both the ventral and dorsal pathways, V1 neurons respond to a variety of features including edges, bars, gratings, colours, motion, disparity and line-endings. Edges appear to be one of the main features that V1 neurons are sensitive to. There are neurons that are selective for many of their properties including orientations, spatial frequencies, movements, and disparities between signals from both eyes.

The outputs of V1 neurons are predominately sent to the V2 cortex [Markov et al., 2014]. V2 neurons are selective for similar features as V1 neurons but over slightly larger RFs. In addition, some V2 neurons can detect more complex non-luminance defined contours such as illusory and texture defined contours. Illusory contours may result from one object occluding another, or a perceived shape that does not actually exist in the visual

[6]It is common to describe non-human primate brain areas and discuss human brain homologs. Many behavioural, anatomical and physiological studies have been done on monkey brains and several similar cortical areas and processing stages have been found in humans.

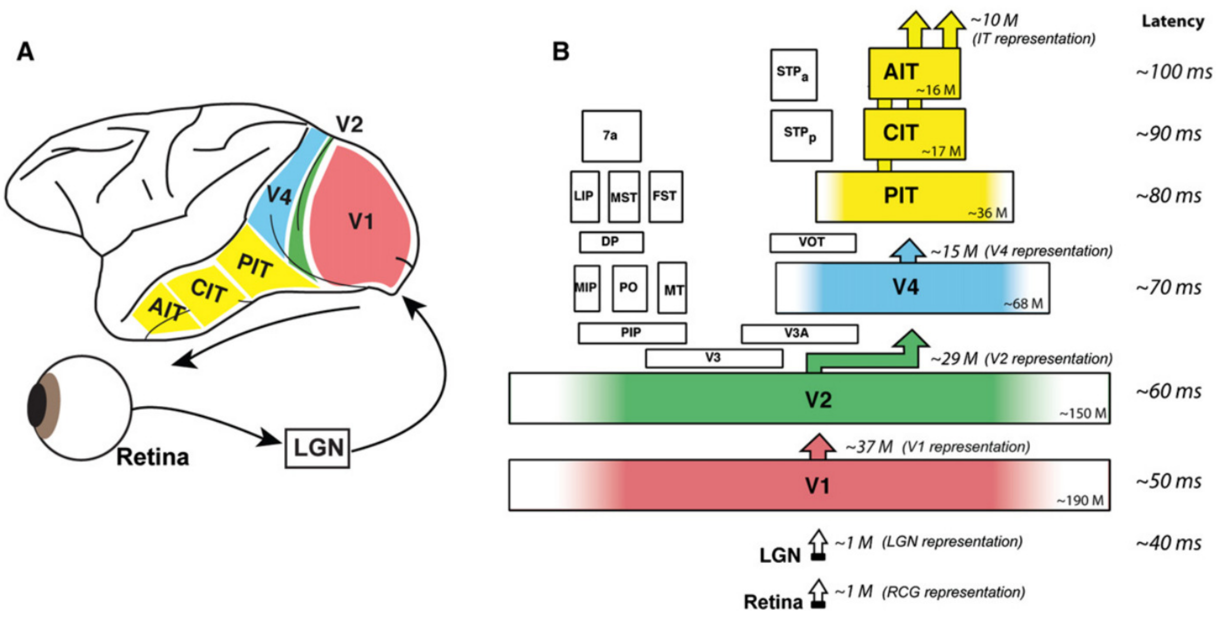

Figure 2.1: **Ventral visual stream**. The feedforward path though the ventral stream. A, Macaque monkey brain. LGN is the Lateral Geniculate Nucleus. PIT, CIT, AIT are the posterior, central and anterior parts of the inferior temporal cortex respectively. B, Block diagram of the ventral stream (coloured blocks only). The approximate population size of each area is given in the bottom of each box (M=million). Right side column specifies the average response latency of each cortex when a stimulus is presented at time 0. Reprinted from DiCarlo et al. [2012] with permission from Elsevier.

environment. Texture boundaries may result from breaks or changes in texture patterns on surfaces which are not defined by a luminance boundary. V2 neurons are also known to encode additional attributes to already detected contours. For example, border ownership neurons respond differently to detected contours based on where the object they belong to lies with respect to their RF [Zhou et al., 2000].

Beyond V2, the two visual pathways split. On the ventral pathway, V2 outputs are sent to the V4 cortex. V4 neurons have larger RFs and build upon the features extracted by V2 neurons. In particular, many V4 neurons are selective for curved contours. These neurons are sensitive to the relative position of contour fragments rather than their precise positions. They respond to specific contour arrangements anywhere in their RFs. Moreover, some V4 neurons are selective for the relative position of the contour with respect to the center of the shape they belong too. This represents a shift away from absolute representation to a more object centric one. Other V4 neurons detect simple shapes formed from specific combinations and positions of contours. The process of combining component contours to form object shapes is one of the famous neural binding problems [Kruger et al., 2012, Feldman, 2013, Treisman, 1996]. Unfortunately, the actual process is not fully understood. Finally, some V4 neurons are selective for even more complex contours such as kinetic and disparity defined contours. Kinetic contours arise from differences in motion or speed of objects in one area compared to another. These separation boundaries may be stationary or moving, but are separate from the movement of component objects.

Outputs of V4 are mostly sent to the IT cortex [Markov et al., 2014]. The IT cortex is the final brain area that processes visual information exclusively. Neurons in the IT cortex have large RFs that typically include the Fovea. They respond to object shapes and specific combinations of shapes. These shapes can be formed from combining any of the different types of contours and simple shapes detected by lower areas, i.e. IT neurons are cue tolerant. Moreover, they respond similarly to the same shape anywhere within their large RFs, i.e they are position tolerant. The IT cortex is typically further divided into sub-regions (see Figure 2.1) through which information mainly flows sequentially. Along this trajectory, the complexity of features IT neurons are selective for increases and they develop further tolerances to other transformations [Rust and DiCarlo, 2010]. Finally, even in the deepest levels of the IT cortex, neurons respond to object parts. There are no 'grandmother' cells that singularly represent complete objects in the IT cortex. The identity of what objects are in visual space have to be read from the population responses of groups of IT neurons [Hung et al., 2005].

Along this central core vision path, each cortex sends just as many feedback connections to downstream cortices as feedforward connections to upstream areas. Direct shortcuts between shallow and deep stages have also been found. For example, V1 directly connects to V4 [Markov et al., 2014]. V1 also directly connects with visual cortices outside the ventral stream which in turn connect back to the ventral stream at deeper stages; V1 directly connects with the medial temporal (MT) cortex which also sends projections to the IT cortex [Kravitz et al., 2013]. Similar to these feedforward shortcuts many feedback shortcuts have also been found. For example, V4 directly feeds back into the V1 cortex [Markov et al., 2014]. Moreover, V1 also receives feedback from outside the ventral visual stream, from V3 and MT cortices [Angelucci et al., 2002]. This suggests recurrent highly interconnected processing of information by the ventral visual stream [Kravitz et al., 2013]. Recently, it has also been suggested that these recurrent connections are necessary for core object recognition in some cases [Kar et al., 2019].

2.2.4 Primary visual cortex

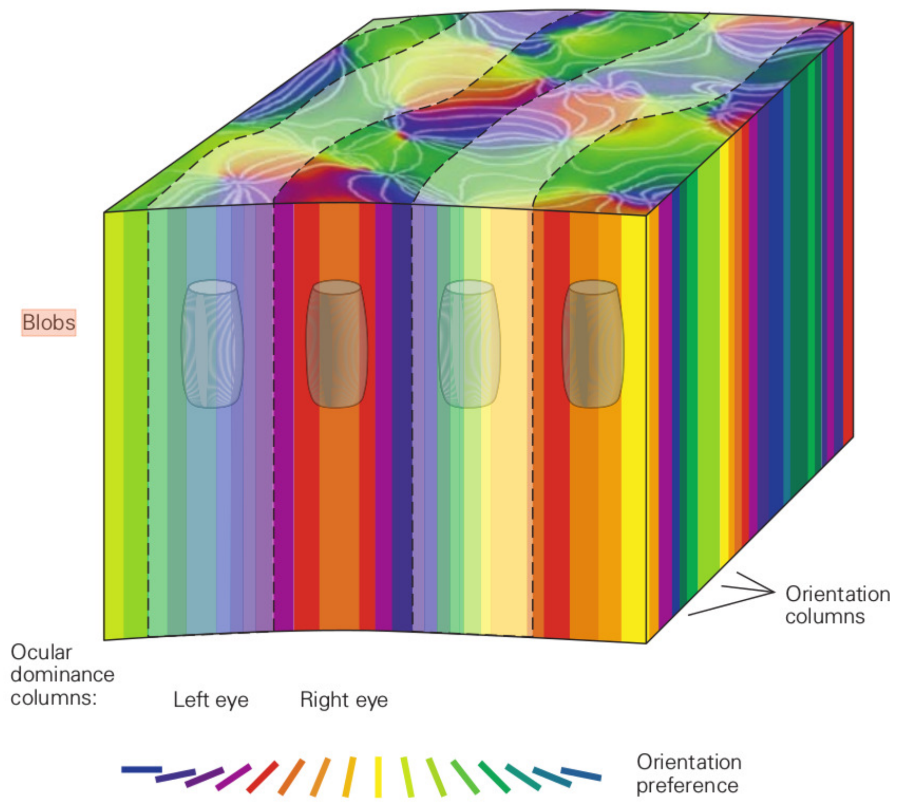

Figure 2.2: **Cross section of V1 Cortex**. Reproduced from Kandel et al. [2013] with permission. See text for description.

The primary visual (V1) cortex is the largest visual area. Like the rest of the brain, it is highly structured. The entire visual field is topographically represented across V1; there is a smooth transition of visual space coverage across its surface. However, not all visual areas are equally represented. The central 3° of eccentricity[7] around the fovea is represented by 30% of the V1 cortex [Schira et al., 2007]. With distance away from the fovea and towards peripheral visual areas, the amount of cortical surface dedicated to visual areas decays logarithmically [Schira et al., 2007]. RF sizes of V1 neurons also increase linearly with eccentricity [Freeman and Simoncelli, 2011]. Neurons responding to areas near the fovea

[7]Eccentricity, the distance from the fovea, and RF sizes of neurons are usually expressed in units of visual angle. Visual angle is the size a stimulus subtends on the retina.

have small cRFs $< 0.5°$ while those close to $30°$ of eccentricity can have RF sizes of up to $5°$. Due to this variability, RF measurements are usually accompanied by the degree of eccentricity where they have been measured.

There is also functional specificity within V1; neurons representing similar stimuli are grouped together into cortical columns. Many studies have found V1 neurons that strongly respond to edges, bars and gratings. As orientation is a primary feature of edges, bars and gratings, many orientation columns are found in the V1 cortex. Within a orientation column, neurons respond to stimuli with similar orientations over overlapping visual areas. Moreover, there is an ordered transition of orientation preferences between adjacent columns; orientation preferences of nearby columns gradually change. The visual areas covered by adjacent columns also overlap [Gilbert, 1992]. The set of orientation columns that respond to similar stimuli at the same visual area covering all possible orientations, form pinwheel structures. In the V1 cortex of a macaque, these orientation hypercolumns have average widths of 0.75mm [Kandel et al., 2013, Stettler et al., 2002]. A typical cross section of the V1 cortex is shown in Figure 2.2.

Most orientation columns also show strong preference for signals from a particular eye (ocular dominance). These ocular preferences form alternating stripes across V1's cortical surface and have widths between 0.75 to 1mm [Kandel et al., 2013]. Interspersed between orientation columns are short blob-like structures. These blob structures contain clusters of neurons that are very sensitive to colour but show little orientation selectivity. Orientation columns span all 6 layers while blobs exists mainly in layer 2/3. Blobs repeat frequently across orientation columns with an average separation 0.75 mm. Occasionally, especially in studies focusing on colour processing, regions surrounding blob structures, that contain orientation columns, are referred to as inter-blobs regions.

V1 Neurons

The majority of V1 neurons are sensitive to edge locations, orientations, spatial frequencies and phases. Two types are commonly found: simple and complex cells. Simple cells have defined areas in their RFs where they prefer light (ON) and dark (OFF) signals and respond maximally when inputs align with them. These RFs are formed from summing activity of specific spatial arrangements of center-surround RFs of LGN cells [Kandel et al., 2013]. Similarly, complex cells sum the output of multiple simple cells to form their RFs. By doing so, complex cells develop tolerances to some properties of simple cells, such as phase and position, while retaining sensitivities to other properties, such as orientation and spatial frequency.

Many edge detecting V1 neurons have RFs that evolve over time; the ON and OFF regions of their RFs shift over time [Dayan and Abbott, 2001]. These neurons are concerned with motion detection and are more relevant to the dorsal stream. Some edge detecting neurons have excitatory and inhibitory regions in their cRFs. These neurons respond only when edges of particular configurations activate only the excitatory regions. These end-stopping or hyper-complex cells are used to signal the ends or corners of bars and gratings. Additionally, they also play a role in determining the direction of motion of moving parts and gratings.

V1 also contains neurons that are sensitive to colours, disparity and motion. As they are not thought to be involved in contour integration, they are outside the scope of this review. For more details see Kandel et al. [2013].

Mathematical model of V1 neurons

A widely used model of the receptive fields of simple cells is the 2D Gabor function [Dayan and Abbott, 2001, Kruger et al., 2012, Movellan, 2002],

$$g(x,y) = \exp\left(\frac{-\left(x'^2 + \gamma y'^2\right)}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda} + \psi\right), \tag{2.1}$$

$$x' = (x - x_c)\cos\theta + (y - y_c)\sin\theta,$$

$$y' = -(x - x_c)\sin\theta + (y - y_c)\cos\theta.$$

It is composed of a Gaussian envelope that defines its spatial extent and a sinusoidal wave that defines ON and OFF regions within the cRF. Parameters for the Gaussian envelope include: (x_c, y_c), the center of the Gaussian, σ, the spatial spread, and γ, a scaling factor that defines elongation in the y axis compared with the x axis. Parameters for the sinusoid include: λ, the wavelength of the sinusoid which determines the number of ON and OFF regions, ψ, the spatial phase which determines the relative positions of ON and OFF regions in the cRF and θ the orientation of the sinusoid (See Figure 2.3).

The response of a model neuron is found by multiplying input signals with the neuron's 2D Gabor profile. Outputs are then passed through a nonlinear activation function (typically a Sigmoid function) to model the nonlinear responses of neurons. Occasionally, the output is additionally passed through a inhomogenous Poisson process to generate spikes. This is known as the linear-nonlinear model of spiking neurons.

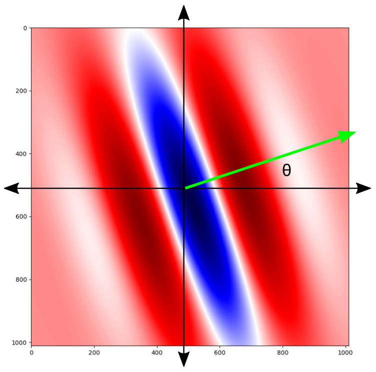

Figure 2.3: **2D Gabor Filter**. 2D Gabor Filters are commonly used to model V1 edge detecting simple cells.

Neuronal connections in V1

Feedforward connections

LGN neurons that project to V1, send most of their axons to layer 4C of V1. Recipient neurons in L4 have short axons that project mainly to more superficial layers. Neurons in layer 2/3 mostly project out of V1 and into the V2 cortex. Hence, the feedforward path through V1 is described as a 2 step process [Callaway, 2004].

Lateral intra-area connections

Lateral connections run parallel to the cortical surface and exist in all layers of V1. They are most prominent in layers 2/3 and 5 and fewest in layer 4 [Bijanzadeh et al., 2018]. Most neurophysiological studies have focused on lateral connections in layer 2/3. Neurons in these layers predominately project externally to deeper cortical areas. However, the axons of some of these neurons branch out (axon collaterals) and connect to neighboring neurons in the same layer. Other neurons project directly to far away locations also within the same layer.

The source of these lateral connections are excitatory pyramidal neurons [Gilbert and Wiesel, 1983]. Approximately 80% terminate at other excitatory cells while 20% terminate at inhibitory interneurons [McGuire et al., 1991]. Lateral connection are known to be modulatory; direct stimulation does not make neurons fire, but their activation does make it easier when presented with stimuli in their RFs [Hirsch and Gilbert, 1991].

Stettler et al. [2002] measured the spatial extent of V1 horizontal connections in macaque monkeys. First, axon labeling dye was injected into a single orientation column. Next, highlighted horizontal axons were mapped onto a columnar orientation map of the corresponding V1 region. At injection sites, the average RF of V1 neurons was measured to be 0.5°. It was found that lateral connections extended up to 4°, approximately 8 times the RF of V1 neurons.

Moreover, lateral axons consistently formed patchy structures with many branching out in small separable clusters. This was attributed to the functional specificity of lateral connections where neurons with similar functionality (edge, colour, etc.) are linked [Malach et al., 1993, Gilbert, 1992]. Strong spatial correlations were seen in axon collaterals at intervals of 0.75mm. As this is similar to the average width of V1 orientation hypercolumns, it was suggested that horizontal connections were connecting V1 columns.

Stettler et al. [2002] also analyzed orientation preferences of neurons near lateral axon terminals and compared them with the orientation preferences of injected orientation columns. At short distances (< 0.5mm), lateral axons connected to columns of all orientation preferences. In contrast, at larger distances, horizontal connections showed a preference for columns with the same orientation as the source location's orientation. The distributions of orientation preferences were quite wide (60° bandwidth), indicating that horizontal cells were connecting cells with similar orientations and may explain the clustering observed at axon collaterals. This orientation selectivity of long range horizontal connections has also been reported by others and in different animal species [Bosking et al., 1997, Rockland and Lund, 1983, Malach et al., 1993, Sincich and Blasdel, 2001].

Sincich and Blasdel [2001] traced V1 lateral connections in new world monkeys. The V1 cortices of these animals lack ocular dominance columns and have a less interrupted retinotopic layout. Similar to Stettler et al. [2002], it was found that lateral connections were connecting orientation columns with similar orientations. However, connections were not targeting all orientation columns with similar orientations. Lateral connections were more dense and elongated in the direction of the preferred orientation of source V1 neurons.

Many other properties of lateral connections have also been discovered. The characteristic view of lateral connections is that they connect columns at far away locations with similar orientations, however, the densest connections are with local columns. Moreover, connections drop off sharply with distance [Chisum et al., 2003]. Lateral connections also tend to be reciprocal [Kisvarday and Eysel, 1992]. Furthermore, lateral connections are slow. Girard et al. [2001] measured conduction speeds of V1 connections. Feedforward and feedback connections were found to have similar conduction speeds (average 3.5m/s). Lateral connections, on the other hand, were an order of magnitude slower (average 0.1 m/s). For this reason, they are thought to affect the late or sustained parts of V1 neural responses.

Layer 2/3 also contains many inhibitory interneurons. At least 10 different types of inhibitory neurons have been discovered [Kubota, 2014]. Their different innervation targets (axons, dendrites, soma), different inhibitory functions they subserve, and their relatively fewer numbers compared to excitatory neurons has made it difficult to identify the patterns and trends of inhibitory neurons. Relatively few studies have been been conducted on the properties of inhibitory neurons in the macaque visual cortex [Vanni et al., 2020]. However, it is known that horizontal spread of inhibitory neurons is much shorter than excitatory neurons [Lund and Wu, 1997, Kritzer et al., 1992] and they connect homogeneously (are non-patchy) [Kritzer et al., 1992, Vanni et al., 2020]. Moreover, in other species, inhibitory neurons have been found to be less orientation selective compared to excitatory neurons [Bosking et al., 1997, Kerlin et al., 2010].

Feedback connections from higher layers

Feedback from higher layers can be excitatory or inhibitory. Conduction velocities over feedback connections are as fast as feedforward connections [Girard et al., 2001]. However, feedback is received from multiple sources which may affect the timescales of its influence.

Stettler et al. [2002] measured the spatial extent and orientations of V2 feedback connections. Axon highlighting dye was injected into V2 columns and highlighted axons in V1 were analyzed. The spatial extent of V2 feedback connections was found to be similar to that of V1 horizontal connections. However, unlike lateral connections, their axons connected with orientation columns of all orientation preferences. Moreover, different injection sites displayed different degrees of spatial clustering. Finally, it was found that the density of feedback connections was an order of magnitude less then that of lateral connections. However, only feedback from V2 was measured. As feedback is received from multiple other sources, this is not a complete picture of their relative densities.

Contrasting this study, Angelucci et al. [2002] found the spatial extent of V2 feedback connections to be larger than that of lateral V1 connections. Moreover, the spatial extent of feedback connections from several deeper cortices (V3 and MT) were also analyzed. As the depth of the feedback source increased, the spatial extent of their feedback connections also increased. It was claimed that spatial extent of lateral connections was not sufficient to explain the full range of V1 contextual interactions and that feedback connections were necessary to explain some contextual interactions.

2.3 V1 contour integration

Natural images contain a plethora of edges of all shapes and sizes. The task of identifying what objects are present includes identifying object edges and linking them together into holistic perceptions. To a large extent this is handled in deep parts of the ventral stream (see Section 2.2.3). However, components of this processing, or mechanisms to assist later area grouping, reside in lower cortices [Gilbert, 1992, Li et al., 2006, Kapadia et al., 1995, Hess et al., 2014]. In the V1 cortex, this mechanism is referred to as contour integration.

Contour integration was first observed psychophysically as the popping out of patterns of small line segments that followed smooth trajectories in the presence of distractors (see Figure 2.4 for an example). Test subjects were able to detect large contours amongst a sea of similar but randomly oriented segments with high accuracy. Kapadia et al. [1995], Li et al. [2006] found that V1 neurons whose RF overlapped aligned segments showed elevated responses, even though the full contour extended well outside their individual RFs. Moreover, concurrent behavioural tests showed that empirically measured enhancement gains were highly correlated with the detection accuracy of test subjects. This showed that contour integration or at least part of it, occurs in the V1 cortex.

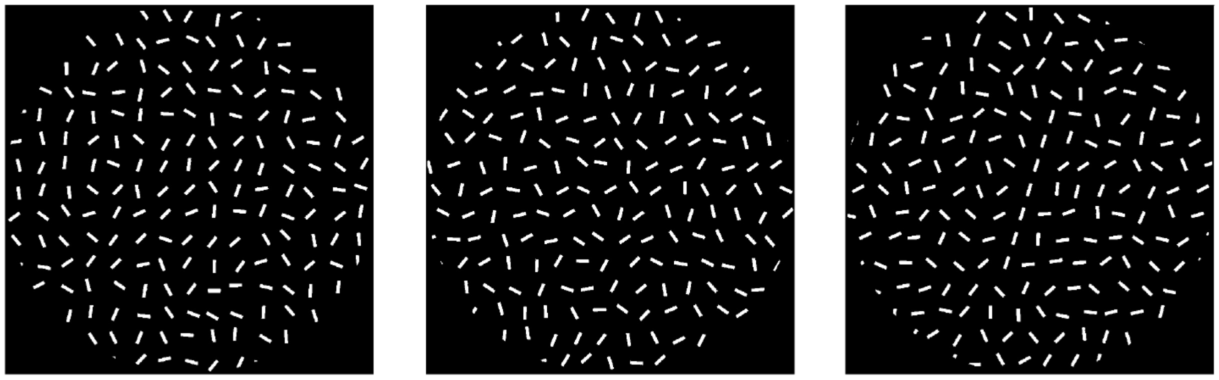

Figure 2.4: **Contour pop out effect**. (Left) a 45° contour formed from co-linear line segments. All line segments are identical except for their orientations. Each segment is slightly smaller than the RF such that only one fragment can completely lie within it. (Middle) As fragment spacing increases, the ability to detect the contour decreases. (Right) As fragment spacing decreases, contour detectability increases. Reprinted from Li et al. [2006] with permission from Elsevier.

In the following section, I discuss some of the known properties of contour integration. The actual mechanism the brain used for contour integration is not fully understood. I describe the two main mechanistic theories of contour integration. The section concludes with a brief description of other forms of context influences used by the brain, especially those that also occur in the V1 cortex.

2.3.1 Extra-classical receptive field

Given that stimuli from outside the RF of a neuron can influence its within RF responses, many researchers expand the definition of the receptive field of a neuron to include a surrounding region where these influences can occur. To distinguish this surround region from the classically defined RF (cRF), where direct stimulation generates responses, this surround region is referred to as the extra-classical receptive field (e-cRF).

2.3.2 Properties of contour integration

Contour integration has been known for a long time and many behavioural and neurophysiological properties have been discovered. Initially, it was considered a fundamental mechanism by which edges are bound together in our perception of complete objects. More recently, and given its relative late time of occurrence, compared with the time frame of fast core object recognition, it is considered to play a role in more complicated visual scenarios and thought to contribute to robust vision. In this section, I highlight some of the well known properties of contour integration.

1. **Contour integration is mostly modulatory**: Enhancement gains from stimuli in the e-cRF are typically observed only when a within cRF stimulus is concurrently present [Kapadia et al., 1995]. In rare occasions, optimally placed e-cRF stimuli can trigger a neuron with no within cRF stimulus [Kapadia et al., 2000]. This may be a possible mechanism of how illusory contours of occluded objects are completed [Albright and Stoner, 2002]. Although filling in of illusory contours is generally associated with the V2 cortex.

2. **Facilitation from nearby co-linear fragments**: Kapadia et al. [1995] analyzed the influence of a flanking bar placed in the e-cRF on the responses of V1 neurons. First a bar was placed inside the cRF of a target neuron. The bar was tuned to the preferred orientation of the target neuron and its response was noted. Next, a second identical bar was placed in the e-cRF. Third, the target neurons firing rate was measured as the flanking bar was moved and reoriented. The procedure was

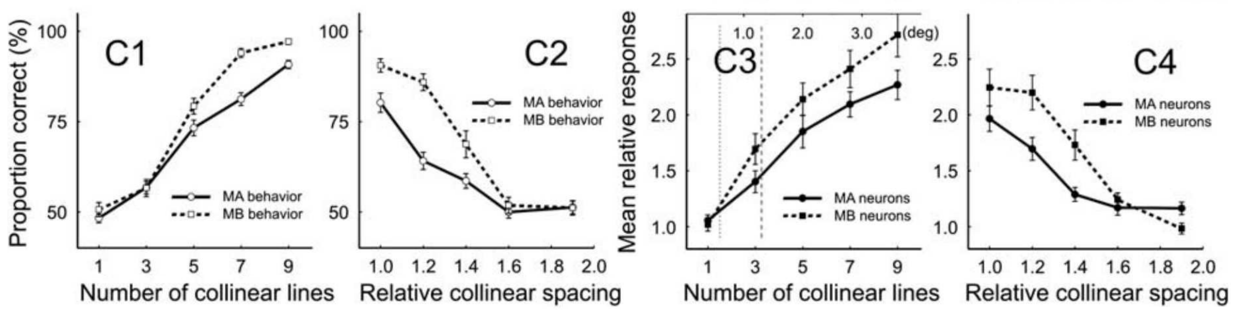

Figure 2.5: **Contextual effects of nearby co-linear fragments**. Li et al. [2006] measured the behavioural and neuronal responses of multiple V1 neurons of two monkeys (MA, MB) when co-linear identical fragments were placed outside the cRF of optimally stimulated neurons. As the length of the contour increased, both the behavioural (C1) performance and V1 enhancement gains (C3) increased. As the spacing between fragments increased, both the behavioural performance (C2) and enhancement gains (C4) decreased. Mean relative response is defined as the ratio of average response to co-linear neighbor conditions to the average response when randomly oriented neighbors were configured. In all conditions, an identical fragment, tuned to the preferred orientation of the target neuron, is present in the cRF of monitored neurons. Relative co-linear spacing is defined as the ratio of the distance between two fragments to fragment length. In C3, dotted and dashed vertical lines are the average and maximum receptive field size of measured V1 neurons (top axis). Reprinted from Li et al. [2006] with permission from Elsevier.

repeated for many V1 neurons. For most neurons, peak responses were observed when the flanking bar was positioned on the preferred orientation axis of the target neuron, oriented similar to the within cRF bar and separated by small distances.

Li et al. [2006] extended these results in two ways. First, the effect of multiple co-aligned fragments was analyzed. It was found that enhancement gain monotonically increased with the number of co-aligned fragments (Figure 2.5C3). The longest considered contour was composed of 9 co-aligned fragments and extended 5.6 times the average cRF length of measured neurons. An average gain of 2.5 was seen over the case when all neighbors were randomly oriented. Even at this length, gain saturation was not seen. Second, the effect of spacing between fragments was analyzed. It was found that enhancement gains decreased with separation distance (Figure 2.5C4). Moreover, these enhancement gains also showed strong correlation with behavioural performance; longer contours were easier to detect (Figure 2.5C1) and contours became less salient as the spacing between fragments increased (Figure 2.5C2).

3. **Time scale of contour enhancement:** The time course of neuronal responses generally consist of two phases. An initial high amplitude but short lived phase and a second lower amplitude but sustained phase. Responses during the initial phase are thought to correspond to the fast feedforward core object recognition path through the ventral visual stream. Late phase responses are thought to involve recurrent and feedback connections and correspond to more selective processing. Li et al. [2006] analyzed the time course of contour enhancement. It was found that contour enhancement did not affect initial transient responses. On the other hand, firing rates during the sustained phase were enhanced (Figure 2.6). Interestingly, the length of the contour only affect the amount of gain and not the response latency; responses were enhanced at the same time for all contour lengths.

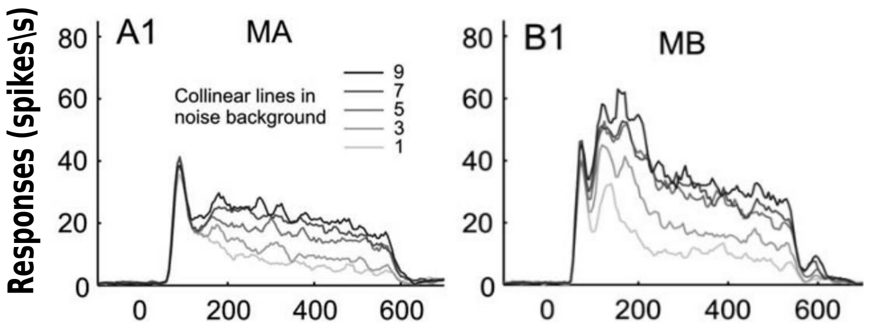

Figure 2.6: **Timescale of Contour Enhancement**. Li et al. [2006] measured the spike rates of two monkeys (MA and MB) to analyze when the effects of contour integration affected responses. Spikes were binned into time bins of 5ms each. Results were averaged over multiple cells in each animal separately. Contour enhancement affects the late sustained phase of neural responses. Gains increased monotonically with contour length. Reprinted from Li et al. [2006] with permission from Elsevier.

4. **Suppression from nearby orthogonal fragments:** Kapadia et al. [1995] also analyzed the effect of moving a flanking bar lateral to the co-linear direction of optimally triggered V1 neurons. First, a flanking bar was placed in the co-linear axis of a V1 neuron that was optimally stimulated. Next, the flanking bar was moved in the orthogonal to the co-linear direction and the responses of the V1 neuron were monitored. Enhancement gains for most cells decreased with movement away from the co-linear axis. In fact, at large displacements, most cells were inhibited to levels below the case when the stimulus in the cRF was presented in isolation.

   Kapadia et al. [2000] constructed 2D maps of regions of contextual influences by sliding two symmetric flankers in the surround of a bar that was optimally activating a target neuron. A four loop structure was found (see Figure 2C of Kapadia et al. [2000]). Excitation was highest in the co-linear direction. Moreover, two forms of inhibitions were seen. First, directional inhibition which was strongest in the orthogonal to co-linear direction. Second, a weak omni-directional inhibition existed in the surround. Excitation was stronger than all forms of inhibition. However, this result was contrast dependent. As the contrast between the stimuli and the background increased, excitatory regions decreased and surround stimuli became mostly inhibitory. Directed inhibition was still the strongest form of inhibition.

5. **Discontinuities between fragments eliminate enhancement**: Kapadia et al. [1995] found that if the flanking bar was replaced with a T-shaped stimulus such that there was an interruption between the two bars, contextual facilitation was eliminated in most cells.

6. **Enhancement gain decreases with contour curvature:** Kapadia et al. [1995] analyzed the effects of rotating a flanking bar away from the preferred orientation axis. Along their preferred orientation, most neurons preferred co-linear flankers and their enhancement gains decreased as the flanking bar rotated away.

   In a series of behavioural experiments, Field et al. [1993] tested detection accuracy of curved contours. Curved contours were constructed by inserting a fixed rotational offset between component fragments. Component fragments were randomly assigned a negative or positive offset with respect to the previous fragment. The same rotational offset (apart for the direction) was used for all fragments of the contour, and the constructed curve was said to have a curvature value defined by this rotational angle. It was found that detectability monotonically decreases as degree of curvature of the contour increased. However, even with very high contour curvature (as $\pm 60°$) test subjects were able to detect contours above chance.

7. **Jitter in contour fragments reduces contour detectability:** Contours are much easier to detect if the orientations of component fragments are aligned with the overall curvature of the contour. In addition to fixed rotational offsets between successive fragments, Field et al. [1993] analyzed the effect of adding random rotation offsets between fragments. These random offsets moved the orientation of component fragments away from the direction of the contour. Random rotation offsets away from the contour direction were found to strongly influence detectability; small jitter offsets of $\pm 30°$ made contours undetectable.

8. **Present only in cluttered environments:** V1 neurons respond strongly when

contours are presented in isolation. Contrastingly, the same stimuli presented with complex backgrounds, evoke relatively smaller responses. Li et al. [2006] found that contour integration enhancement gains were also different in the two scenarios. Enhancement gains are only observed when background clutter was present. Surprisingly, for isolated contours, as the number of co-linear segments increased, most V1 cells were slightly inhibited.

9. **Concurrent development of V1 and V4 responses**: Chen et al. [2014] investigated the role of higher layer feedback in contour integration by simultaneously recording responses of V1 and V4 neurons. First, V4 neurons that responded to large linear contours were found. Second, V1 neurons with the same preferred orientation and that lay within the large cRF of V4 neurons were isolated. Third, test contours, composed of fragments smaller than the cRF of V1 neurons were constructed. These test contours were embedded in a sea of similar but randomly oriented fragments. Fourth, V1 neurons were grouped into those whose cRF overlapped co-linear fragments (V1 contour sites) and those that were 1° degree away from the contour (V1 background site). A detailed time course of events was established: V1 feedforward response, V4 feedforward response, V1 contour enhancement and finally V1 background suppression. Furthermore, the authors also noted that both V1 and V4 responses continued to develop even though feedforward input remained stationary. They suggested that contour integration involves multiple bidirectional interactions between V1 and V4 neurons.

10. **Top down attention can selectively strengthen enhancement gains:** Li et al. [2006] recorded V1 responses when monkeys were presented with contour stimuli but were engaged in unrelated tasks. In the absence of attention, neural responses still increased monotonically with contour length. However, the amount of gain was significantly less compared with the case when monkeys were attending the contour task. Moreover, not all V1 neurons responding to smooth contours are enhanced. In Roelfsema et al. [1998], monkeys were engaged in a task that involved tracing smooth contours while avoiding similar distractor contours. The firing rates of neurons whose cRF overlapped target and distractor contours were monitored. Enhancement of V1 neurons whose cRFs overlapped with that target contour were enhanced more than those whose cRFs overlapped distractor contours.

11. **Contour closure enhances detectability:** Even though curvature reduces contour detectability, if the ends of contours close in onto themselves or appear as they will, contour detectability is improved [Kovacs and Julesz, 1993].

2.3.3 The Association Field Model

Based on a series of psychophysical experiments on curved contour enhancement, Field et al. [1993] suggested a linking structure between V1 neurons that could be responsible for contour integration. This association field does not spread equally in all directions and is elongated in the direction of the preferred orientation axis of neurons. Moreover, it is more spread out than just the co-linear path (see Figure 2.7 Left). Within this association field, individual fragments must align with the direction of the contour at each point (see Figure 2.7 Right).

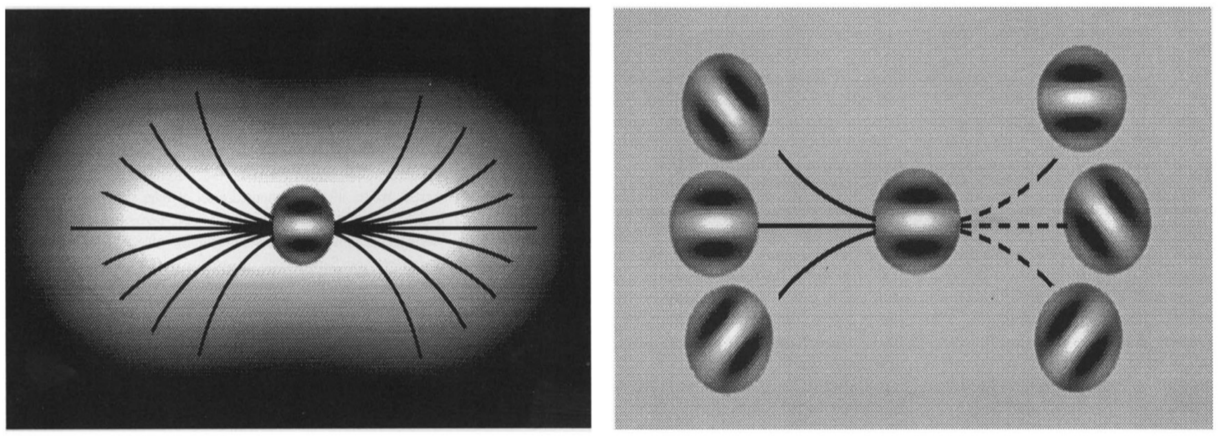

Figure 2.7: **Association Field Model**. (Left) The association field model specifies regions outside the classical receptive fields of V1 neurons where curve-linear (including co-linear) contours are enhanced. Contour fragments within this regions are allowed to have different orientations than the source V1 neuron. However, they must follow smooth trajectories and individual fragments need to be in the same direction as the contour. (Right) Fragments on the left side of the central fragment enhance responses, while those on the right do not. Reprinted from Field et al. [1993] with permission from Elsevier.

The association field model has become commonly associated with lateral connections in V1. In fact, most computational models of contour integration (see Section 2.4) use this model to define lateral connections between component units. It has also been extended to model directed inhibitory connections in these models. More recently, in deep neural network based models where connections are learnt rather than predefined, learnt lateral connection structures are often compared with the association field model [Linsley et al., 2018, Spoerer et al., 2020].

2.3.4 Mechanistic theories of contour integration

The mechanism that the brain uses to perform contour integration is not yet known. Two main theories exist. The first hypothesizes that V1 horizontal connections are principally responsible [Stettler et al., 2002, Li et al., 2006, Liang et al., 2017]. Proponents of this theory cite the anatomy of lateral connections [Bosking et al., 1997, Malach et al., 1993, Stettler et al., 2002]. Horizontal connections preferentially target cells with similar orientation in nearby spatial locations - a requirement for identifying smooth contours. Natural objects have mostly smooth contours with only abrupt sharp changes [Geisler et al., 2001]. Hence lateral connections can potentially enhance object contours. Moreover, existing computational models have shown lateral connections on their own are sufficient to boost smooth contours [Li, 1998, Ursino and La Cara, 2004a, Piëch et al., 2013]. Feedback from higher layers may still be involved but is not necessary for contour enhancement.

The second theory postulates that feedback from higher layers is more heavily involved [Roelfsema, 2006, Angelucci et al., 2002, Chen et al., 2014, Hu and Niebur, 2017]. Object templates learnt by higher layers may be upsampled and sent down to V1, thereby providing V1 neurons with an alternative source of smooth contours to enhance. Proponents of this theory cite the need to selectively enhance object contours as opposed to all smooth contours [Roelfsema et al., 1998]. Moreover, constant response latencies were measured for different contour lengths [Li et al., 2006, Chen et al., 2014]. If lateral connections were responsible, latencies for large contours should be longer. Lateral connections may still be involved but play a secondary role of enhancing all smooth contours [Hu and Niebur, 2017], enhancing weak feedforward responses in low-contrast visual inputs [Angelucci et al., 2002] or texture and shading continuation [Ben-Shahar and Zucker, 2004].

2.3.5 Other forms of V1 contextual influences

The majority of contextual influences are inhibitory; multiple stimuli in nearby areas suppress neural responses. Enhancements due to e-cRF stimuli are only observed when within-cRF stimuli have low contrast (weak feedforward input) or when stimuli are present in highly cluttered environments. Surround modulation (SM) [Angelucci et al., 2017, 2002] is a neuronal phenomenon that focuses more on the suppressive impacts of e-cRF. Typically, SM experiments stimulate the whole e-cRF rather than selective parts of it (as normally done in contour integration experiments). Suppression is strongest when within-cRF and e-cRF stimuli are similarly oriented and weakest when they are orthogonal. It is thought that this mechanism highlights dissimilarities in images such as corners and reduces redundancies in neural responses (by suppressing neurons responding to similar stimuli).

Another major suggestion of SM is the existence of two distinct regions of the e-cRF: a near e-cRF and a far e-cRF. These two regions have different tuning properties and may be generated by different circuits. The near e-cRF is more suppressive and depends on the orientation difference between within-cRF and e-cRF stimuli. While stimuli in the far e-cRF suppress within-cRF responses regardless of their orientation. Similar to contour integration, there is an ongoing debate regarding whether lateral or feedback connections are chiefly responsible. However, the spatial extent of horizontal connections was found to be congruent with the near e-cRF only and higher layer feedback connections are thought to be responsible for far e-cRF effects [Shushruth et al., 2013].

Another closely related phenomenon is contrast normalization [Carandini and Heeger, 2012]. The linear-nonlinear model of neurons does not account for population influences. To address this, mechanistic models of contrast normalization, whereby the responses of individual neurons are divisively normalized with the responses of a pool of nearby neurons, were developed. Two types of V1 inhibitory influences, cross-oriented and omni-directional suppression, have been accounted for using these models [Carandini et al., 1997]. Contrast normalization has also been used to explain nonlinear responses in other cortices (reviewed in Carandini and Heeger [2012]). Similar to SM, the emphasis of contrast normalization is on the inhibitory effects of surrounding stimuli.

Many other forms of contextual modulation, from outside the cRF, exist in the ventral stream including: filling in of occluded and illusory contours, filling in of homogeneous surfaces, filling in of contours and surfaces in the blind spot and retinal lesions, and border ownership cells which respond differently if foreground objects lie to the left or right of their cRF. For a review, see Albright and Stoner [2002]. As these are associated with cortices deeper than V1, they are outside the scope of the current work.

2.4 Computational models of contour integration

Several computational models of contour integration exist in the literature. In this section, I select four models for a more detailed description. Each selected model is compared according to the following criteria: (1) the architecture of the model, (2) how lateral connections are modelled, (3) how feedback connections are modelled if included, and (4) which neurophysiological properties of contour integration are incorporated.

2.4.1 Li-1998 Model

The contour integration model of Li [1998] is heavily cited. Specifically targeting the hypothesis that contour integration is possible in V1 alone, the model is built using common V1 elements and connections: localized excitatory edge detectors, inhibitory interneurons, feedforward, horizontal and recurrent connections. The model replicates many neural properties, including response amplification of neurons representing the contour, suppression of neurons responding to spurious edges, and synchronized oscillations of responses of neurons responding to contours. Furthermore, it was qualitatively shown that enhancement gains and oscillation synchrony increased with contour length [Eckhorn et al., 1988] and if component fragments formed a closed shape, while both decreased as contour curvature increased. Finally, with her model, Li [1998] demonstrated how lateral connections can be used to fill in missing parts of a contour provided sufficient feedforward input is present.

The architecture of the model is shown in Figure 2.8. It consists of a hexagonal grid of hypercolumns at discrete nonoverlapping spatial locations (Figure 2.8B). Each spatial location, i, contains a hypercolumn that is composed of K neuron pairs. A neuron pair consists of an excitatory (x) and an inhibitory (y) neuron. Component neurons are edge detectors and prefer edges of the same orientation, θ. Within a hypercolumn, orientation preferences of neuron pairs change gradually covering all orientations, $\theta_k = \frac{k\pi}{K},\ k = 1, 2...K$.

Visual inputs $\hat{I}(i\beta)$, consisting of fragmented oriented edges, enter the model only at discrete nonoverlapping spatial locations. At each location, i, only the strongest edge, with orientation β, is passed through. The feedforward input to neuron pair, $i\theta$, is this visual input processed by its tuning profile, $I_{i,\theta} = \hat{I}(i\beta)\exp\left[\frac{|\theta-\beta|}{\pi/8}\right]$. Only excitatory neurons receive feedforward input.

V1 activity is modeled down to the dynamics of membrane potentials of individual

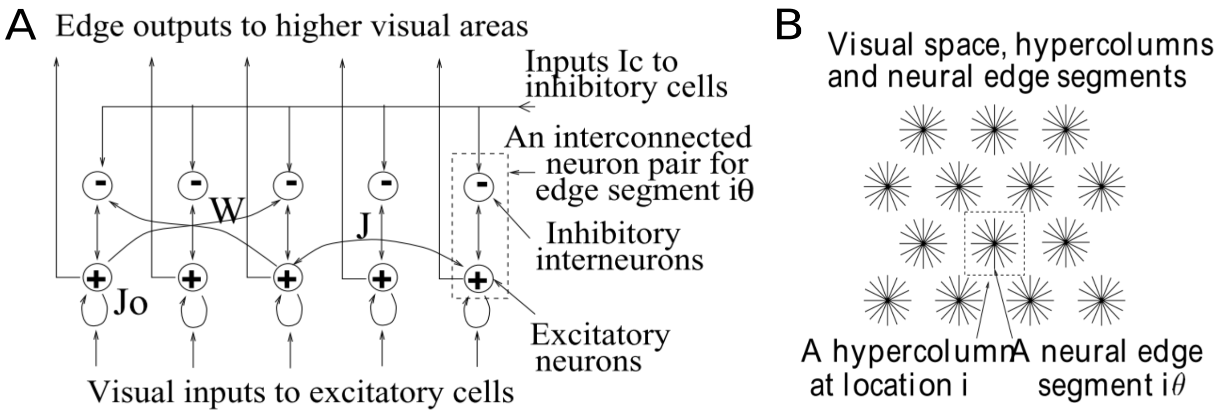
A
Edge outputs to higher visual areas
Inputs Ic to inhibitory cells
An interconnected neuron pair for edge segment iθ
W
J
Inhibitory interneurons
Jo
Excitatory neurons
Visual inputs to excitatory cells
B
Visual space, hypercolumns and neural edge segments
A hypercolumn at location i
A neural edge segment iθ

Figure 2.8: **Contour integration model of Li** [1998]. (A) Model components and the connections between them. A vertical set of excitatory (+) and inhibitory (-) neurons defines a neuron pair (neural edge segment), the fundamental building block of the model. Black lines denote connections between neurons. See text for parameter descriptions. (B) Hexagonal grid layout of hypercolumns. At each spatial location i there are K=12 neuron pairs that uniformly cover all orientations.

neuron pairs. For a single neuron pair, it was defined as,

$$\dot{x}_{i\theta} = -\alpha_x x_{i\theta} - \sum_{\triangle\theta} \psi(\triangle\theta) g_y(y_{i,\theta+\triangle\theta}) + J_o g_x(x_{i\theta}) + \sum_{j\neq i,\theta'} J_{i\theta,j\theta'} g_x(x_{j\theta'}) + I_{i,\theta} + I_o, \quad (2.2)$$

$$\dot{y}_{i\theta} = -\alpha_y y_{i\theta} + g_x(x_{i\theta}) + \sum_{j\neq i,\theta'} W_{i\theta,j\theta'} g_x(x_{j\theta'}) + I_c, \quad (2.3)$$

where, $\dot{x}_{i\theta}$, $\dot{y}_{j\theta'}$ are first order time derivatives, and $\frac{1}{\alpha_x}$, $\frac{1}{\alpha_y}$ are time constants of the excitatory (x) and inhibitory neurons (y) at location i and with an orientation preference of θ, respectively.

Each excitatory neuron sums its membrane potential, $x_{i,\theta}$, before passing it through a non-linear activation function, $g_x(x_{i,\theta})$. An excitatory neuron sends its output to its inhibitory counterpart, back to itself with synaptic strength, J_o and to hypothetical feed-forward neurons in higher layers.

Each excitatory neuron is suppressed by all inhibitory neurons within its hypercolumn, $\sum_{\triangle\theta} \psi(\triangle\theta) g_y(y_{i,\theta+\triangle\theta})$. The term $\psi(\triangle\theta)$ models the effect that as the orientation preference of the neighboring inhibitory neuron diverges, the suppression it causes also decreases.

Excitatory neurons also receive excitation from nearby excitatory neurons at other spatial locations, $\sum_{j \neq i, \theta'} J_{i\theta, j\theta'} g_x(x_{j\theta'})$. Here, $J_{i\theta, j\theta'}$ is the strength of the connection with its excitatory neighbor at spatial location j and orientation preference θ'.

Finally, I_o is the background inputs to the excitatory neurons. It includes an activity normalization term,

$$I_o = I_{e,background} - 2 \left[\frac{\sum_{j \in S} \sum_{\theta'} g_x(x_{j\theta'})}{\sum_{j \in S} 1} \right]^2, \tag{2.4}$$

where S defines the spatial extent of activity normalization. These short omni-directional connections connect all excitatory neurons within a small region. These connections normalize the output of a neuron with the responses of a pool of nearby neurons. The primary purpose of these connections is to suppress background neurons once contours have been enhanced.

Inhibitory neurons only send their outputs to the excitatory neuron in their same neuron pair. Nearby excitatory neurons can also suppress the neuron pair by stimulating the inhibitory neuron, $\sum_{j \neq i, \theta'} W_{i\theta, j\theta'} g_x(x_{j\theta'})$, where $W_{i\theta, j\theta'}$ is the strength of the connection between excitatory neuron at $j\theta'$ and the inhibitory neuron at $i\theta$. All these connections are shown in Figure 2.8A. I_c is the background activity of the inhibitory node.

$J_{i\theta, j\theta'}$ and $W_{i\theta, j\theta'}$ represent oriented lateral connections between neighbors. Both of these matrices are fixed and model many properties including: (1) only neighbors with similar orientations are connected, (2) connection strengths decrease with orientation differences, (3) neighbors that are co/curve-linearly aligned with the neuron pair's orientation are enhancing (have positive $J_{i\theta, j\theta'}$ values) while neighbors orthogonal to this direction are suppressing (have positive $W_{i\theta, j\theta'}$ components) and (4) that the contribution from neighbors decreases with distance. Laterally connected neighbors for an example neuron pair with a horizontal orientation preference are shown in Figure 2.9.

Not discounting the fact that top down signals can influence V1 contour integration, a feedback mechanism is also included. A simple model of feedback is used where oriented edge signals (similar to visual inputs) are fed into inhibitory neurons (via the background activity, I_c). However, feedback is modeled as a separate external input without specifying how higher layers could generate such a precise signal and how specific neuron pairs may be targeted.

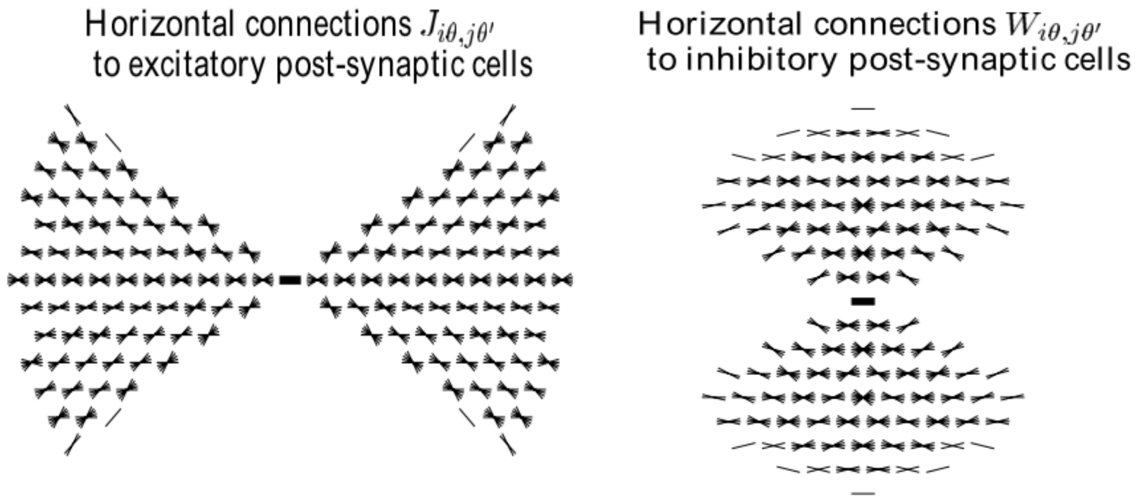
Horizontal connections $J_{i\theta,j\theta'}$ to excitatory post-synaptic cells
Horizontal connections $W_{i\theta,j\theta'}$ to inhibitory post-synaptic cells

Figure 2.9: **Example lateral connections of Li [1998] model for a neuron pair with a horizontal orientation preference**. Each visible edge represents a nonzero connection weight with its neighbor. The left figure shows an excitatory association field, [Field et al., 1993], while the right figure shows that a similar suppressive association field is also included. Mathematical descriptions of $J_{i\theta,j\theta'}$ and $W_{i\theta,j\theta'}$ can be found in Li [1998].

2.4.2 Ursino and La Cara - 2004 Model

The model of Ursino and La Cara [2004a] is also based on the hypothesis that contour integration is possible in V1 alone. Compared with the model of Li [1998], where visual inputs need to be pre-processed to extract the highest amplitude oriented edge segments at each spatial location, edge extracting neurons are incorporated into the model, as thalamic inputs. This allows the model to operate on any visual input. The model consists of a grid of 50$\times$50 orientation hypercolumns defined over discrete nonoverlapping spatial locations (Figure 2.10 right). Each hypercolumn is composed of 16 excitatory and 4 inhibitory interneurons. Within each column, neurons have similar stimulus preferences except for orientation. Orientation preferences change gradually and uniformly cover all orientations ($\triangle\theta = 11.25^o$ for excitatory and $\triangle\theta = 45^o$ for inhibitory neurons).

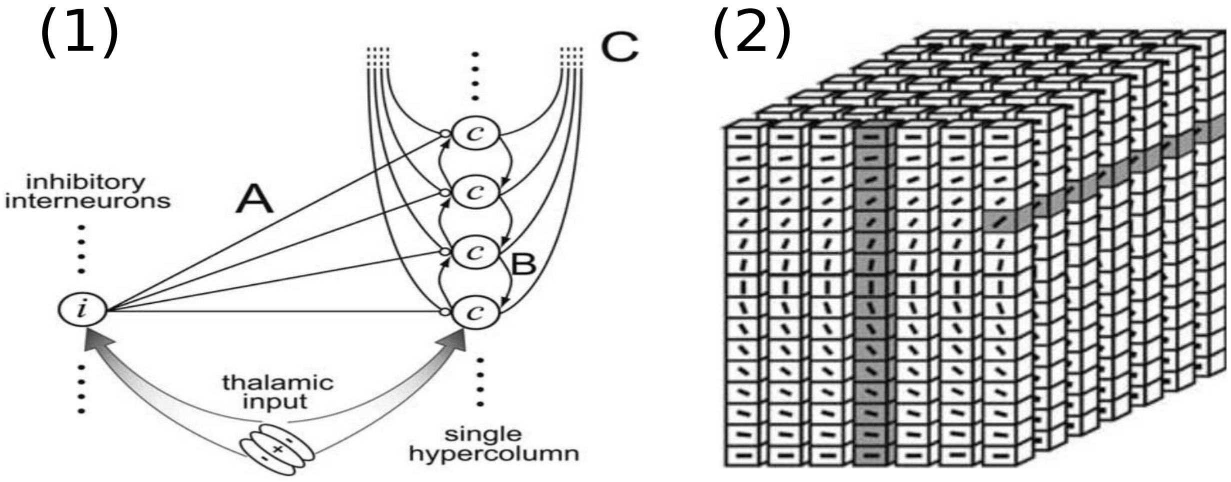

Figure 2.10: **Contour integration model of Ursino and La Cara [2004a].** (1) Intra-cortical connections of the model. Feedforward responses from the Thalamus are sent to both excitatory c and inhibitory i neurons. (A) Short-range inhibitory connections. i neurons send directed inhibition to c neurons. (B) Mid-range excitatory connections. c neurons send directed excitation to other c neurons, (C) Long-range inhibitory connections. c neurons also send omnidirectional inhibition to normalize the activity of other c neurons over a large area. (2) The full model is constructed using a spatial grid of 50x50 hyper-columns. At each spatial location (x, y) there is a hypercolumn of 16 excitatory neurons and 4 inhibitory neurons. Reprinted from Ursino and La Cara [2004a] with permission from Elsevier.

The cRFs of thalamic neurons are modeled as 2D Gabor functions,

$$R(x - x_c, y - y_c) = R_o \text{exp}\left(\frac{-\upsilon_1^2}{2\sigma_1^2}\right) \text{exp}\left(\frac{-\upsilon_2^2}{2\sigma_2^2}\right) cos(2\pi f \upsilon_2), \tag{2.5}$$

$$\upsilon_1(x - x_c, y - y_c, \theta_c) = (x - x_c)cos(\theta_c) + (y - y_c)sin(\theta_c),$$
$$\upsilon_2(x - x_c, y - y_c, \theta_c) = -(x - x_c)sin(\theta_c) + (y - y_c)cos(\theta_c),$$

where (x_c, y_c, θ_c) indexes a neuron centered at (x_c, y_c) and with an orientation preference of θ_c, σ_1 and σ_2 represents the spatial extent of the cRF in its preferred orientation (θ_c) and the orthogonal-to-preferred $(\theta_c + \frac{\pi}{2})$ directions respectively, f specifies the neuron's spatial frequency preference, and R_o is a fixed constant. Feedforward input to V1 cells is defined as the dot product of the image with the cRFs of Thalamic neurons followed by a non-linear activation function.

Three types of lateral connections are included in the model: (1) short-range oriented inhibition, W_i, mid-range oriented excitation, W_e, and long-range omnidirectional inhibition, W_l. The three types of connections model the three types of V1 contextual influences identified by Kapadia et al. [2000]. All lateral connections follow the general form,

$$W_j = W_{0,j} \text{exp}\left(\frac{-(\theta_c - \theta_h)^2}{2\sigma_{\triangle\theta}}\right) \text{exp}\left(\frac{-d_{\theta_c}^2}{2\sigma_{\theta_c}^2}\right) \text{exp}\left(\frac{-d_{\theta_c+\pi/2}^2}{2\sigma_{\theta_c+\pi/2}^2}\right). \tag{2.6}$$

Here, $j \in (e, i, l)$ indexes the type of connection, and W_j specifies the connection strength between a presynaptic neuron at (x_h, y_h, θ_h) and a target postsynaptic neuron at (x_c, y_c, θ_c). The first Gaussian term, $\exp\left(-(\theta_c-\theta_h)^2/2\sigma_{\triangle\theta}\right)$, models the influence of preferred orientation differences between the two neurons. Its impact is highest for two neurons with identical orientations and drops monotonically as the difference between their preferred orientations diverge. The second two Gaussian terms $\exp\left(-d_{\theta_c}^2/2\sigma_{\theta_c}^2\right)$ and $\exp\left(-d_{\theta_c+\pi/2}^2/2\sigma_{\theta_c+\pi/2}^2\right)$ model the impact of distance in the preferred and orthogonal-to-preferred directions of the postsynaptic neuron. To get these two distances, first a vector from the postsynaptic neuron to the presynaptic neuron is constructed. The magnitude of this vector is given by $d = \sqrt{(x_c - x_h)^2 + (y_c - y_h)^2}$ while its orientation is defined by $\psi = \arctan\left(y_c-y_h/x_c-y_h\right)$. Next, the vector is projected onto the preferred orientation axis, $d_{\theta_c} = d\cos(\psi - \theta_c)$ and the orthogonal-to-preferred axis, $d_{\theta_c+\pi/2} = d\sin(\psi - \theta_c)$ of the target neuron. $W_{0,j}$ is a constant that represents the strength and direction of the connection (excitatory connections have positive while inhibitory connections have negative values). σ parameters model the extent of these component Gaussians in various dimensions. Finally, the net input from a specific type of connection to a target cell is found by summing contributions from all other neurons in the model. The contribution of a presynaptic neuron is defined as its feedforward input multiplied by its connection weight as defined above.

The intra-cortical connections of the model are shown in Figure 2.10 left. In contrast with the model [Li, 1998], both excitatory and inhibitory neurons receive feedforward input. Another notable difference is the use of short directed lateral inhibitory neurons, W_i. These connections are designed to model the contrast (feedforward input strength) invariant responses of V1 neurons. Contrast invariant V1 neurons respond similarly to edges regardless of input brightness; they preserve their orientation specificity across a wide range of input contrast. On the other hand, simple Gabor filters respond similarly to a bright nonoptimal edge and to a dim optimal edge. To model contrast invariant edge detectors, inhibitory neurons respond to visual inputs and send regulatory signals to nearby excitatory neurons [Ursino and La Cara, 2004b, Ferster and Miller, 2000]. These interneurons are broadly tuned and are activated with any visual input. Consequently, their dependence on orientation preference differences in Equation 2.6 is removed. W_i connections are also used to model suppression from neighbors that are unlikely to be simultaneously involved in smooth contours. Accordingly, spatially they spread more in the $\theta_i + \frac{\pi}{2}$ than in the θ_i direction ($\sigma_{\theta_i+\frac{\pi}{2}} > \sigma_{\theta_i}$), see Figure 2.11A.

A contour enhancing association field is modeled using mid-range excitatory connections, W_e. W_e depends on both distance and the orientation difference between pre-/post-synaptic neurons. They cover a larger area compared with W_i and extend further in the θ_e direction ($\sigma_{\theta_e} > \sigma_{\theta_{e+\frac{\pi}{2}}}$), see Figure 2.11B. Lastly, W_l connections model contrast normalization after the large omnidirectional inhibition field found in [Kapadia et al., 2000]. As the spatial extent of these connections is omnidirectional $((\sigma_{\theta_c} = \sigma_{\theta_{c+\frac{\pi}{2}}}))$, the projections in the preferred and orthogonal-to-preferred terms in Equation 2.6 are combined.

Finally, the dynamics of the excitatory neurons are modelled as,

$$u(x_c, y_c, \theta_c) = g(x_c, yc, \theta_c) + i(x_c, yc, \theta_c) + e(x_c, yc, \theta_c) + l(x_c, yc, \theta_c), \tag{2.7}$$

$$\tau \frac{dr(x_c, y_c, \theta_c)}{dt} = -r(x_c, y_c, \theta_c) + S\left(2u(x_c, y_c, \theta_c)\right), \tag{2.8}$$

where, $g(x_c, yc, \theta_c)$ is the thalamic at (x_c, y_c, θ_c), $i(x_c, yc, \theta_c)$ is the summed contribution of all short range inhibitory connections, $e(x_c, yc, \theta_c)$ is the contribution of all mid-ranged excitatory connections, $l(x_c, yc, \theta_c)$ is the contribution of all long-range inhibitory connections, $r(x_c, y_c, \theta_c)$ represents the firing rate (output) of the excitatory neuron, τ is its time constant and S is the sigmoid non-linearity.

Through a sensitivity analysis, the influence of each lateral connection and its parameters was investigated. It was found that long-range omnidirectional inhibition was essential

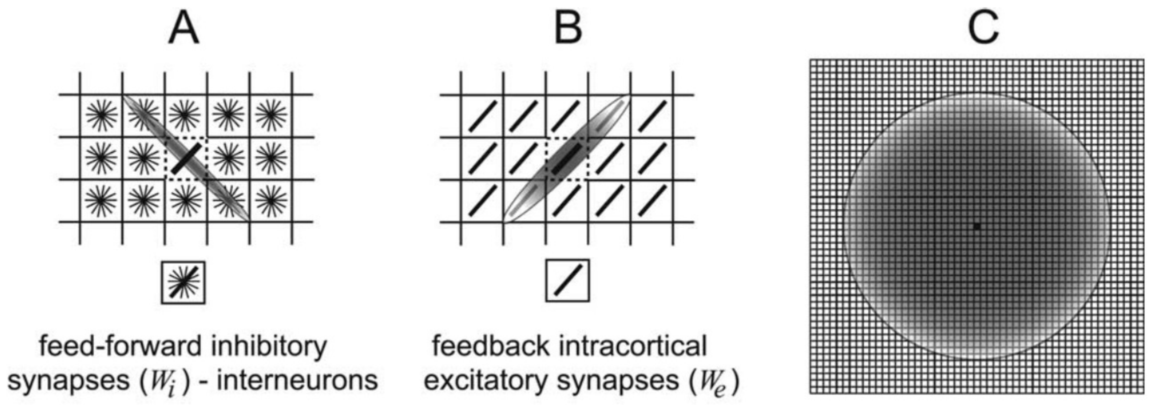
A
B
C
feed-forward inhibitory synapses (W_i) - interneurons
feedback intracortical excitatory synapses (W_e)

Figure 2.11: **Sample lateral connections of Ursino and La Cara [2004a] model**. Short bold bar shows orientation preference of an excitatory neuron. Shaded regions show lateral connections of various types. (A) Short range oriented inhibitory connections, W_i. These model suppression in the orthogonal to the preferred direction. (B) Mid-range excitatory connections. These model enhancement of co-linearly located neighbors. The dependence on orientation differences between pre and post synaptic neurons is not shown. (C) Long range omnidirectional inhibitory connections. Reprinted from Ursino and La Cara [2004a] with permission from Elsevier.

for noise reduction. However at high strengths, weak contour segments (those with higher curvature) disappeared. Better noise reduction is possible with larger inhibition fields, but too large values tend to interfere with nearby contours that may themselves not have received similar enhancement. Increasing the strength of excitatory connections resulted in broken contours; straight contour segments were boosted more which caused contrast normalization to suppress weaker segments. Increasing the spread of excitatory connections in the preferred direction cause contours to over run their boundaries especially at points of large curvatures (corners). However at reasonable strengths these connections enabled the fill-in of missing fragments. Increasing the spread of the excitatory connections in the orthogonal to the preferred direction, had the effect of thickening contours. Unfortunately, an analysis of oriented suppressive connections was not preformed.

2.4.3 Piech et. al. - 2013 - Model

The model of Piëch et al. [2013] expands on the role of top-down feedback in V1 contour integration. Lateral V1 interactions are still the dominant connections in realizing contour integration, however, their efficacy is strongly influenced by top-down signals. Li et al. [2006, 2008] found that V1 contour enhancement was stronger when monkeys performed a contour integration task but much weaker when the monkey was involved in an unrelated task. This was interpreted as variable lateral connection strengths that are steerable by external top-down signals.

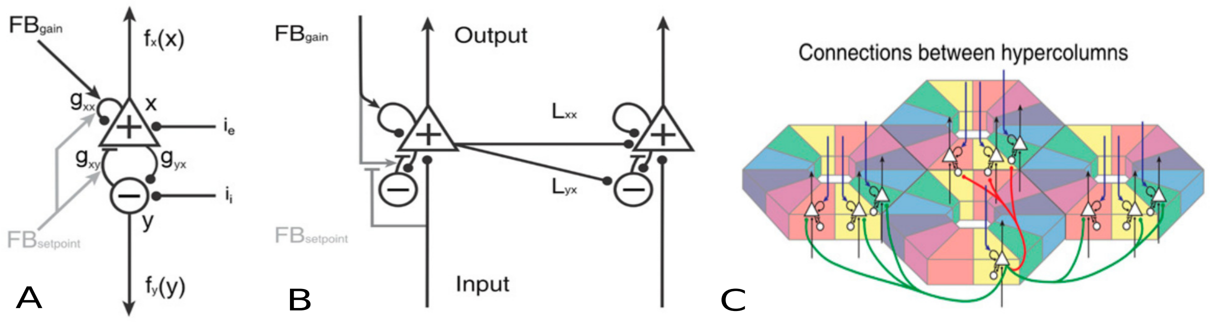

Figure 2.12: **Contour integration model of of Piëch et al. [2013]**. (A) The basic unit of the model is an pair of excitatory (+) and inhibitory nodes (−). Connections ending in a circle are excitatory, while those ending with a bar are inhibitory. g_{xx}, g_{xy}, g_{yx} are conductances of the $+ \to +, - \to +, + \to -$ connections respectively. i_e and i_i are non-local inputs and include lateral inputs from neighboring nodes, background activity and external feedforward inputs. Piëch et al. [2013] considered three models. All models had identical connection structures but differed in their parameters and how their neural dynamics were set up. Only parameters of the conductance model with subtractive inhibition are shown. For the other current based models, conductances are replaced by corresponding currents parameters ($g_{xx} \to J_{xx}$ etc). (B) Lateral connections of the model. + nodes connect to both + and − nodes, while − nodes only connect with the local + node. FB_{gain} is top down external signal that increases the self excitation gain of + nodes. FB_{setpoint} is an internal self regulation term that adjusts $- \to +$ connections proportional to its inputs. (C) (best viewed in colour) The full model consists of a hexagonal grid of hypercolumns defined over non-overlapping spatial locations. Each hypercolumn contains 12 (+, −) pairs with gradually changing orientation preferences. Different colours correspond to different orientation preferences. Green connections correspond to + while red connections correspond to − connections. Figure reproduced from [Piëch et al., 2013].

The architecture of their model is similar to the [Li, 1998] model. The basic unit of the model is a pair of reciprocally connected excitatory (E) and inhibitory (I) nodes (Figure 2.12A). Each node pair represents the population activity of neurons in the superficial layers of a V1 orientation column. E nodes selectively connect to other E and I nodes in nearby orientation columns (Figure 2.12B). Slightly different from the [Li, 1998] model, direct inhibition to E nodes is only received from the I node in the same node pair and not from other I nodes in the hypercolumn. Furthermore, no omnidirectional inhibitory normalization was considered.

At each spatial location (x, y) there exists a hypercolumn of node pairs whose orientation preferences (θ) gradually change to cover all orientations. The full model consists of a hexagonal grid of 220×220 hypercolumns (Figure 2.12C).

Oriented lateral connections are modeled as,

$$L_{xx} = I_{xx}\exp\left(\frac{-(\triangle x^2 + \triangle y^2)}{2d_\sigma^2}\right)\cos_{fwhm}\left(-\theta_2, \theta_1, \frac{\pi}{4}\right)\cos_{fwhm}\left(\frac{|\theta_1| + |\theta_2|}{2}, 0, \frac{\pi}{4}\right), \tag{2.9}$$

$$L_{yx} = I_{yx}\exp\left(\frac{-(\triangle x^2 + \triangle y^2)}{2d_\sigma^2}\right)\cos_{fwhm}\left(-\theta_2, \theta_1, \frac{\pi}{4}\right)\cos_{fwhm}\left(\frac{|\theta_1| + |\theta_2|}{2}, \frac{\pi}{2}, \frac{\pi}{4}\right). \tag{2.10}$$

Here, L_{xx} is the connection strength between two neighboring E nodes, L_{yx} is the strength of the connection from an E to an I node, $(\triangle x, \triangle y)$ is the separation distance between the two considered neurons, d_σ is the spatial extent of the lateral association field, I_{xx}, I_{yx} are the full strengths of these connections, θ_1, θ_2 are the orientation preferences of the pre/post-synaptic neurons and $\cos_{fwhm}(\theta, \theta_{opt}, \theta_{fwhm})$ is a tuning curve given by,

$$\cos_{fwhm}(\theta, \theta_{opt}, \theta_{fwhm}) = \begin{cases} \frac{1}{2}\cos\left(\frac{\pi(\theta-\theta_{opt})}{\theta_{fwhm}}\right) + \frac{1}{2} & |\theta - \theta_{opt}| \leq \theta_{fwhm} \\ 0 & \text{otherwise} \end{cases}, \tag{2.11}$$

where θ_{opt} is the preferred orientation and θ_{fwhm} is the orientation at full-width-half-maximum (FWHM) of the tuning profile. Equations 2.9 and 2.10 model excitation and suppression association fields. Two orientation dependent tuning curves are included in each of them. The first decreases the strength of neighbors as their orientation preference diverges from the preferred orientation of the target postsynaptic neuron and the second is a curvature penalty that enhances co-linear curves more than curve-linear ones. There is also Gaussian decay of connections strengths with distance between the two nodes. An example of these connections for a neuron with a horizontal orientation preference is shown in Figure 2.13.

Although the architecture and connections were similar to [Li, 1998], the neural dynamics were constructed differently. Three different models were considered: (1) current

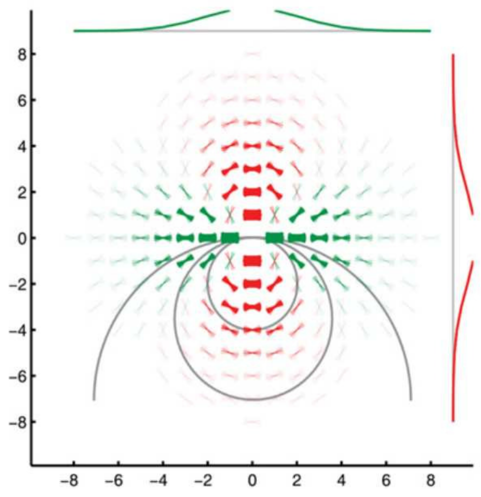

Figure 2.13: **Sample lateral connections of Piëch et al. [2013] model neuron with a horizontal orientation preference**. Best viewed in colour. Edge thickness represent the strength of the connection. Green edges model the excitatory association field while red edges form the suppression association field. Figure reproduced from Piëch et al. [2013].

based model with subtractive inhibition (CurrSI), (2) current based model with divisive inhibition (CurrDI) and (3) conductance based model with subtractive inhibition (CondSI). The CurrSI model is the most similar to the [Li, 1998] model and involves inhibitory nodes exerting a subtractive influence on excitatory nodes,

$$\dot{x} = \frac{1}{\tau_x}\left(-x + J_{xx}f_x(x) - J_{xy}f_y(y) + I_{0e} + I + \sum_{x'} L_{xx'}f_x(x^{'})\right), \tag{2.12}$$

$$\dot{y} = \frac{1}{\tau_y}\left(-y + J_{yx}f_x(x) + I_{0i} + \sum_{x'} L_{yx'}f_x(x^{'})\right), \tag{2.13}$$

where x and y are membrane potentials of E and I nodes, J_{xx}, J_{xy}, and J_{yx} are strengths of the $E \rightarrow E, I \rightarrow E$, and $E \rightarrow I$ connections, respectively, $f_.(.)$ are non-linear activation functions that transforms membrane potential to firing rates, $\tau_.$ are membrane time constants, $I_{0.}$ represents the background activity of a node, I is the external input from a previous layer, $L_{xx'}$ is the connection strength from a neighboring E node x' to x, and $L_{yx'}$ is the connection strength between E node x' to y.

In the CurrDI model, I nodes interact divisively with E nodes,

$$\dot{x} = \frac{1}{\tau_x}\left(-x + \frac{J_{xx} f_x(x) + I_{0e} + I + \sum_{x'} L_{xx'} f_x(x^{'})}{1 + J_{xy} f_y y(y)}\right), \tag{2.14}$$

$$\dot{y} = \frac{1}{\tau_y}\left(-y + J_{yx} f_x(x) + I_{0i} + \sum_{x'} L_{yx'} f_x(x^{'})\right). \tag{2.15}$$

In the conductance based model (CondSI), feedforward and lateral inputs were configured as conductances rather than constant currents terms. This was done by multiplying inputs with the difference between a nodes activity and its equilibrium potential. As a result of this, their effective strengths varied based on how close they were to a node's equilibrium potential,

$$\dot{x} = \frac{1}{\tau_x}\left[-x + \left(g_{xx} f_x(x) + \sum_{x'} L_{xx'} f_x(x^{'}) + G_{0e} + G\right)(v_e - x) + g_{xy} f_y(y)(v_i - x)\right], \tag{2.16}$$

$$\dot{y} = \frac{1}{\tau_y}\left[-y + \left((g_{yx} f_x(x) + \sum_{x'} L_{yx'} f_x(x^{'}) + G_{0i})\right)(v_e - y)\right], \tag{2.17}$$

where, all current (I) parameters in the CurrSI model have been replaced by conductances (g/G), and $v_.$ are resting potentials of nodes.

Piëch et al. [2013] analyzed potential mechanisms higher layers feedback could use to change the efficacy of lateral connections. In contrast with the Li [1998] model, feedback modulated self-excitation, J_{xx}, of E nodes rather than entering as an external input to I nodes. Moreover, an internal mechanism was also used to adjust inhibitory to excitatory connection strengths, J_{xy}, in tandem. This approach allowed the effective strength of lateral connections to be strengthened without affecting responses to feedforward inputs. Unfortunately, feedback was considered a separate external input to the system, and it was not specified how specific V1 neuron pairs could be targeted by higher layers.

2.4.4 Hu and Niebur - 2017 - Model

The model of Hu and Niebur [2017] is a hierarchical model that incorporates parts of V1, V2 and V4 cortices. This multi-purpose model assimilates contour integration, figure ground separation and external attentional influences into a single coherent model that can replicate a range of neurophysiological results. The model postulates that contour integration results from a combination of V1 lateral connections and feedback from V4 neurons. Contrasting previous models, feedback is generated internally through the use of proto-object V4 neurons that send modulatory signals back to V1 neurons. In the context of contour integration, these V4 neurons respond to oriented linear contours. This additional connection replicates the interplay between V1 and V4 neurons and can account for the simultaneous development of contour responses in both layers, [Chen et al., 2014]. Moreover, coarse top-down signals, such as object based attention, can utilize these same connections to target spatially precise V1 neurons. Finally, Hu and Niebur [2017] demonstrate how this accounts for larger enhancement gains when a contour is attended to but are still present in the unattended state [Li et al., 2006, 2008].

The architecture of the model is shown in Figure 2.14. As can be seen, component units are densely connected. The following brief description is restricted to components involved in contour integration only. Input to the model consists of a binary map indicating the presence of oriented edge fragments over a discrete grid of 64x64 locations. At each spatial location, there exists a V1 hypercolumn that contains four edge detecting (E) units. The orientation preferences of component edge units uniformly change to cover the range $(0, 180^\circ)$. As is the case for all units of the model, each edge detecting unit represents a population of neurons with identical tuning profiles. E cells send their outputs to inhibitory cells, IE, in their vicinity. These omni-directional connections are modeled as 2D Gaussians and depend only on the distance between neurons. Following Stettler et al. [2002], the spatial extent of these connections was set to 8 times the cRF of E cells. IE cells reciprocally connect to E cells in their proximity. These connections are modeled similar to E $\rightarrow$ IE connections but with opposite polarity.

E cells also connect to other E cells,

$$W(E^{o_1}_{x1,y1})(E^{o_2}_{x2,y2}) = N_{etoe} * \begin{cases} \exp\left\{\frac{-(x_1-x_2)^2}{2etoe^2_{sd}}\right\} & \text{if } o_1 = o_2 = 0,\ y_1 = y_2 = 0 \\ \exp\left\{\frac{-(x_1-x_2)^2}{2etoe^2_{sd}}\right\} & \text{if } o_1 = o_2 = \frac{\pi}{4},\ x_1 = y_1,\ x_2 = y_2 \\ \exp\left\{\frac{-(y_1-y_2)^2}{2etoe^2_{sd}}\right\} & \text{if } o_1 = o_2 = \frac{\pi}{2},\ x_1 = x_2 = 0 \\ \exp\left\{\frac{-(y_1-y_2)^2}{2etoe^2_{sd}}\right\} & \text{if } o_1 = o_2 = \frac{3\pi}{4},\ x_1 = -y_1,\ x_2 = -y_2 \end{cases}, \tag{2.18}$$

where $E^{o_1}_{x1,y1}$ is a presynaptic edge detecting unit at (x_1, y_1) with orientation preference o_1, $E^{o_2}_{x2,y2}$ is a similarly defined postsynaptic edge unit, N_{etoe} represents the strength of the connection and $etoe_{sd}$ its spread. Compared with the lateral facilitation structures of previously described models, this is a simpler structure; it can only enhance co-linear contours in the 4 considered orientations. The spatial extent of lateral facilitation was identical to contrast normalizing, IE → E connections. Oriented suppression in the orthogonal to the preferred direction is not included.

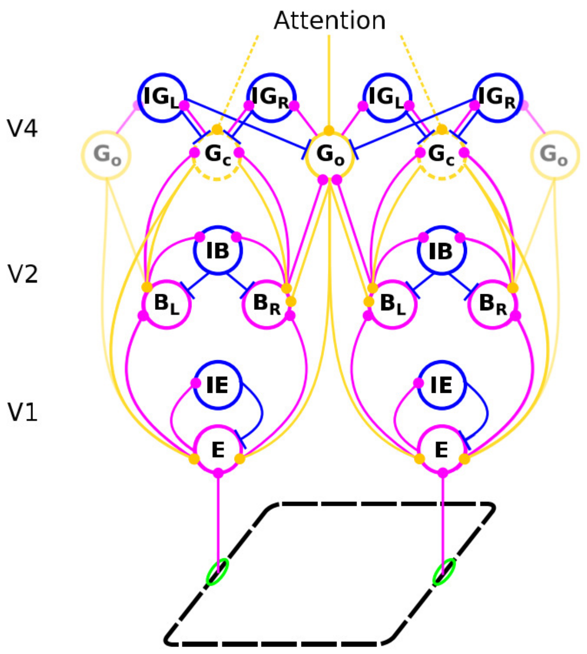

Figure 2.14: **Contour integration model of Hu and Niebur [2017].** Best viewed in colour. E cells respond to edges of a particular orientation. Border ownership (B) cells similarly respond to edges but react differently if the foreground object they represent lies to the left or right of their receptive field. Grouping (G) cells respond to configurations of edges representing proto-objects, either simple objects (G_o) or contours (G_c). IE, IB and IG cells represent inhibitory cells. Magenta, blue and yellow lines represent feedforward, inhibitory lateral and feedback connections respectively. Connections ending in a circle are excitatory, while those ending with a bar are inhibitory. Lateral excitatory connections are not shown. Reprinted from Hu and Niebur [2017] with permission from Springer Nature.

E units sum their inputs before passing them through a nonlinear activation function. E cells send their feedforward outputs to border ownership (B) cells in V2 [Zhou et al., 2000]. B cells have similar properties to V1 edge detecting cells they are connected with, but respond differently if the foreground shape they are representing is located to the left or right of their cRF centers. Interestingly, lateral connections, similar to those in V1, are included in V2 as well. These connections propagate border ownership information between neighbors and play an integral role in separating foreground objects from the background [Zhaoping, 2005]. B cells send their feedforward outputs to contour grouping cells in V4.

E cells receive modulatory feedback from V4 cells in two ways. Once V4 grouping cells (G) cells are activated, they send direct feedback to V1 E cells. V4 G cells also send feedback to V2 B cells which in turn send feedback to V1 E cells. Differential equations are set up over the activities of all V1, V2 and V4 cells in the model to study their temporal properties. Hu and Niebur [2017] also investigated the impact of the internal and external feedback in their model. Removing the feedback from V4 to V1 decreased responses in both cortices. However, degradation was greater in V1 than in V4. On the other hand, external attentional signals, which were received by V4 G cells, boosted responses comparatively more in V4 than in V1, consistent with the results of Chen et al. [2014].

2.4.5 Discussion

Contour integration is a low-level neural phenomenon that assists the ventral visual stream in identifying what objects are in the visual environment. While the differential equation models described in this section can account for many neurophysiological properties, they do not provide much insight on how contour integration assists the ventral visual stream. With these models, it is not possible to see what specific tasks, under what conditions and by how much does contour integration contribute to the high-level goals of the ventral visual stream.

Another limitation of these models is that they do not address scalability. Typically, they are designed using a single type of edge extractor (a single Gabor type with different orientations) at non-overlapping spatial locations and expect as inputs, a single type of edge at predetermined locations. However, in natural images, edges come in a variety of shapes and forms and can occur anywhere. To handle natural images, similar to the ventral visual stream, it is necessary to address scalability. Moreover, the ventral stream uses many other cues, such as colour and textures, to detect objects. Simply scaling up these models is not only computationally expensive but cannot account for these other cues.

Even for the limited type of data these models are designed to handle, results are usually presented with a few exemplar stimuli that showcase particular features of the model. Different models focus on different sets of neurophysiological properties and therefore use different stimuli. Typically, models are only compared on the basis of how many properties they can account for. It is difficult to quantitatively compare models even over the range of stimuli they support.

Finally, parameters and architectures of these models are fixed during the design phase. The choice of their settings are based on neurophysiological and neuroanatomical studies. Studying neurons in the brain is difficult and only offers a partial view of all the processing involved [Olshausen and Field, 2006]. Moreover, techniques used to study the structure of neurons also have limitations. For example, in axon labeling investigations of lateral V1 connections (see Section 2.2.4) it is not possible to get a clear picture of the distribution of connections near injection sites [Stettler et al., 2002, Angelucci et al., 2002]. In these models, the findings, biases and assumptions of these investigations are built into the model.

In Chapter 4, I present a model that can address these limitations while retaining the ability to replicate many neurophysiological properties of contour integration, similar to these models. Moreover, many of the parameters in the model are learnt and can be used to validate some of the assumptions made in these models.

Chapter 3

Artificial visual systems

Artificial Intelligence (AI) is a field of computer science that is concerned with the theory and development of machines capable of performing tasks that are normally associated with *intelligent* beings. Machine Learning (ML) is a sub-field of AI that focuses on teaching machines to perform input-to-output mapping tasks by extracting patterns from raw data rather than using a hard coded set of decision rules [Goodfellow et al., 2016]. Representation learning is an approach to ML where models can additionally learn different representations of the input data to simplify the learning of input-to-output mappings [Goodfellow et al., 2016, LeCun et al., 2015]. Artificial Neural Networks (ANNs) are a special class of representation learning models that are inspired by networks of neurons in the brain. Deep Neural Networks (DNNs) are hierarchical multi-stage ANNs that can learn multiple levels of representation that build on each other.

In this chapter, I review ANNs, the other major focus of my research. I begin by briefly describing the fundamentals of ANNs. Second, I discuss Convolutional Neural Networks (CNNs), a type of ANN model that specializes in the processing of visual data. Third, I describe Recurrent Neural Networks (RNNs). RNNs are another type of ANN model that specialize in the processing of sequential data. However, they have also been used to process static spatial data. As I am interested in modeling dynamic contour integration in ANNs, the combining of RNNs and CNNs is a good starting point. In fact, many researchers actively working at the intersection of AI and computational neuroscience consider recurrent convolutional neural networks as currently the best artificial models of the ventral visual stream [Kubilius et al., 2019]. Fourth, I highlight the work that has been done in comparing ANNs and the brain's visual system. I describe both the similarities as well as the differences. Finally, I conclude this chapter with precautions others have suggested that must be considered when comparing DNNs with the visual system.

3.1 Artificial Neural Networks

The goal of an Artificial Neural Network (ANNs) is to learn a mapping between its inputs and outputs. A common task for ANNs is classification, where inputs need to be classified into a smaller number of categories. In the context of classification, the function that needs to be learnt is the conditional probability distribution, $p(y|\mathbf{x})$; the probability of label, y, given an input, $\mathbf{x}$. For complicated tasks, this function does not have a closed form mathematical expression. ANNs approximate this mapping using a combination of many small well-defined functions or neurons.

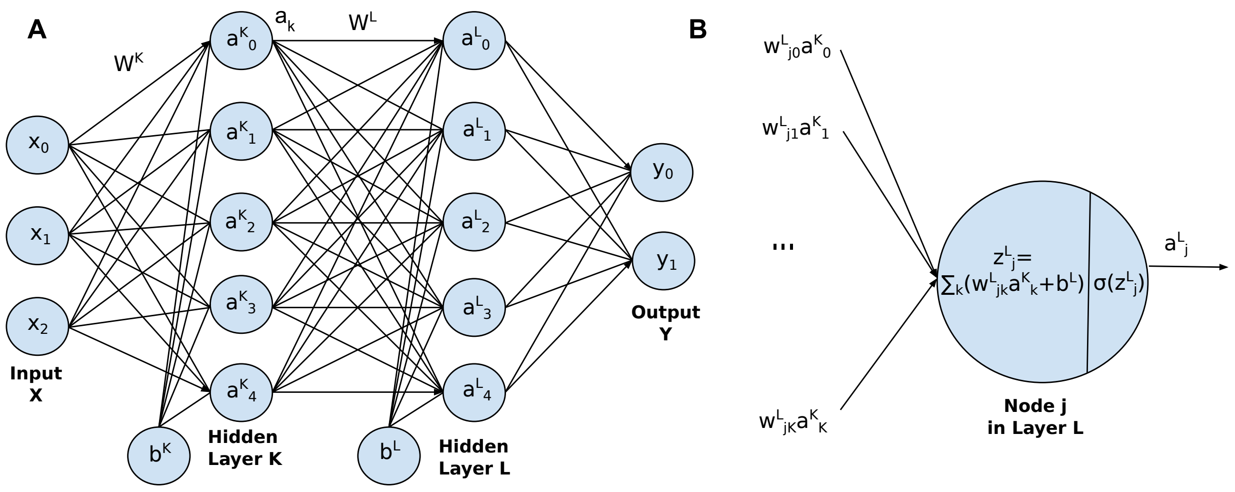

Figure 3.1: **Feedforward artificial neural network**. (A) An example network with 2 hidden layers. Each layer contains multiple nodes. Each node receives inputs from nodes in a preceding layer. Each node sends its outputs to nodes in the succeeding layer. (B) Single network node. The pre-activation of node j in layer L, the weighted sum of its inputs, is given by $z_j^L = \sum_k w_{jk}^L a_k^{L-1} + b^L$, where w_{jk}^L is the weight of the connection from node k in layer $L-1$ to node j in layer L, a_k^{L-1} is the output activation of node k in the previous layer and b^L is a bias term. The output activation of node j^L is its pre-activation passed through a nonlinear activation function, $a_j^L = \sigma(z_j^L)$. For an entire layer, this can be expressed compactly as, $\mathbf{a}^L = \sigma(W^L\mathbf{a}^{L-1} + b^L)$. Here, $\mathbf{a}^L$, W^L are the feature map and connection matrix of layer L, respectively.

The architecture of a basic feedforward ANN is shown in Figure 3.1A. It consists of several artificial neurons or nodes and the set of connections between them. These connections are inspired from synapses between neurons in the brain. Each connection is associated with a weight whose value represents the strength of the connection between attached units. Nodes are typically grouped into layers and the entire network is a layer-wise arrangement of interconnected nodes. For example, the model shown in Figure 3.1A is composed of 3 layers; 2 hidden layers and 1 output layer. Information flows from the input to the output layer through a series of hidden layers in the forward direction only. Nodes receive inputs only from nodes in the previous layer; there are no intra-layer lateral connections nor any feedback connections from neurons in subsequent layers.

Each node computes a weighted sum of its inputs (input or presynaptic node's activity multiplied by its connection weight) and then uses an activation function to map its stimulation to an output (see Figure 3.1B). Typically, a non-linear activation function is used. Non-linear activation functions allow the network to detect complex non-trivial patterns in inputs. Common types of non-linear activations functions include: the Rectified Linear Unit (ReLU), the Sigmoid Function, and the Hyperbolic Tangent Function.

This simple operation is repeated at each location within a layer. Collectively this is represented as,

$$\mathbf{a}^L = \sigma(W^L \mathbf{a}^{L-1} + b^L), \tag{3.1}$$

where $\mathbf{a}^L$ is the outputs of all nodes in layer L and is commonly referred to as the activation or feature map of a layer, W^L is a connection matrix that includes the weights of all connections of all nodes in layer L with all their neighbors in the previous layer, b^L is a constant output (bias), and $\sigma()$ is the nonlinear activation function.

In a basic ANN, each node is connected to all nodes in the previous layer. The parameter space (connection weights) of ANNs increases exponentially as the number of nodes in a layer increases. This large number of parameters increases the capacity of ANNs but also makes them prone to over-fitting, especially when trained on limited data. The large capacities of these models make them fit not only useful patterns in the data, but noise as well.

3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of feedforward ANN that specialize in the processing of visual images. CNNs address the problem of a large parameter space, that needs to be learnt, by incorporating apriori knowledge of the pixel representation of objects into their architecture. Specifically, two changes are made. First, each node is only connected to a subset of nearby nodes in the previous layer. The justification for this assumption is that object features are local structures; it is likely that far away nodes view different features that do not contain much information about the current feature. Second, within a layer, connection weights are tied across multiple locations. A node's connection weights can be thought of as a specific input combination that maximally excites it. Since natural images are generally fairly homogeneous (smooth regions with few edges), if a good input combination is learnt at one location, it is useful to look for it at other positions as well.

Incorporating these two assumptions significantly decreases the parameter space of CNNs. A convolutional layer's connection weights can be rewritten as a much smaller kernel. With this smaller kernel, a layer's feature map is found by striding the kernel across the layer and applying it at each node. This resembles the convolution operation and is the reason for their name. The activity of a convolutional layer can then be expressed as,

$$\mathbf{a}^L = \sigma(W^L \circledast \mathbf{a}^{L-1} + b^L), \tag{3.2}$$

where $\circledast$ is the convolutional operator and W is a kernel now that is much smaller than the number of units in a layer. For image processing, typically 2D kernels, inputs and layers are used.

Reducing the number of parameters also makes it computationally efficient to use large architectures. In particular, multiple hidden layers (deep architectures) can be stacked [LeCun et al., 2015, Goodfellow et al., 2016] on top of each other, enabling deeper layers to learn more complex features from lower layer feature maps. Furthermore, smaller kernels make the use of multiple kernels feasible, allowing multiple features to be simultaneously extracted at each layer.

A famous deep CNN model, AlexNet [Krizhevsky et al., 2012], is shown in Figure 3.2. Typically, in such large scale models, convolutional layers are used in conjunction with other layers. Pooling layers summarize the activations of nodes over a confined area for higher layers. These layers are specifically designed to develop tolerances to information less pertinent to the overall goal of the network. For example, classification networks frequently use max pooling layers. A max pooling layer sends the output of the most active node within a neighborhood to higher layers and helps make the network position invariant. Dense fully connected layers are also typically used at the deep end of the network. Nodes within these layers see the entire information flowing in the network which is useful when making a final decision on object labels. Another commonly included layer is Batch Normalization (BN) [Ioffe and Szegedy, 2015] (Not included in network shown in Figure 3.2). The use of multiple non-zero centered activation functions (such as ReLU and Sigmoid Functions) can result in feature maps that have non-zero means and non-unit variance. Batch normalization layers can restore feature maps to have zero mean and unit

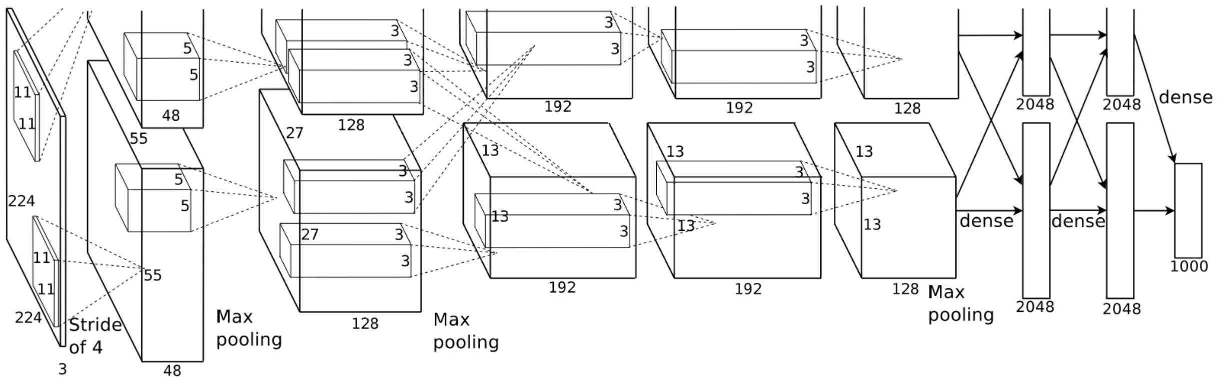

Figure 3.2: **AlexNet CNN model [Krizhevsky et al., 2012].** This deep CNN model consists of 5 convolutional layers and 3 fully connected (dense) layers. Inputs to the model are 227x227x3 sized RGB Images. Each convolutional layer extracts multiple features. Large rectangular boxes represent the activation volume of a layer, the set of feature maps of a layer. Features maps are also referred to as channels. Small rectangular boxes are the kernel sizes used at each layer.

variance and have been shown to improve performance (but see Brock et al. [2021]).

The last layer of classification networks contain the same number of nodes as the number of classes of objects they are trained to predict. Furthermore, a special nonlinear activation function (e.g. Softmax Function) that converts network outputs (logits) to probabilities of each class, is used. For a given input, once a network's probabilistic predictions for each class is known, it can be directly compared with the one hot (1 for the true class, 0 for all others) ground truth label for that input.

3.2.1 Training

Teaching a CNN to approximate $p(y|\mathbf{x})$ is called training. This involves adjusting connection weights such that the network's predictions match ground-truth labels. Many types of training methods exist. For the task of object classification, typically supervised training is used. In supervised training, a large dataset of inputs and their corresponding ground truth labels are provided to the network to train with. Once trained, network predictions are tested on held-out or unseen inputs to see if the model learnt good features that generalize to data outside the training set.

Network probabilistic predictions and ground truth labels can be quantitatively compared using a cost (loss) function. A typical cost function for object classification is the cross entropy loss function,

$$L_{CE}(x) = -\sum_{i=n}^{N} p_i \log(q_i), \tag{3.3}$$

where p_i is the ground truth label for object class i and q_i is the network's prediction for the same class and N is the total number of classes. The cross entropy function measures the difference between the probability distribution of the network and the ground truth labels of the data.

Given the quantitative difference, the parameters of the network need to be adjusted such that the next time the network encounters a similar input, the difference is smaller. The large parameter space of these models makes this a nontrivial optimization problem. As there is no direct connection between the loss function and weights in earlier layers, training is especially hard for deep networks. The back propagation(BP) algorithm [Rumelhart et al., 1986] provides an elegant solution by allowing error gradients to be propagated through the entire network in a single backward pass. First, error gradients with respect to the weights in the deepest layer are computed. Next, error gradients with respect to the weights of the preceding layer are expressed in terms of the already computed gradients in the current layer. This process is repeated until error gradients of all weights in the network are found.

Once error gradients are known, gradient descent optimization is used to update network weights,

$$\theta_j \leftarrow \theta_j - \lambda \frac{\partial L_{CE}(x)}{\partial \theta_j} \tag{3.4}$$

where j is some connection weight in the network, ${}^{\partial L_{CE}(x)}/_{\partial \theta_j}$ is the error gradient of the loss function $L_{CE}(x)$ with respect to the weight j for input x and $\lambda \in [0, 1]$ is the learning rate which controls the amount weights are updated per input during training. In practice, it is common to use more efficient optimizers such as batch Stochastic Gradient Descent (SGD) or Adam [Kingma and Ba, 2014]. This process is repeated multiple times over multiple cycles (epochs) of the training dataset until the network settles to some optimal configurations. Typically, to speed up training time, inputs are grouped together into mini-batches and weights are updated after every mini-batch using batched averaged gradients.

3.2.2 More advanced CNNs

Since the conception of CNNs [LeCun et al., 1998] and their break through performance in object classification [Krizhevsky et al., 2012], innumerable improvements have been made.

In fact, it is an active line of research that has surpassed human level performance on object classification [He et al., 2015] (at least on clear natural images that contain typically centered and unobstructed views of objects). Here I highlight a few improvements that have lead to substantial improvements on state-of-the-art (SOTA) performance and have been widely adopted by the research community.

In [He et al., 2016] the Residual Network (ResNet) model was introduced[1]. ResNets use identity (skip) connections across layers. The main motivation for using these skip connections was to address the vanishing gradient problem of very deep networks. As networks get deeper, the repeated application of multiplications in the back propagation of error gradients results in very small gradients, especially at the shallow end of networks. These small gradients make it difficult to adjust network weights. By adding skip connections, error gradients of shallow layers parameters do not need to flow though every higher layer and can reach them with fewer impediments. The use of these skip connections allowed ResNet models to reach depths of hundreds of layers which in turn helped them achieve SOTA performance when they were proposed.

Increasing depth is one way of increasing network capacity to improve performance. There are other ways of improving network capacity including: increasing network width (typically interpreted as the number of channels of a layer) and input resolution (the size of input images). Images in large publicly available datasets come in all sizes. It is common practice to include a preprocessing step that resizes them to a standard size before feeding them into models. Large image sizes allow the network to learn more fine grain features but comes with a cost of an $O(N^2)$ increase in computational complexity. Tan and Le [2019] found that scaling any one dimension independently quickly saturates performance. However, if all three are jointly scaled under the constraint that product of the increases to depth, the square of the width and the square of the resolution was ≈ 2, higher performance can be achieved. After finding a base architecture using an automated network architecture search algorithm [Zoph et al., 2018, Tan et al., 2019], Tan and Le used their network scaling method to develop a new family of models, EfficientNets. EfficientNets achieve higher performance while using an order-of-magnitude fewer parameters compared to their counterparts. Moreover, these have consistently featured on the the ImageNet classification leadership board, `https://paperswithcode.com/sota/image-classification-on-imagenet`.

CNNs have also been used for a wide variety of other vision tasks including object and instance segmentation [He et al., 2017, Redmon and Farhadi, 2018, Liu et al., 2016,

[1]ResNets are family of model. Many specific architectures, that differ in the number of layers and hence the number pf parameters they contain, have been proposed. A particular ResNet architecture is specified by including the number of layers it contains with its name (e.g. ResNet18, ResNet50 and ResNet152).

Ronneberger et al., 2015], contour/edge detection [Xie and Tu, 2015, Shen et al., 2015, Poma et al., 2020, Linsley et al., 2018], artistic picture-to-picture style transfer [Gatys et al., 2015, 2016, Jing et al., 2019], and image restoration [Yeh et al., 2016, Pathak et al., 2016]. Typically, models used in these tasks modify their architectures based on the requirements and observed problems. However, it is common to build upon a main 'backbone' structure that has been pre-trained on large scale object classification tasks.

3.3 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) [Rumelhart et al., 1986] are another type of ANN that specialize in the processing of sequential data, such as audio signals, videos and textual data. In sequential data, more input becomes available to the network at each time step. Recurrent networks need to store information about past inputs in an internal memory and use it to generate outputs at each subsequent step. RNNs handle this by storing information in a hidden state and recurrently referring to it as well as new inputs in each time step.

While RNNs can be set up in many ways, the simplest form is expressed as,

$$\mathbf{h}_{t+1} = \tanh\left(\mathbf{W}_x\mathbf{x}_t + \mathbf{W}_h\mathbf{h}_t + \mathbf{b}\right), \tag{3.5}$$

where $\mathbf{h}_{t+1}$ is the hidden state activity at time $t+1$, $\mathbf{x}_t$ is the input at time t from the sequential input stream $(\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_T)$, T is the length of the sequence, $\mathbf{W}_\mathbf{x}$ is the connection matrix between input and hidden layer nodes, $\mathbf{W}_\mathbf{h}$ is the connection matrix that describes recurrent connections between hidden nodes, $\mathbf{b}$ is a bias term and tanh is the hyperbolic tangent non-linear activation function.

In the equation above and in the following discussion, a time step is used to refer to the next item in the sequence. The actual data does not need to be a time series, but only sequential in nature; RNNs work just as well with words in a sentence.

The architecture of the RNN described above is shown in Figure 3.3A. It is similar to a regular ANN but with additional recurrent connections between nodes in the hidden layers. These recurrent connections connect hidden layer nodes with all other hidden layer nodes (including the same node) but with a one step time delay.

In the first time step, the first item of the input sequence, $\mathbf{x}_1$, is fed into the network. The network stores useful information about the input in its hidden state. At each subsequent time step, the network receives the next item in the input sequence, $\mathbf{x}_t$, and uses it to adjust its stored hidden state information. This process repeats until a fixed number of iterations, T, (typically defined large enough to include the longest item in the training dataset) have passed. The output of the model, $\mathbf{y}_t$, depends on the type of task the network is designed to handle. For example, in classification tasks such as sentiment analysis [Medhat et al., 2014], typically a single class label is generated after the full input stream has been parsed. In language translation tasks [Sutskever et al., 2014], an input sequence is used to generate an output sequence. In these sequence-to-sequence mapping tasks, an output is generated at each iteration. There are even RNNs that work on static inputs and use them to generate a sequence of outputs, such as image captioning networks [Vinyals et al., 2015].

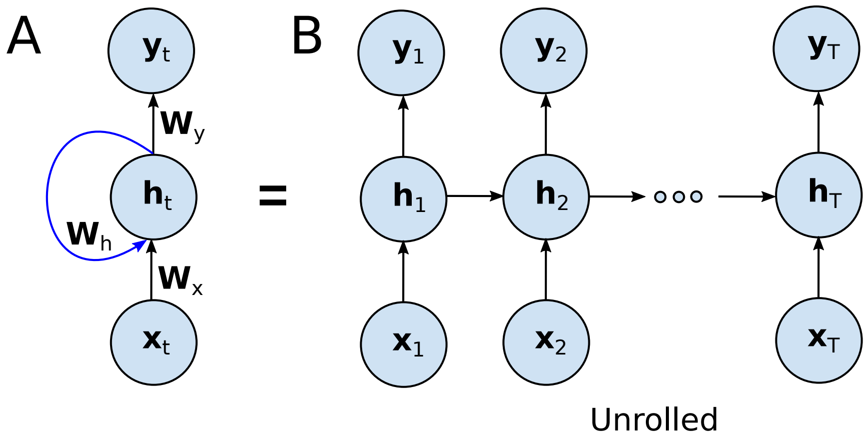

Figure 3.3: **Simple Recurrent Neural Network**. (A) Architecture of a simple recurrent neural network. At each recurrent step the hidden layer, $\mathbf{h}_t$, receives a new input $\mathbf{x}_t$ as well as hidden layer outputs of the previous time step. Activity propagates through the network for a predefined number of time steps, T. The output of the network, $\mathbf{y}_t$, depends on the task the RNN is designed to handle. Hidden layer outputs can be passed to output nodes every iteration or at the end of all iterations. $\mathbf{W}_x, \mathbf{W}_h, \mathbf{W}_y$ are the connection matrices between nodes in the input-to-hidden, hidden-to-hidden, and hidden-to-output layers respectively. (B) An across time steps unrolled view of the RNN shown in A. The hidden layer at each time step is similar to an individual layer in a deep feedforward network with the added constraint that parameters are shared across different layers.

Another way of looking at the processing of an RNN is to roll it out across time steps (see Figure 3.3B). When unrolled, the hidden layer at each iteration is similar to individual layers of a deep feedforward ANN but with the added constraint that weights of the network are shared across different layers. Increasing the number of iterations, expands the temporal depth of RNNs and allows them to detect long range relationships in input sequences. It does not increase network capacity (as the parameters are shared) but does increase the computational complexity and run time of models. The depth of RNNs can also be increased by stacking recurrent layers on top of each other, similar to how feedforward ANNs are made deeper. This increase expands the capacity of RNNs and allows them to detect more complex patterns.

RNNs are considered function approximators for dynamic systems, similar to how feedforward ANNs are considered function approximators of static functions. If equation 3.5 can be considered to be the difference equation obtained by using Euler's Method, $h(t+\delta t) \approx h(t) + \delta t^{dh}/_{dt}$, on a dynamical system that is described by an ordinary differential equation (ODE), and assuming $\delta t = 1$, the corresponding dynamical system is given by,

$$\frac{d\mathbf{h}}{dt} = \tanh\left(\mathbf{W}_x\mathbf{x}(t) + \mathbf{W}_h\mathbf{h}(t) + \mathbf{b}\right) - \mathbf{h}(t), \tag{3.6}$$

Most existing models of observed neurophysiological dynamic phenomenon, including contour integration, are ODE based computational models (see Section 2.4). The ability to relate simple RNNs directly to their ODE representation, makes them particularly interesting for modelling these phenomena in ANNs.

3.3.1 Training

RNNs are most commonly trained using the Back Propagation Through Time (BPTT) algorithm [Werbos, 1990]. The algorithm works similar to regular back propagation (BP) but operates on unrolled RNNs (Figure 3.3B). In fact, most existing machine learning frameworks implement both algorithms using the same function and implicitly distinguish between the two based on the type of network being trained. The rest of the training procedure including the type of training, cost functions and optimizers are similar to those used for training feedforward ANNs (see Section 3.2.1).

In practice RNNs have been found to be difficult to train. They are weak at detecting long-term relationships between input items that are separated by a large number of time steps. The activation preserved in hidden nodes across time steps, that is pertinent to the current sequence being processed, is sometimes call short term memory. Simple RNNs are said to have short short-term memory. There are two root causes for this issue: the vanishing and exploding gradients problems [Bengio et al., 1994, Pascanu et al., 2013]. During training, if gradients at the output of models are small, they become exponentially smaller when back propagated through time steps rendering them too small to make effective changes to network weights. On the other hand, if gradients are too large, multiplications during the backpropagation process causes them to exponentially grow or explode and results in large changes to network weights for small changes in the output. Both problems prevent RNNs from learning efficiently. Many solutions have been proposed to tackle these problems including: gradient clipping [Bengio et al., 1994], orthogonal weight initialization of recurrent weight matrices [Saxe et al., 2013], and new recurrent architectures (covered in the next section).

3.3.2 More advanced RNNs

In Hochreiter and Schmidhuber [1997], the Long Short Term Memory (LSTM) network was introduced. Different from simple RNNs, the model uses new multistage nodes inside its hidden layer. The internal architecture of these nodes was specifically designed to address the vanishing gradient problem. In addition to the hidden state, each composite node includes an extra cell state, where short term memory about the current sequence is stored. Another prominent feature of LSTM networks is the use of new learnable multiplicative gates that control the flow if information both in to and out of hidden and cell nodes.

The operation of an LSTM network is best understood by the set of equations that describes it. The equations from an update variant [Graves et al., 2013] are defined below.

Compute the input, forget and output gates,

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i), \tag{3.7}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f), \tag{3.8}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o), \tag{3.9}$$

where $\mathbf{x}_t$ is the input at time t, $\mathbf{h}_t$ is the hidden state activity at time t, $\mathbf{W}_{xi}, \mathbf{W}_{hi}$ are learnt connection matrices between input layer $\rightarrow$ input gate, hidden layer $\rightarrow$ input gate and $\mathbf{b}_i$ are bias terms. The connection matrices and biases for the forget and the output gate follow similar terminology. $\sigma()$ is the Sigmoid non-linear activation function which restricts gate values to the range $[0, 1]$.

Next, the cell state is updated by multiplying its forget gate with its previous value and mixing it with the input gate modulated total input to the cell state,

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c), \tag{3.10}$$

where $\odot$ is the Hadamard product.

Finally, the hidden state is calculated by applying the output gate on the updated cell state after it has passed though a tanh non-linear activation function,

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \tag{3.11}$$

The architecture of the LSTM was empirically designed to improve performance on sequential tasks. It succeeded in making RNNs better at learning long term dependencies. In fact, for quite some time after they were proposed, they achieved SOTA performance on many sequential tasks. Unfortunately, their architectures are quite complex and difficult to relate back to biology. They are generally not considered biologically plausible (but see Costa et al. [2017]).

Cho et al. [2014] introduced the Gated Recurrent Unit (GRU) network. GRU models can be considered simplified LSTM networks and were similarly designed to maximize performance. Like LSTMs, they use gates to control the flow of information into and out of hidden states. However, GRUs use 2 instead of 3 gates: a reset and an update gate. Moreover, they do not use a secondary cell state that needs to be preserved across iterations. Instead, they calculate a potential update every iteration and use the update gate to mix the current hidden state with the potential update.

GRU networks are also best understood by the set of equations that describe them,

Compute the update and reset gates,

$$\mathbf{u}_t = \sigma(\mathbf{W}_{xu}\mathbf{x}_t + \mathbf{W}_{hu}\mathbf{h}_{t-1} + \mathbf{b}_u), \tag{3.12}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1} + \mathbf{b}_r). \tag{3.13}$$

Calculate the potential update,

$$\hat{\mathbf{h}}_{\mathbf{t}} = \tanh(\mathbf{W}_{x\hat{h}}\mathbf{x}_t) + \mathbf{W}_{h\hat{h}}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_{\hat{h}}). \tag{3.14}$$

Mix the potential update with the actual hidden state,

$$\mathbf{h}_t = (1 - \mathbf{u}_t) \odot \mathbf{h}_{t-1} + \mathbf{u}_t \odot \hat{\mathbf{h}}_{\mathbf{t}}. \tag{3.15}$$

GRUs are faster (less steps) and more memory efficient (do not have a secondary cell state) than LSTMs. At least on some tasks they have been shown to have similar performance to LSTMs [Chung et al., 2014]. Unfortunately, their architecture do not align with recurrent connections in the brain.

In Vaswani et al. [2017], the Transformer model was introduced. Transformer networks were a paradigm shift in the handling of sequential data. In particular, they got rid of RNNs altogether and extensively used multiple attention layers [Bahdanau et al., 2014] to learn dependencies between individual items in a series. Another prominent feature of Transformer networks is the use of position embedding which are added to each item in the sequence. By adding these position embedding, the order of the sequence was built into the sequence itself and allowed the entire sequence to be processed simultaneously. This dramatically improved the run and training time of Transformer networks (on graphics processing unit (GPU) architectures). Transformers outperformed all RNN models and are currently considered SOTA in almost all sequence tasks (but see Legendre Memory Units (LMU) RNNs [Voelker et al., 2019]). Unfortunately, the attention mechanism (which uses a Softmax function over all inputs in the sequence) and the parallel processing of all items in the sequence are different from how the brain handles sequential data and are not considered biologically plausible.

3.4 Biologically plausible ANNs

From the very beginning, the architecture of ANNs was inspired by the structure of the brain [Rosenblatt, 1958, Fukushima, 1988]. Their use of populations of simple and identical non-linear units to develop large complex systems is similar to how the brain uses neurons as building blocks for complex sensory systems. Moreover, their stacking of layers of these units into hierarchical systems is reminiscent of the anatomically and functionally distinct areas of sensory systems, through which information passes sequentially (at least as a first pass). CNNs take further inspiration from the visual cortex and restrict connections between neurons to spatially confined areas, resembling the small cRFs of simple V1 cells [Hubel and Wiesel, 1962]. Moreover, commonly included pooling layers are designed to tolerate small translations in the responses of feature extracting neurons, in a manner similar to how complex cells in V1 develop tolerances to spatial phase changes of simple cells.

There are also some remarkable similarities between the features learned by trained CNNs and tuning preferences of neurons in the ventral stream. This is even more astonishing considering that CNNs are trained with objective maximizing cost functions without any neurophysiological constraints [Yamins et al., 2014, Yamins and DiCarlo, 2016]. The first feature extracting layer of CNNs learns multiple Gabor-like edge detectors and colour blobs that are analogous to tuning profiles of V1 neurons [Krizhevsky et al., 2012, Zeiler and Fergus, 2014]. The complexity of features learnt by CNNs increases with depth [Guclu and van Gerven, 2015, Zeiler and Fergus, 2014], rivaling the increase in neuron selectivities as one traverses down the ventral stream [Kruger et al., 2012].

Many studies have shown that there are also similarities between internal representations in CNNs and the ventral visual stream. Khaligh-Razavi and Kriegeskorte [2014] used representation similarity analysis [Kriegeskorte et al., 2008] to show that objects CNNs represent similarly are likewise similarly represented by the primate IT cortex. The similarity was greatest for the last dense layer of AlexNet [Krizhevsky et al., 2012]. Yamins et al. [2014], Cadieu et al. [2014] showed that measured neuronal responses in various parts of ventral stream can be expressed as simple linear combination of activations of deeper layers of CNNs. In the model they considered [Zeiler and Fergus, 2014], the last network layer was the best at explaining measured IT responses while the penultimate layer best explained responses in V4 cortex. In Khan and Tripp [2017], Tripp [2017], statistical properties of IT neurons and their populations were compared with those of nodes in CNNs. Many similarities were found including population level spareness and size tuning bandwidth. Given their best-in-class performance and superior ability to explain neural activities in multiple cortices, deep CNNs were considered best in class mechanistic models of the ventral visual stream [Yamins and DiCarlo, 2016].

More recently, driven by the goal of maximizing task level performance, the architectures of state-of-the-art (SOTA) models have become significantly more complex; they can include hundreds of layers and multiple branching structures. Not only has it become difficult to map these models back to the brain, but Schrimpf et al. [2018] has shown that they are also worse at explaining internal representations of the ventral visual stream. Moreover, many other differences between CNNs and the brain's visual system have been observed. Rajalingham et al. [2018] showed that even though deep CNNs categorized object categories similarly, their within category responses to specific instance of objects were not predictive of primate responses, suggesting that the underlying mechanisms by which ANNs reach their decisions may be different. Geirhos et al. [2018] observed that CNNs trained to classify objects on ImageNet [Deng et al., 2009], a large dataset containing millions of natural images of 1000 different objects, showed a clear bias for object texture over object contours and shape. Humans, on the other hand, rely mainly on object contours when making decisions. Deep networks also have well-known functional limitations, such as being easily fooled by adversarial stimuli [Szegedy et al., 2013, Nguyen et al., 2015], poor generalization outside the distribution of training data [Hendrycks and Dietterich, 2019, Geirhos et al., 2018, Serre, 2019], and poor sample efficiency [Lake et al., 2015].

These differences and commonly found recurrent connections in the ventral stream has lead to many researchers interested in developing biologically plausible ANNs to focus on incorporating recurrence into CNNs. There is even recent evidence that recurrent activity is involved in fast 'core' object recognition [Kar et al., 2019]. Several different approaches have been tried. Below, I highlight some of the more heavily cited ones.

3.4.1 Recurrent convolutional neural networks

Liang and Hu [2015] attempted to capture lateral within-layer interactions by including an extra convolutional operation focused on the same layer's activations but at a previous time in a new recurrent convolutional layer,

$$x_{ijk}(t) = \sigma\left(\mathbf{w}_k^f \mathbf{u}^{(i,j)}(t) + \mathbf{w}_k^r \mathbf{x}^{(i,j)}(t-1) + b_k\right), \tag{3.16}$$

where, $x_{ijk}(t)$ is the activation of the node at position $(i,\ j)$ for kernel k at time t, $\mathbf{w}_k^f$ is the kth feedforward kernel, $\mathbf{w}_k^r$ is the kernel that models lateral interactions, $\mathbf{u}^{(i,j)}(t)$ is the feedforward input to the layer at time t, $\mathbf{x}^{(i,j)}(t-1)$ is the recurrent input (the activation of the layer) at time $t-1$ and $\sigma()$ is the activation function. In the full model, where activations of all nodes are considered simultaneously, multiplications are replaced with convolutions. Recurrency in this recurrent convolutional neural network (rCNNs) is different from general RNNs as inputs are static and only lateral interactions change with time.

Liang and Hu [2015] constructed a 5 convolutional layer CNN model with the top 4 layers replaced with recurrent convolutional layers. An identical number of kernels was used in the feedforward and lateral directions. The model was compared with 2 control feedforward networks with a similar number of parameters and convolution operations respectively. Classification error was compared over the MNIST, CIFAR 10/100 [Krizhevsky et al., 2009] and the Google Street View Numbers [Netzer et al., 2011] datasets. The network outperformed these control models. Since then, newer models have surpassed this model's performance.

Spoerer et al. [2017] extended the recurrent convolutional layer of Liang and Hu [2015] to additionally include feedback from a higher layer,

$$h_{\tau,m,i,j,k} = \sigma\left(\mathbf{w}_{m,k}^b \boldsymbol{h}_{\tau,m-1,i,j} + \boldsymbol{w}_{m,k}^l \boldsymbol{h}_{\tau-1,m,i,j} + \boldsymbol{w}_{m,k}^t \boldsymbol{h}_{\tau-1,m+1,i,j} + b_{m,k}\right), \tag{3.17}$$

where $\boldsymbol{h}_{\tau,m,i,j,k}$ is the activation of the node at position (i,j) in response to kernel k, in layer m at time τ, $\boldsymbol{w}_{m,k}^b$, $\boldsymbol{w}_{m,k}^l$, $\boldsymbol{w}_{m,k}^t$ are the kth kernel in the bottom up, lateral and top down directions of layer m, $b_{m,k}$ is a bias term and $\sigma()$ is the activation function. By omitting some of the connections, three different recurrent convolutional layers were defined: bottom up and lateral connections (BL), bottom up and top down connection (BT), and all three connections (BLT). For each type, a 2 hidden layers CNN was constructed. These models were compared with two feedforward control models for which the kernel size or the number of feature kernels where increased to match the parameters of the recurrent models. Classification performance of all five networks were compared over MNIST with various distortions. Two types of occlusion distortions were added to images: digital debris where fragments of other classes were added on top of the target class and digital clutter where multiple complete classes were overlapped. Additive white Gaussian noise was also added to test images.

Spoerer et al. [2017] found that under all considered distortions, recurrent neural networks outperformed feedforward models. Moreover, under the digital debris condition recurrent connections offered greater resilience to higher levels of occlusion; percentage decrease of classification error dropped much less for recurrent networks with increasing levels of occlusion. This was not the case for digital clutter where full classes overlapped. Recurrent convolutional still performed better than feedforward connections, but fell similarly with occlusion increased. Amongst the recurrent networks, BLT and BL had similar performance except under the highest levels of occlusions. When white Gaussian noise was added to input images, recurrency did not offer much advantage. Classification performance was similar to the case when no noise was added. However, this may be due to the amount of noise added, as a similar result was seen for feedforward networks.

Spoerer et al. [2020] expanded the BL model variant to a 7 layer network and showed that recurrent CNNs can be used on much larger datasets such as ImageNet. They compared their model with several feedforward control models whose parameter were matched to the recurrent model in several ways including: increasing the kernel sizes, adding more feature maps and adding more layers. The recurrent BL model was able to outperform all feedforward controls of similar capacities. Moreover, they observed that with every iteration the classification accuracy of model predictions increased. By using a threshold on the probabilistic predictions of the model, they could flexibly trade off between time and accuracy in the model. Finally, they found a high correlation between human reactions time and the number of iterations the model spent on individual images.

3.4.2 CORnet-S

In Kubilius et al. [2019] the CORnet-S model was introduced. Its architecture was inspired by the high level neuroanatomy of the brain's visual system and consisted of four operational blocks that were modeled after the V1, V2, V4 and IT cortices. Each block contains 2 to 3 convolutional layers and other non-feature extracting layers, such as max pooling and batch normalization. Recurrent connections were added at the block level rather than at individual convolutional layers; block outputs are fed back into their inputs at each time step. Moreover, each block iterates though a predefined number of time steps before passing its outputs to the next block. Inputs from lower layers feed into a block only in the first time step. In all subsequent time steps, inputs are replaced by recurrent outputs by means of a gate. The output of the last IT layer feeds into a dense layer that maps its activity into category predictions.

The model was trained on the task of object classification on ImageNet. Trained models were tested for accuracy and for the ability to explain a large benchmark of V4 and IT neural recordings as well as behavioural measurements (Brain-Score [Schrimpf et al., 2018]). The CORnet-S model achieved the highest ranking on Brain-Score benchmark and outperformed many feedforward models of similar capacities on classification accuracy. Based on their results, Kubilius et al. [2019] claimed CORnet-S as the current best mechanistic model of the ventral visual stream.

3.4.3 Horizontal Gated Recurrent Unit and Gamma-Net

The Horizontal Gated Recurrent Unit (hGRU) Model of Linsley et al. [2018] is a single layer convolutional RNN model that was inspired by lateral connections in the V1 cortex. Unlike standard RNNs, recurrent interactions were designed based on the interactions between units in an existing computational model of the neural phenomenon of surround modulation [Mély et al., 2018]. Linsley et al. showed that their model is proficient at detecting long range spatial patterns on a synthetic visual task that involved tracing fragmented contours to determine if two circles were connected. The model matched or outperformed several feedforward models including those that had several orders of magnitudes more parameters. In addition, learnt lateral connections between component units contained several examples of concentric circles with opposing polarities. A connection structure that is consistent with hypothesized excitatory near-cRF and inhibitory far-cRF of surround modulation [Angelucci et al., 2017, Shushruth et al., 2013]. Finally, when the model was trained on another synthetic visual task that involved detecting fragmented linear contours, its performance monotonically dropped as the spacing between fragments increased, consistent with behavioural trends observed by [Li and Gilbert, 2002].

In building their model, although they started by modeling interactions between units as proposed by Mély et al. [2018], several modifications were made to improve model trainability and expressiveness. In particular, a Gated Recurrent Unit (GRU) [Cho et al., 2014] inspired architecture that included multiple learnable gates was incorporated. Compared to vanilla RNNs, gated RNNs are better at learning long range dependencies [Tallec and Ollivier, 2018]. The set of equations below describe how lateral interactions between units were modeled,

Compute suppression gates,

$$\boldsymbol{G}_1[t] = \sigma\left(\boldsymbol{U_1} \circledast \boldsymbol{H}_2[t-1] + \boldsymbol{b}_1\right), \tag{3.18}$$

where $\boldsymbol{U}_1 \in \mathbb{R}^{1\times1\times K\times K}$ are learnable kernels that act on the output activity of all component units, $\boldsymbol{H}_2[t-1]$, at time step $t-1$, K is the number of input as well as output channels, $\circledast$ is the convolution operator, $\boldsymbol{b_1}$ are biases and σ is a sigmoid non-linear activation function.[2]

Compute suppressive influences,

$$\boldsymbol{C_1}[t] = \boldsymbol{W} \circledast \left(\boldsymbol{G}_1[t] \odot \boldsymbol{H}_2[t-1]\right), \tag{3.19}$$

where $\boldsymbol{W} \in \mathbb{R}^{S\times S\times K\times K}$ are learnt lateral weights, $S\times S$ is the spatial extent of lateral kernels and $\odot$ is the Hadamard product. A default size of 15×15 was used as the spatial extent of lateral kernels.

Calculate the recurrent input,

$$\boldsymbol{H}_1[t] = \zeta\left(\boldsymbol{X} - \boldsymbol{C}_1[t](\boldsymbol{\alpha}\boldsymbol{H}_2[t-1] + \boldsymbol{\mu})\right), \tag{3.20}$$

where $\boldsymbol{X}$ is the feedforward input, $\boldsymbol{\alpha}$ and $\boldsymbol{\mu}$ control the multiplicative and additive influences of suppression respectively and ζ is the hyperbolic tangent non-linear activation function.

Compute the mixing gate,

$$\boldsymbol{G}_2[t] = \sigma\left(\boldsymbol{U}_2 \circledast \boldsymbol{H}_1[t] + \boldsymbol{b_2}\right), \tag{3.21}$$

where $\boldsymbol{U}_2 \in \mathbb{R}^{1\times1\times K\times K}$ are learnable kernels that act on the recurrent input, $\boldsymbol{H}_1[t]$ at time t and $\boldsymbol{b_1}$ are biases.

Compute enhancing influences,

$$\boldsymbol{C_2}[t] = \boldsymbol{W} \circledast \boldsymbol{H}_1[t]. \tag{3.22}$$

[2]The notation used has been slightly modified from that of Linsley et al. [2018] to improve clarity. In particular, the collective interactions of the entire layer are used instead of a particular unit in the layer. Furthermore, the superscripts identifying recurrent inputs (1) and outputs (2) have been changed to subscripts.

Calculate the potential output update,

$$\hat{\boldsymbol{H}}_2[t] = \zeta\left(\boldsymbol{k}\boldsymbol{H}_1[t] + \boldsymbol{\beta}\boldsymbol{C}_2[t] + \boldsymbol{w}\boldsymbol{H}_1[t]\boldsymbol{C}_2[t]\right), \tag{3.23}$$

where $\boldsymbol{b}$ and $\boldsymbol{w}$ controls the additive and multiplicative influences of excitatory influences and $\boldsymbol{k}$ controls the influence of recurrent input directly (without excitatory influences).[3]

Mix the potential output update with the previous output to get the recurrent output at time step t,

$$\boldsymbol{H}_2[t] = \eta\left(\boldsymbol{H}_2[t-1](1-\boldsymbol{G}_2[t]) + \hat{\boldsymbol{H}}_2[t] * \boldsymbol{G}_2[t]\right) \tag{3.24}$$

The above recurrent layer received inputs from a preceding convolutional layer that consisted of 25 kernels of size 7 × 7. These kernels were initialized as Gabor filters with 12 different orientations plus a radially symmetric difference of Gaussian kernel. Outputs of the convolutional layer were passed though a squaring non-linear activation function before being fed into the hGRU layer. Outputs of the hGRU layer were passed to a read out stage that consists of two 1 × 1 convolutional filters, batch normalization and a global max pooling layer that mapped onto desired output dimensions.

In Linsley et al. [2020b], they extended their model into a multi-layer hierarchical model, γ-net, that included similarly designed feedback connections(fGRU) and multiple hGRU layers. When trained on the task of object contour detection in natural images, γ-net models matched the performance of SOTA feedforward models. However, when trained with smaller subsets of the data, γ-net models outperformed their feedforward counterparts. Trained models also replicated a well known psychophysical phenomenon, the orientation tilt illusion [O'Toole and Wenderoth, 1977]. In the orientation tilt illusion, the perceived orientation of a central circular grating tilts away from the orientation of a surrounding concentric grating when both have similar orientations and tilts towards the orientation of the surrounding grating when both have relatively orthogonal orientations.

3.5 Precautions when comparing biological systems and DNNs

While DNNs can be used to shed more light and explore the inner workings of the brain [Cichy and Kaiser, 2019, Barrett et al., 2019], there are some important factors that need to

[3]In later revisions of the model [Linsley et al., 2020b], the $\boldsymbol{k}\boldsymbol{H}_1[t]$ term was removed and non-negativity was more strictly enforced by replacing, ζ with a ReLU nonlinear activation function. In addition, ReLU activation functions were added to the enhancing and suppressive influences as well.

be taken into account before drawing conclusions. DNNs are infamous black-boxes whose internals are hard to interpret. It is tempting to attribute the human-like performance of DNNs to the same underlying principles [Buckner, 2019]. However, just because two systems have the same performance does not mean they utilize the same internal mechanisms. It is possible to solve tasks in multiple ways, especially complicated high dimensional ones.

DNNs are also prone to shortcut learning [Geirhos et al., 2020]; they will use any discriminative feature, available in training data, to solve the task. More often than not, they will use the simplest ones and will overly rely on them at the expense of others. On complicated tasks these features can be very different and unexpected from those used by humans [Geirhos et al., 2018, Brendel and Bethge, 2019, Funke et al., 2021]. The problem with using these features is that although they work well on training data and on test data generated from the same underlying probability distribution, they typically fail to generalize to data that is out-of-distribution from the training set.

For complicated tasks, it is often difficult to detect shortcut learning. Recently, a number of studies have suggested guidelines on how to design and test models so that they learn *intended* solutions as opposed to a shortcut. In particular, [Funke et al., 2021] pointed out that it is important to carry out several different analyses to equate strategies and at multiple levels Lindsay [2020]. Sinz et al. [2019] suggested the possibility of adding constraints to models to narrow the solutions space of DNNs. These inductive biases can guide networks to prefer certain solutions over others. Suggestions included adding more biologically realism to constrain model architectures, matching neurophysiological data and simultaneously training on multiple tasks [Tripp, 2019]. Finally, Geirhos et al. [2020] highlighted the importance of testing trained models with out-of-distribution data to ensure models learnt the underlying principles correctly.

ex

Chapter 4

Contour Integration in Artificial Neural Networks

In the previous chapters, I reviewed several existing computational models of contour integration. These models can replicate neural mechanisms in exceptional detail and can reproduce many observed neurophysiological data. Unfortunately, they cannot satisfactorily relate contour integration to high-level goals of the complex systems that they are part of. ANNs on the other hand, are capable of solving high-level tasks. However, solutions learnt by DNNs are difficult to interpret. Often these solutions are different from those anticipated by humans. It is important to constrain these models so that they favor desired solutions over shortcut solutions [Geirhos et al., 2020]. In this chapter, I propose my approach to constraining DNNs by incorporating architectures, nodes, and the interactions proposed by these computational models into DNNs. Incorporating realistic neural mechanisms is a good way to relate mechanism to function.

In this chapter, first, the proposed ANN based model of contour integration is described. Connections within the model and incorporated attributes of V1 lateral connections are discussed. Second, the model is trained on a dataset of synthetic fragmented contours. The dataset contains stimuli typically used to study properties of contour integration in the brain. Third, to see if the model learnt to integrate contours in a manner similar to the brain, trained models are analyzed for consistency with several known behavioural and neurophysiological properties of contour integration. Fourth, learnt lateral connection structures of trained models are compared with observed properties of V1 lateral connections. Fifth, a sensitivity analysis of the model's parameters is carried out to investigate how individual parameters influence its performance. Sixth, different variants of the model are explored and compared to investigate the role of its individual components. Finally, the model is compared with existing models of contour integration.

4.1 Contour integration block

The central component of the model is the contour integration (CI) block [Khan et al., 2020]. It models V1 orientation columns[1] and the interactions between them. Each orientation column represents a population of neurons that respond to edges of similar orientations at a particular spatial location. Individual orientation columns are modeled with a pair of excitatory (E) and inhibitory (I) nodes. An E-I pair and all of its connections are shown in Figure 4.1. E and I nodes reciprocally connect within a node pair and selectively with other nodes in their vicinity via lateral connections. E nodes receive inputs from neighboring E nodes and from the I node in their node pair. E nodes send their outputs to neighboring E and I nodes. I nodes receive inputs from neighboring E nodes but only send their outputs to the E node in their node pair. They do not connect with other I nodes.

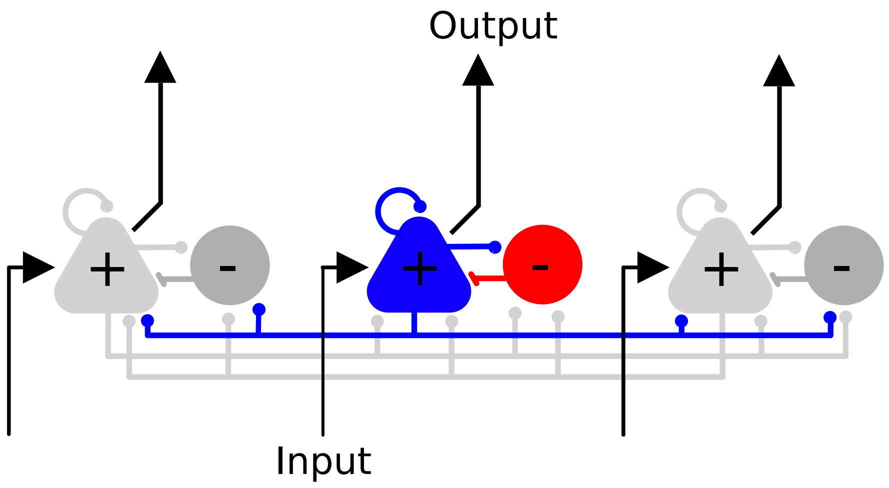

Figure 4.1: **Internal connections of the contour integration block**. Best viewed in colour. Each excitatory (+) and inhibitory (-) pair represents a V1 orientation column. Outbound connections of a single excitatory node are highlighted in blue, while those of its paired inhibitory node are shown in red. Connections ending in a circle are excitatory while those ending in a bar are inhibitory.

[1]More precisely, it models neurons in the superficial layers of V1 orientation columns.

In the brain, lateral connections of V1 orientation columns are sparse and preferentially connect with neighbors with similar orientations [Stettler et al., 2002]. Existing contour integration models (see Section 2.4) use fixed association-field shaped [Field et al., 1993] connection structures to model directed lateral connections of orientation columns. In this implementation, I do not force fixed lateral connection patterns, but learn them through optimizing for task performance; all E-I pairs are connected over a defined spatial area and by minimizing loss on high-level tasks, the model learns which connections to keep and which ones to get rid of. Additionally, a sparsity constraint is used to retain only the most prominent connections (see Section 4.3). The constraint penalizes weights that are far away from the origin and encourages them to be small. Lastly, batch normalization [Ioffe and Szegedy, 2015] is included to model the spatially spread but weak omni-directional inhibition of contextual interactions [Kapadia et al., 2000].

The interactions between nodes are derived from the differential equation based contour integration model of Piëch et al. [2013]. To incorporate it into neural networks, I used Euler's method to express its components as difference equations. The resultant equations can then be trained using standard RNN training techniques [Linsley et al., 2018, Tallec and Ollivier, 2018]. The final form of these interactions is represented by,

$$x_t = (1-\sigma(a))x_{t-1} + \sigma(a)\left[-\sigma(J_{xy})f_y(y_{t-1}) + I_{0e} + I + W_e \circledast f_x(X_{t-1})\right], \tag{4.1}$$

$$y_t = (1-\sigma(b))y_{t-1} + \sigma(b)\left[\sigma(J_{yx})f_x(x_t) + I_{0i} + W_i \circledast f_x(X_t)\right]. \tag{4.2}$$

Here, x and y are the membrane potentials of E and I nodes, respectively, $f_{.}(.)$ is a non-linear activation function that transforms membrane potentials into firing rates, $\sigma(a)$, $\sigma(b)$ are functions of the membrane time constants of component neurons, $\sigma(J_{xy})$, $\sigma(J_{yx})$ are local I $\rightarrow$ E, E $\rightarrow$ I connection strengths, $\sigma()$ is the Logistic Sigmoid function which constrains time constants and local connection strengths to be positive, W_e are lateral excitatory connections from E nodes in nearby columns to E, W_i are connections from nearby E nodes to I, $f_x(X_t)$ is the output of all modeled nodes at time t, $\circledast$ is the convolution operator, I is the external input and $I_{0.}$ is a node's background activity. E nodes also locally self connect, E $\rightarrow$ E. This is included in W_e which connects neighboring columns and includes those that operate over the same spatial location.

The model of [Piëch et al., 2013] used fixed positive only lateral connections between E-I pairs. In my model, these connections are learnt. However, regular ANNs do not put any sign constraints on their learnt parameter. To ensure internal nodes interacted in a manner consistent with the model of [Piëch et al., 2013], lateral weights are constrained to be positive only; during the learning process, negative weights are clipped. Details on how the above equations were derived from the differential equation model of [Piëch et al., 2013] are given in Appendix A.

The full CI block consists of a 3D grid of E-I pairs; separate E-I pairs are defined at each spatial location and at each input channel. Here, channels refers to the different features extracted by the preceding layer. Orientation columns connect with neighboring columns at different spatial locations and on different channels (including those at the same location) via lateral connections. Parameters of E-I node pairs are shared across spatial locations but not across channels. The CI block acts on the output of an edge extraction layer (a standard first layer of a convolutional network). The external input to each E-I node pair, I, is the output of a single node at the corresponding spatial and channel location in the preceding layer. Feedforward input is received only by E nodes. After iterating through the CI block for N_{itr} steps, E node outputs are passed to the next layer. The spatial extent of V1 lateral connections S is much larger than cRF of V1 neurons neurons [Stettler et al., 2002]. Consistent with this, S was defined to be much larger than edge extracting kernels. Parameters of the CI block as well as their default settings are shown in Table 4.1.

Feedforward control block

The CI block was compared with a control feedforward block with matching capacity (number of parameters). The control block contained a matching number of convolutional layers but ordered them sequentially. Each convolutional layer had the same configurations and parameters as those used in the CI block. Additionally, batch normalization and dropout layers ($p_{dropout} = 0.3$) were added after every convolutional layer. This was necessary to prevent the control from over-fitting training data. The same sparsity constraint as used on lateral connection kernels in the CI block was used on the convolutional kernels of the control block. However, unlike the CI block, no positive-only weight constraint was used for the control block and it was free to choose whichever weights led to the highest performance. For ease of notation, the term CI block refers to this control block as well as the actual CI block and the network type is used to differentiate between the two (model vs. control).

4.2 The complete model

The CI block, of both the model and the control, sits atop an edge extraction block. The same edge extraction block was used. For edge extraction, the first convolutional layer of a

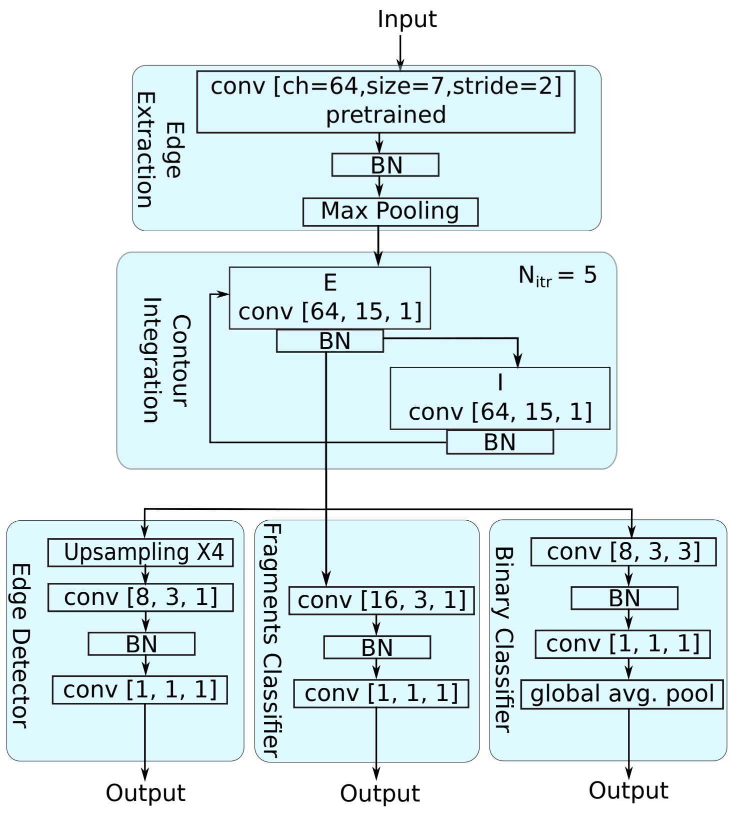

Figure 4.2: **Model architectures**. The main component of the model is the contour integration (CI) block. It consists of a 3D grid of orientation columns and models the horizontal interactions between them. Each orientation column is modeled by a pair of excitatory (E) and inhibitory (I) nodes. Each orientation column receives as input the output of an edge extraction unit at the same spatial location and channel. Horizontal connections connect orientation columns with other orientation columns at different spatial locations and channels. These connections are learnt by optimizing performance on high-level tasks. The full model consists of three main blocks: edge extraction, CI and classification blocks. Edge extraction and CI blocks are common for all tasks. For edge extraction, the first convolutional layer of a ResNet50 [He et al., 2016] that was previously trained on ImageNet [Deng et al., 2009] was used. Task specific classification blocks: edge detector (Section 5.1), fragments classifier (Section 4.3), binary classifier (Section 5.2) map contour integration activations to output labels. For each convolutional (conv) layer, the square brackets specify the number, size, and stride length of kernels. Batch normalization (BN) layers were typically used after convolutional blocks. Bi-linear interpolation was used for up-sampling in the edge detector classification block.

Name	Description	Setting	Trainable
N_{iters}	Number of recurrent iterations	5	No
S	Spatial Extent of lateral connections	15	No
$\sigma(J_{xy})$	I $\rightarrow$ E connection strength	0.1	Yes
$\sigma(J_{yx})$	E $\rightarrow$ I connection strength	0.1	Yes
$\sigma(a)$	E node membrane time constant	0.5	Yes
$\sigma(b)$	I node membrane time constant	0.5	Yes
W_e	Excitatory lateral connections	$[ch_{in}, S, S]$	Yes
W_i	Inhibitory lateral connections	$[ch_{in}, S, S]$	Yes

Table 4.1: **Parameters of the contour integration block.** Values for non-trainable parameters were fixed at the start of training. Trainable parameters were initialized by the indicated value and allowed to change during training. S is the spatial extent of lateral kernels and ch_{in} is the number of input channels. Parameters shown are for a single unique orientation column. The full contour integration block modeled ch_{in} unique orientation columns.

ResNet50 model [He et al., 2016] that was pre-trained on ImageNet [Deng et al., 2009] was used. Typically, a batch normalization (BN) and max pooling layer (stride length of 2) were included after the edge extraction block and before the CI block. Not only did this reduce computational complexity (by reducing the spatial dimensions over which the recurrent CI block acts) but improved performance as well. The output of the CI block is passed to classification blocks that map activations to desired outputs. Separate classification blocks were used for each considered task. In general, capacities of classification blocks were kept to a minimum to allow the CI block to do most of the work. Figure 4.2 shows the architectures of all the models that were used in this work.

4.3 Synthetic contour fragments task

As a first test of the network, I used stimuli typically used to study biological contour integration [Field et al., 1993, Li et al., 2006, 2008]. Li et al. [2008] found that macaque monkeys progressively improved at detecting contours and had higher contour enhanced V1 responses with experience on these stimuli. Hence, contour integration is learnable from these stimuli. Each input stimulus consisted of a grid of Gabor fragments that were identical in every aspect except for their orientations. The orientations of a few adjacent (contour) fragments were aligned to form a smooth contour. The orientations

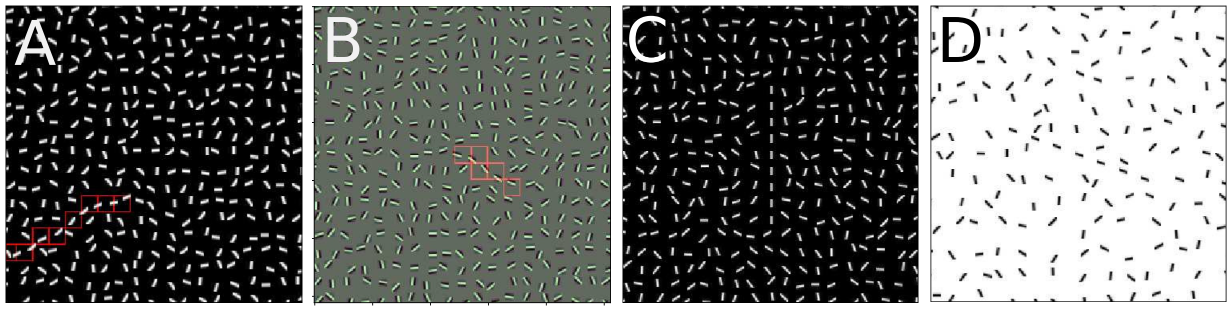

Figure 4.3: **Contour fragments stimuli**. (Best viewed in colour). A and B, Training stimuli. All line segments are Gabor fragments, which are identical except for their orientations. The orientations of a few adjacent fragments were aligned to form a smooth contour (highlighted in red). Remaining fragments had orientations that were uniformly distributed. Contours differed in their location, length, inter-fragment curvature and their component Gabors. C and D, Test stimuli use to analyze the impact of length and inter-fragment spacing. Test stimuli consisted of centrally located contours with different lengths (C) and different spacing between contour fragments (D).

of the remaining (background) fragments varied randomly. Embedded contours differed in their location, length l_c (number of fragments), inter-fragment degree of curvature β, and the type of Gabor fragment used in their construction. Example stimuli are shown in Figure 4.3.

Stimulus construction

Stimuli similar to those of Field et al. [1993] were used. To construct an input stimulus, first, a Gabor fragment, contour length in number of fragments, l_c, and curvature, β, were selected. Each Gabor fragment was a square tile the same size as the cRF (kernel spatial size) of the edge extracting layer. Second, a blank image was initialized with the mean pixel value of the boundary pixels of the selected Gabor. This was done to blend the edges of Gabor fragments with the background. Third, the input image was sectioned into a grid of squares (full tiles) whose length was set to the pixel length of a fragment plus the desired inter-fragment spacing, d_{full}. For training stimuli, inter-fragment spacing was set to the same length as fragment length. The grid was aligned to coincide the center of the image with the center of the middle full tile. Fourth, a starting contour fragment was randomly placed in the image. Fifth, the location of the next contour fragment was determined by projecting a vector of length $d_{full} \pm {}^{d_{full}}/_{8}$ and orientation equal to the previous fragment's orientation $\pm\beta$. The random direction of β and distance jitter were added to prevent them from appearing as cues to the network. Sixth, a fragment rotated by β, was added at this position. The fifth and sixth steps were repeated until $\lfloor l_c/2 \rfloor$ contour fragments were added to both ends of the starting fragment. Seventh, background fragments were added to all unoccupied full tiles. Background fragments were randomly rotated and positioned inside the larger full tiles. Lastly, a binary label of whether the center of a contour fragment was present in each full tile was generated.

All input images were of a fixed size of 256×256 pixels. The first convolutional layer of a ResNet50 model uses kernels of size 7×7. Consequently, Gabor fragments of size 7×7 pixels and full tile of size 14×14 pixels were used in stimulus construction. This resulted in labels of size 19×19 for each input stimulus.

Dataset

The contour fragments dataset contains 64,000 training and 6,400 validation images. In its construction, 64 different Gabors types, contours of lengths, l_c, 1, 3, 5, 7, 9 fragments and inter-fragment rotations, β, of $0^\circ \pm 15^\circ$ were used. Gabor parameters were manually picked with the only restriction that the resultant Gabor fragment visually appear as a well-defined line segment. Each Gabor fragment was defined over three channels and the dataset included coloured as well as standard black and white stimuli. $l_c = 1$ stimuli were included to teach the model to not do contour integration when there are no co-aligned fragments outside the cRF. Contour integration requires inputs from outside the cRF and the model had to learn when not to apply enhancement gains. For these stimuli, the label was set to all zeros. An equal number of images were generated for each condition. Due to the random distance jitter, the random inter-fragment rotation between fragments, and the random location of contours, multiple unique contours were possible for each condition. Moreover, background fragments varied in each image. Code to generate the dataset is freely available at `https://github.com/salkhan23/contour_integration_pytorch`.

Task

Networks were tasked with predicting which full tiles contained contour fragments (red squares in Figure 4.3A and B) and which did not. A fragment classifier block (see Figure 4.2) was used to map CI block outputs to the desired label size. The fragments classifier block's capacity was intentionally kept low to allow the CI block to do most of the work. The same classifier block was used by the model and control.

Network performance was evaluated using the mean Intersection over Union (IoU) score of predictions and labels,

$$IoU(A, B) = \sum_{n=1}^{N} \frac{1}{N} \frac{(A_n \cap B_n)}{(A_n \cup B_n)}, \tag{4.3}$$

where, N is the number of images in the dataset, and A and B are network predictions and labels for image n. To get network predictions for an image, its output was passed through a Logistic Sigmoid non-linearity. Next, sigmoid outputs were converted to binary predictions by thresholding. A threshold value of 0.5 was used. Given the binary prediction of a network for each full tile location, the intersection with the label was found by multiplying the predictions with the label while the union was found by summing labels and predictions followed by subtracting the intersection of the two. An IoU score of 1 signifies a perfect match between predictions and labels, while an IoU score of 0 means that there is no match between what the network predicted and the label. The mean IoU score was found by averaging IoU scores over all images in the dataset.

Training

Models were trained to minimize binary cross entropy loss,

$$H_p(q) = -\frac{1}{N} \sum_{i=n}^{N} \left[p \log(q) + (1-p) \log(1-q) \right], \tag{4.4}$$

where p is the label and q is the soft (not binary but in range $[0, 1]$) prediction of the model. Here, N represents the total across all images as well as the total predictions per image.

To encourage sparse lateral connections, L1 regularization loss multiplied with an inverted 2D Gaussian mask was applied over excitatory and inhibitory lateral kernels,

$$L_{sparsity} = |(1 - G(\sigma_M))W_e| + |(1 - G(\sigma_M))W_i|, \tag{4.5}$$

where $G(.)$ is a normalized 2D Gaussian mask whose spatial spread, σ_M, is defined by its standard deviation (SD). Importantly, the use of the Gaussian mask allowed for a gradual tapering of connection strengths rather than a hard cutoff of lateral connection size.

The total loss was defined as,

$$L_{total} = H_p(q) + \lambda L_{sparsity}, \tag{4.6}$$

where λ is a weighting term for sparsity loss. For the contour fragments dataset, λ was set to 10^{-4} and σ_M was set to 10 pixels. Learnt lateral connections of the model (but not the control) were restricted to be positive-only. After every weight update step, negative weights were clipped to 0. Networks were trained for 100 epochs with the Adam [Kingma and Ba, 2014] optimizer. The starting learning rate was set to 10^{-4} and was reduced by a factor of 2 after 80 training epochs. A fixed batch size of 32 was used during training.

All input images were of a fixed size of 256 × 256 pixels. Input images were normalized with the dataset mean and standard deviation to be zero centered with a unit SD on average.

4.4 Results

In this section, performances of the model and the control on the contour fragments dataset are analyzed and compared. First, networks are compared on task level performance. IoU scores while training are compared to analyze training efficiency. Peak IoU scores over all training are also compared. Second, trained networks were tested on stimuli similar to those used to analyze behavioural and neurophysiological properties of contour integration in the brain. Networks were tested for consistency with observed trends. Third, learnt lateral connections of the model were compared with known properties of lateral connections in the V1 cortex.

4.4.1 Behavioural (task level) performance

	Train	Validation	Straight Contours Only
Model	$87.33 \pm 0.28\%$	$84.48 \pm 0.30\%$	$85.73 \pm 3.15\%$
Control	$71.62 \pm 0.35\%$	$73.61 \pm 0.38\%$	$82.16 \pm 5.53\%$

Table 4.2: **Peak IoU scores on the contour fragments dataset.** Peak values (mean ± SD) were averaged across 5 independent runs for each network. In the straight contours only column, trained networks were tested on a new dataset composed of centrally located straight contours only. These results were compared with observed data [Li et al., 2006].

The loss and IoU scores over the time course of training for both the model and the control are shown in Figure 4.4. Results were averaged over 5 independent runs with different random seeds. Early in training, IoU scores of the control rose faster than the model. However, they quickly plateaued while the models performance kept improving.

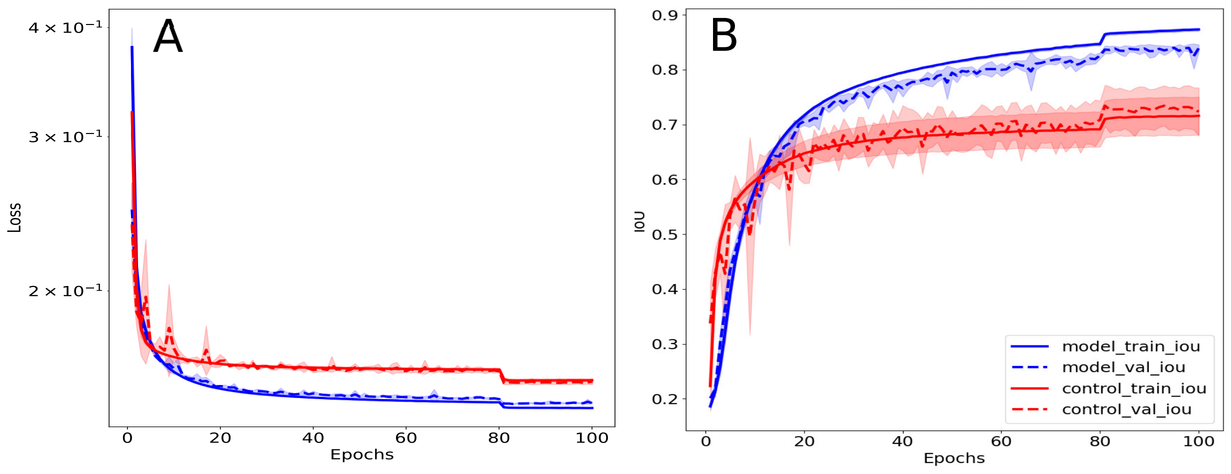

Figure 4.4: **Loss and intersection-over-Union (IoU) scores over the time course of training on the contour fragments dataset.** Best viewed in colour. Model (blue) and control (red) (A) loss and (B) Mean IoU vs training time. Results were averaged over 5 independent runs with different random seeds. Dark lines show mean values averaged over multiple runs and shaded area shows one standard deviation from means.

Average peak IoU scores over training are shown in Table 4.2. The model performed $\approx 11\%$ better than the control (Validation). Table 4.2 also shows mean IoU scores over centrally located straight contours from the analysis of the effects of contour lengths (see Section 4.4.2). Both the model and the control were better at detecting straight contours compared with curved contours, consistent with Field et al. [1993]. For straight contours, the model outperformed the control although the relative increase in performance when comparing all contours in the dataset and straight contours was higher for the control.

4.4.2 Effects of contour length and fragment spacing

Trained networks were tested for consistency with behavioural and neurophysiological data with centrally located straight contours only (consistent with available neurophysiological data [Li et al., 2006]). Behavioural performance was quantified using task-level mean IoU scores. Neurophysiological responses were monitored at the output of centrally located neurons in CI Blocks. For the model, this corresponded to the outputs of E neurons while for the control, the outputs of the second convolutional layer were monitored. In test stimuli, the starting contour fragment was always centered at the image center such that it was fully contained within the cRF of monitored neurons. This ensured centrally located neurons always received a full stimulus within their cRF. When testing for the effect of fragment spacing, the distance between fragments was varied; a condition that was not seen by networks during training. New test stimuli were generated for each analyzed neuron and were not seen by networks during training. Example test stimuli are shown in Figure 4.3C and D.

Neural responses of each of the 64 channels of the CI block were analyzed separately. For each channel, first, the optimal stimulus was found by monitoring which of the 64 Gabor fragments elicited the maximum response in the cRF. Next, test stimuli were constructed by extended contours in the preferred direction (the orientation of the optimal Gabor fragment). Similar to Li et al. [2006], neurophysiological responses were quantified by the contour integration gain,

$$G(l_c, RCD) = \frac{\text{Output } l_c,\ RCD}{\text{Output }\ l_c = 1,\ RCD = 1}, \tag{4.7}$$

where the relative co-linear distance (RCD) is defined as the ratio of inter-fragment spacing to fragment length in pixels and l_c is the number of contour fragments in the contour, i.e, contour length. The condition $l_c = 1$, RCD=1 is when a neuron receives its optimal stimulus within its cRF and no neighboring contour fragments align with it. All training stimuli used a fixed inter-fragment spacing of RCD=1. In test stimuli, variable inter-fragment spacing was modeled by changing d_{full} (see Section 4.3) while keeping the fragment length constant.

The effects of contour length were analyzed using $l_c = 1, 3, 5, 7, 9$ fragments and a fixed spacing of RCD=1 (see Figure 4.3C). The effects of inter-fragment spacing were analyzed using RCD = [7, 8, 9, 10, 11, 12, 13, 14] / 7 and a fixed $l_c = 7$ fragments (see Figure 4.3D). For each condition, results were averaged across 100 different images.

Results were collected over 5 separate runs of each network. While analyzing results, only neurons for which the optimal stimulus (non-zero CI block output for any single Gabor fragment in the cRF) were considered. Out of the 320 possible, 188 model and 178 control neurons met this criteria. Average IoU scores as contour length increases are shown in Figure 4.5A. Both the model and control excelled ($\geq$ 95%) at detecting the absence of contours. There were dips in performance for length 3 contours as they were the hardest to detect. For all other lengths, prediction accuracy increased with length. For straight contours, behavioural performance of both networks were quite similar.

Larger contrasts between the model and control were observed in neurophysiological gains. Figure 4.5B shows population average gains as contour lengths changed. For population average gains, neurons that were unresponsive to any contour condition (all zero

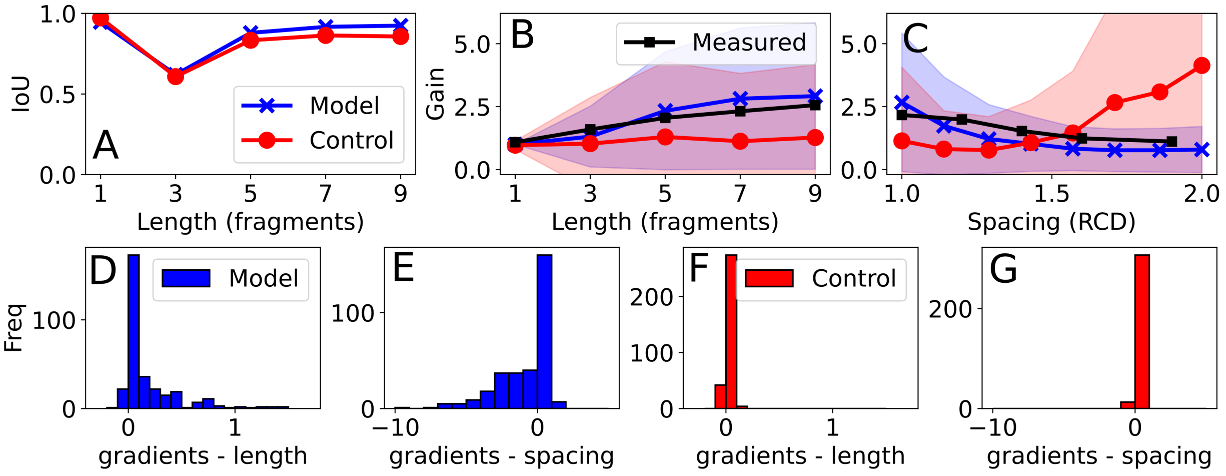

Figure 4.5: **Synthetic contour fragments results**. A, IoU vs. contour length for straight contours. Behavioural classification performance of the model and control were similar. B and C, Population average gains vs. length and vs. fragment spacing, respectively. Contour lengths are expressed in number of fragments and spacing between fragments are expressed in relative co-linear distance (RCD). RCD is defined as the ratio between the spacing between fragments to the length of a fragment in pixels, consistent with available neurophysiological data. Measured neurophysiological gains were extracted from the results of Li et al. [2006]. Dark lines show mean values and shaded areas represent unit standard deviation from means. The results were averaged across 5 different runs for each model. D and E, Gradients of linear fits of the outputs of individual neurons as contour length and as inter-fragment spacing were increased. F and G, similar plots as D and E but for the control. The model shows consistent trends with neurophysiological data while the control behaved differently.

gains) and those that had outlier gains ($\geq$ 20) for any contour condition were also removed. Typically, these large gains were seen for neurons that had small responses to $l_c = 1, RCD = 1$ contours and small changes in the CI block outputs significantly affected their gains. This resulted in the removal of an additional 36 and 144 control neurons. For the model, average gains increased monotonically with contour length. Control average gains did not change appreciably with contour length. Across populations (model and control), individual neurons displayed different amounts of contour enhancement. There was a wide range of enhancement gains exhibited by individual neurons as shown in the mean ± 1 SD shaded area in Figure 4.5B. This may be an artifact of the limited dataset size used during training; it may not have contained enough data to stimulate all edge extraction kernels sufficiently to learn their contour enhancing lateral kernels.

Figure 4.5B also shows measured neurophysiological gains from Li et al. [2006] experiments. Here, population average gains from two monkeys used in their study were extracted using WebPlotDigitizer [Rohatgi, 2020] and their weighted averages are plotted. The impact of contour length on observed and model gains was consistent. Figure 4.5C shows population average gains as the spacing between fragments was increased. Model gains decreased monotonically with spacing, consistent with neurophysiological data [Li et al., 2006]. Control gains, surprisingly increased with spacing.

To get a better picture of overall trends, I plotted histograms of the gradients of linear fits to CI block outputs as contour length and as inter-fragment spacing were increased. This was done for all neurons for which the optimal stimulus was found. Since outputs and not gains were considered, additional filtering due to outliers gains were removed. Results of the model are shown in figure 4.5D and E while those of the control are shown in Figures 4.5F and G. Most model neurons were consistent with population average trends and showed positive slopes as contour lengths increased and negative slope when fragment spacing increased. The results of the model are consistent with observed neurophysiological trends while the control behaved differently. Remarkably, their behavioural predictions were comparable. The model and control appear to be employing different strategies to solve the task and only the model aligns with neurophysiology.

4.4.3 Learnt lateral kernels

Sincich and Blasdel [2001] found that superficial layer V1 lateral connections are anisotropically distributed; lateral connections spread out more densely in the direction of the preferred orientations of source V1 neurons. In their experiments, first, axon staining dye was injected into V1 orientation columns and the spread of lateral connections was analyzed. Areas where axons terminated in clusters were identified as patch sites (after the patchy structure of lateral connection, see Section 2.2.4). Second, the directional selectivity of lateral connections was quantified using an averaging vector, R, of all patch locations surrounding an injection site (see details below). Third, the axis of elongation of lateral connections was compared to the orientation preferences of stained V1 orientation columns. In 11 of the 14 injections sites, a highly elliptical distribution of lateral connections was found (mean index of ellipticity = 0.42) as well as a close correspondence between the axis of elongation of lateral connections and the preferred orientation of injected V1 columns (mean difference of 11°).

The magnitude, r, of R of an injection site quantifies the ellipticity of lateral connections, while the angle, θ, points in the direction along which patches are densest (axis of elongation). The degree of ellipticity measures the compression of a sphere to form an ellipsoid. It is defined as the ratio of the semi-major (largest) axis to the semi-minor (smallest) axis of an ellipsoid. To compute (r, θ) of an injection site, first patch vectors starting at the injection site and ending at the center of all patch sites were constructed. Next, the vector sum of all patch vectors was calculated. Before the vector sum was calculated, the orientation of individual patch vectors were doubled. This was done to account for the fact that orientations separated by $180°$ point in the same direction. As a result, patches that were in opposite directions summed constructively while those that were orthogonal summed destructively. After computing the vector sum, the resultant angle was halved to get θ. An injection site with equidistant patches uniformly distributed around it, gave an average vector of $(0, 0)$ and reflected an omni-directional spread of lateral connections. Finally, r was divided by the magnitude sum of all patch vectors to give a normalized index of ellipticity, r_n. A value of r_n of 0 indicated no ellipticity while a value of 1 indicated a straight line.

Following the analysis of Sincich and Blasdel [2001], the degree of ellipticity and axis of elongation of lateral connections of trained models were investigated. First, the orientation preferences of feedforward edge extraction kernels were found by least square fitting individual kernels to 2D Gabor functions [Movellan, 2002]. The main component of the edge extraction block was the first convolutional layer of a ResNet50. It contains 64 kernels and each kernel has 3 input channels and a spatial spread of 7×7 (see Figure 4.6). The fitting algorithm fit each kernel to 8 parameters of a 2D Gabor including: the x and y location of its center, its amplitude, the orientation, wavelength and phase offset of its sinusoid component and the spatial extent and ratio of the spread in the x versus y direction of its Gaussian envelope. Separate fits for each input channel were found and the orientation of the channel with the highest absolute amplitude was selected as the kernel's preferred orientation. Orientation preferences of the pre-trained edge extraction kernels are shown in Figure 4.6. Feedforward kernels for which Gabor fits could not be found or for which fitted Gabors did not visually appear to be directed were excluded in the analysis. Out of the 64 feedforward kernels, orientation preferences of 42 kernels were considered.

Example learnt lateral excitatory and inhibitory kernels of a trained model are shown in Figures 4.7 and 4.8 respectively. Each set contained 64 kernels. Each individual kernel had 64 input channels (output of edge extraction block) with a spatial spread of 15×15. To visualize lateral kernels, their channel dimension was summed to compress their dimensionality. Visually, lateral excitatory connections spread out further and were more directed than inhibitory connections.

Next, the average r_n for each lateral kernel of the CI block was found. Excitatory and inhibitory lateral kernels were analyzed separately. Slightly different from the method of Sincich and Blasdel [2001], individual patch vectors were calculated for every lateral weight. Moreover, as the weights of each connection were available, they were used to weight individual patch vectors. Stronger weights contributed more to the average vector compared to weaker ones. Only those lateral kernels for whom the orientation of feedforward kernels were found were considered in the analysis. For the trained model discussed above, the average r_n for excitatory neurons was found to be 0.27 while for the inhibitory kernels it was substantially lower at 0.10. The distributions of r_n for excitatory and inhibitory kernels are shown in Figure 4.9A and 4.9B respectively. Results were consistent across the 5 independently trained models; population average excitatory r_n= 0.25 ± 0.02 and inhibitory $r_n = 0.10 \pm 0.01$ (mean $\pm$ 1 SD).

The average r_n of excitatory neurons was lower than what was found by Sincich and Blasdel [2001]. However, they excluded nearby connections and only considered connections that were outside a radius of 200 μm of the injection location. Moreover, all patch vectors were equally weighted. The strength of lateral connections drops sharply with distance [Chisum et al., 2003]. In this analysis, connection weights were used to weight component patch vectors. As Sincich and Blasdel [2001] only considered excitatory connections, a similar comparison with inhibitory connections was not possible.

Next, the orientation difference, θ_{diff}, between that axis of elongation of lateral connections and the orientation preferences of their source feedforward kernels was analyzed. Results of the trained model discussed previously are shown in Figure 4.9C. In the figure, points were scaled by the normalized index of ellipticity. Larger markers show more anisotropic connections and are in general more aligned with the feedforward orientation preference. To account for wrapping effects around 0° and 180°, the smallest angular difference between θ_{diff} and $180 - \theta_{diff}$ were considered. This resulted in angular differences in the the range ±90°. Both the excitatory and inhibitory connections show a strong correspondence with feedforward kernels (mean excitatory θ_{diff}=29°, mean inhibitory θ_{diff}=31°). There were a few outliers where the axis of elongation was orthogonal to the preferred orientation of feedforward kernels. The results were consistent across the 5 independently trained models (population average excitatory $\theta_{diff} = 29° \pm 2°$ and inhibitory $\theta_{diff} = 29° \pm 4°$). The difference in orientations between lateral connections axis of elongation and feedforward orientation preferences was larger than what Sincich and Blasdel [2001] found, however the trend was similar; most lateral connections project in the same direction as the orientation preference of their associated feedforward kernel.

In other models with fixed connection structures (see Section 2.4), typically a similar size is used for both excitatory and inhibitory connections. Contrastingly, the model learns smaller omni-directional inhibitory kernels compared with excitatory kernels. Moreover, previous models align the orientation of lateral inhibitory kernels in the orthogonal-to-the-preferred direction of feedforward kernel. These are modelled after the observed orthogonal-to-the-preferred direction regions of inhibitory contextual interactions [Kapadia et al., 2000]. Contrastingly, the model learns inhibitory connections aligned with the preferred orientation of feedforward kernels. One possible reason for this discrepancy may be related to the input filtering done by these models. In the models of Li [1998], Piëch et al. [2013], at each spatial location only the input edge with the strongest magnitude passed through to the contour integration model. All other inputs at that location are suppressed. There is no interference from other edge extracting neurons responding to sub-optimum stimuli on the trajectory of the contour. The results here indicate this source of interference is strong and most inhibitory connections appear to be suppressing less relevant neurons in the same direction as the contour.

The elliptical structure of learnt excitatory lateral connections is in agreement with the hypothesized layout of lateral connections proposed in the Association Field Model [Field et al., 1993]. This structure was believed to be necessary to do curve-linear contour enhancement. In an update to their original model, Field et al. included short range omni-directional inhibition connections [Field et al., 2013]. Their inclusion was based on the results of Das and Gilbert [1999]. Many of the neurons in V1 in Layer 2/3 have dendrites and axons that branch out omni-directionally over short distances [Malach et al., 1993]. This is the case even for neurons that were close to hypercolumn boundaries and pinwheel singularities (see Section 2.2.4) where orientation preferences change sharply. Das and Gilbert [1999] measured the correlation between the firing rates of several pairs of V1 neurons. Many of these neuron pairs crossed orientation boundaries and point of sharp orientation changes. By using stimuli that specifically targeted these connections as opposed to long range horizontal connections, it was found that they are relatively independent of direction and largely suppressive. The learnt inhibitory connection structures of my model are consistent with this result (see Figure 4.10).

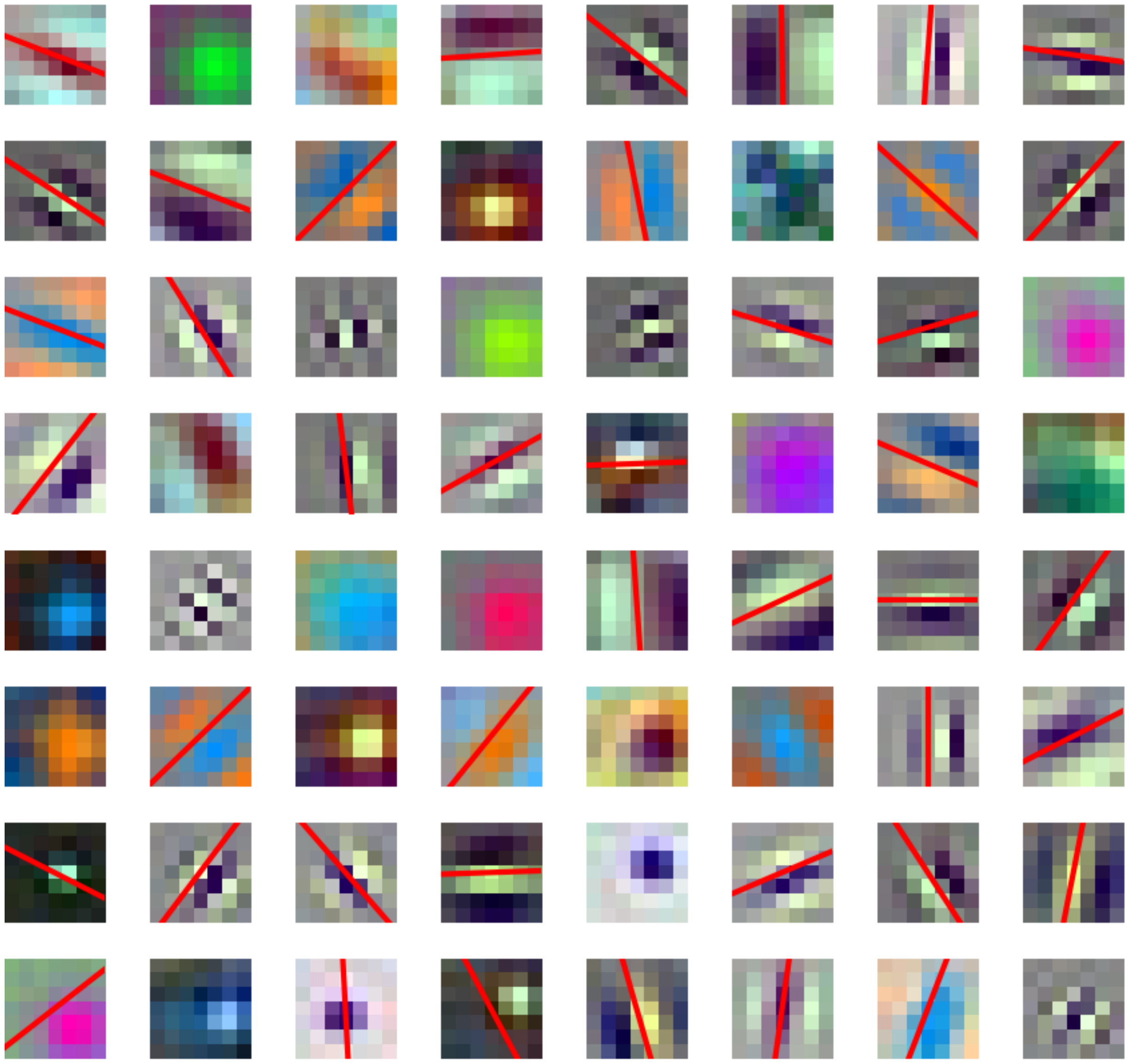

Figure 4.6: **Feedforward edge extraction kernels and their preferred orientations.** Best viewed in colour. Each subplot shows one of the 64 kernels of the first convolutional layer of a ResNet50 model that was trained on ImageNet. This served as the main component of the edge extraction block. Each kernel was fit to a 2D Gabor function to find its preferred orientation(red lines). Kernels for whom no fits were found (no red line) were skipped.

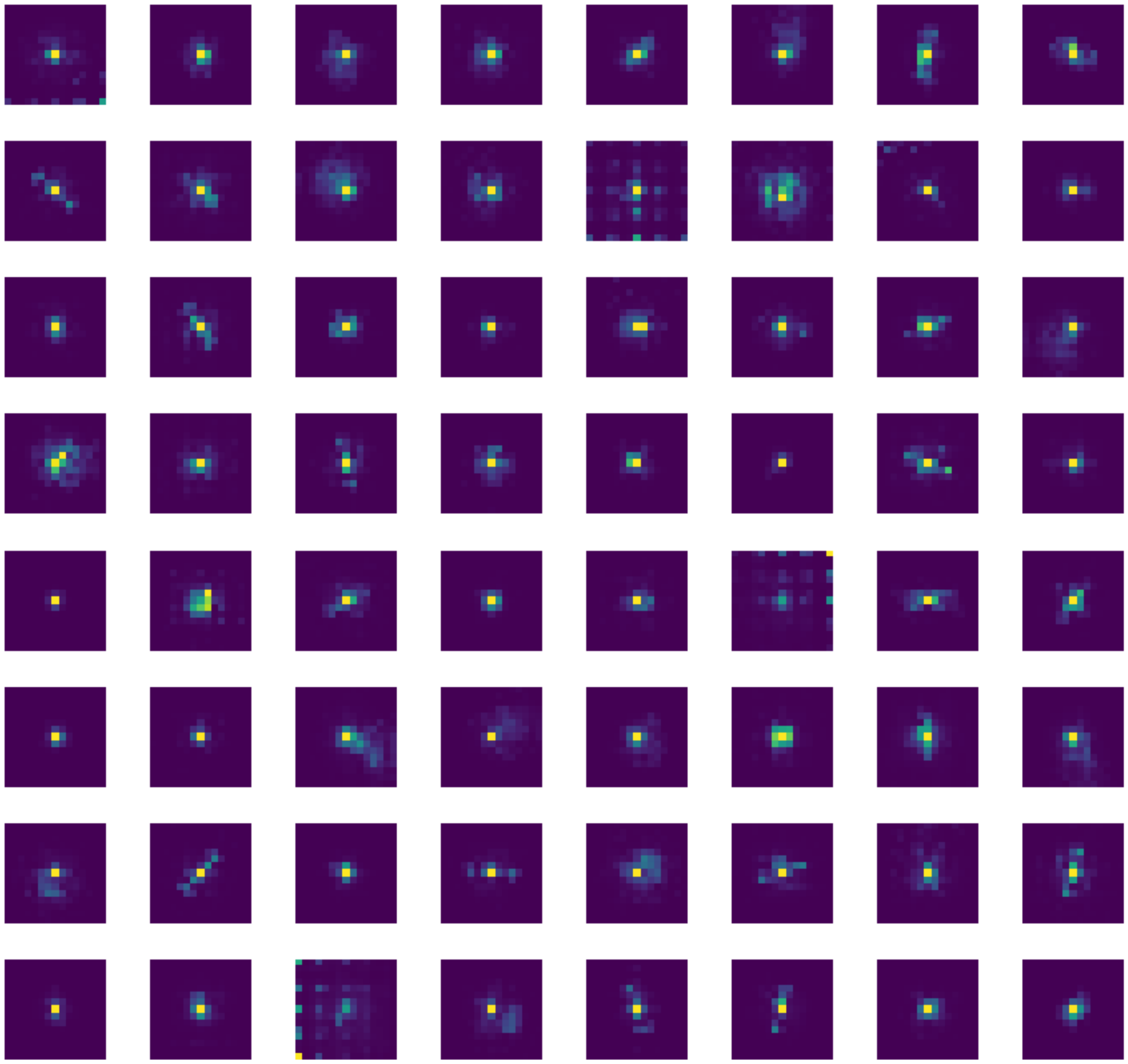

Figure 4.7: **Learnt lateral excitatory kernels of a trained model.** Each subplot plots one of 64 learnt lateral kernels. Individual kernels had 64 input channels and had a spatial spread of 15 × 15. To view the kernels, the channel dimensions were compressed by summing over all input channels. Many excitatory kernels appear to be highly directed, spreading out in one dimension more than others.

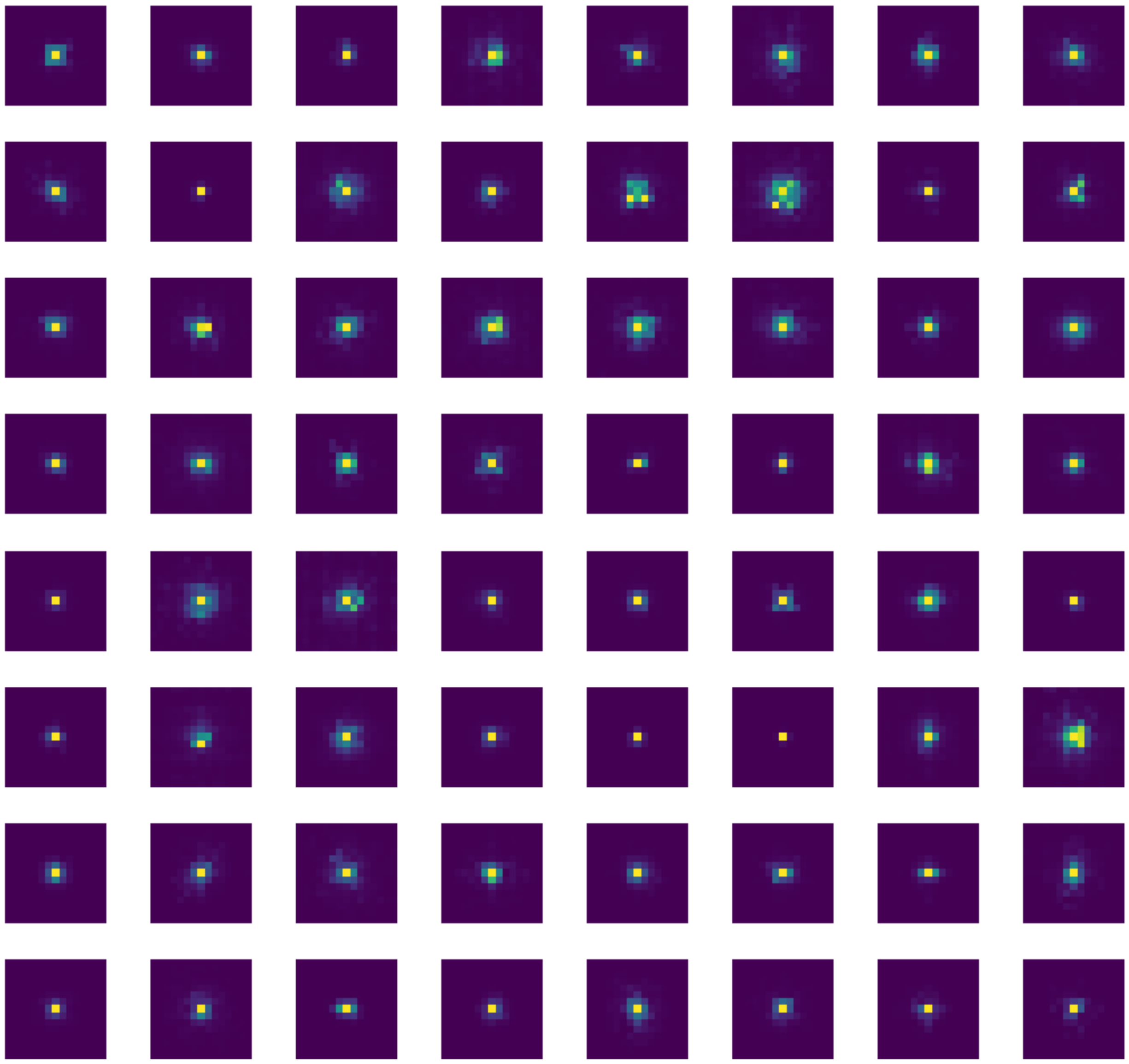

Figure 4.8: **Learnt lateral inhibitory kernels of a trained model.** Each subplot plots one of 64 learnt lateral kernels. Individual kernels had 64 input channels and had a spatial spread of 15 $\times$ 15. To view the kernels, the channel dimension were compressed by summing over all input channels. The spatial extent of inhibitory kernels was less than the spread of excitatory kernels and appears to be omni-directional.

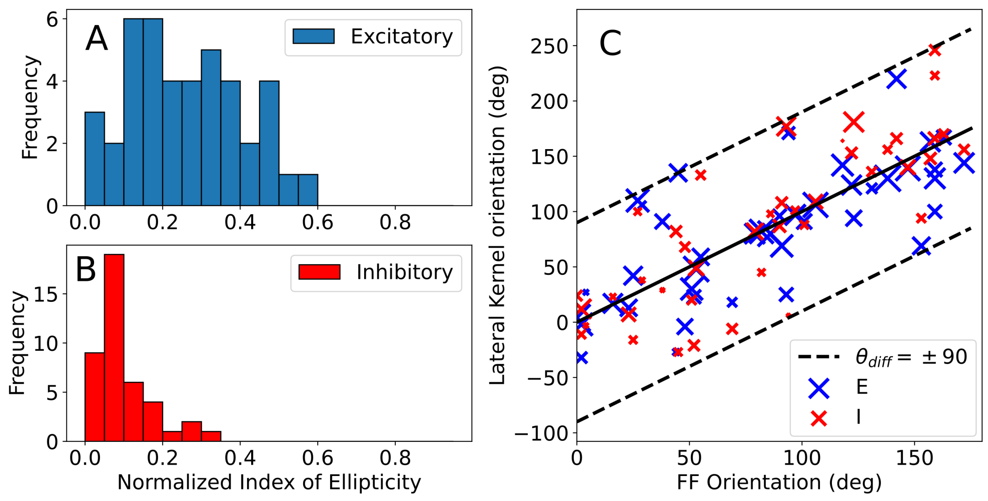

Figure 4.9: **Feedforward kernel orientations and lateral kernels axis of elongation.** Best viewed in colour. A and B, Histograms of normalized index of ellipticity of lateral excitatory and inhibitory connections respectively. Lateral excitatory connections spread out further and are more directed than inhibitory connections. C, Axis of elongation of lateral connections plotted against the orientation of their corresponding feedforward edge extraction kernels. Each point is scaled by its normalized index of ellipticity; larger markers are more directed kernels. Dashed lines show ±90° angular difference. Lateral kernels that lie on these lines are orthogonal to feedforward kernels.

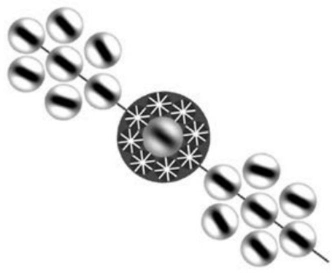

Figure 4.10: **Updated Association Field Model.** Figure reproduced from [Field et al., 2013]. Field et al. updated their theorized projections of lateral connections (association field model), by superimposing short-range omnidirectional inhibitory connections on top of the long range directed excitatory connections [Field et al., 2013]. The model proposed here, learns excitatory and inhibitory lateral connections aligned with their updated proposed layout.

4.5 Sensitivity analysis

In this section, I analyze the roles of individual CI block parameters and training hyper-parameters on task-level performance. Additionally, alternative choices for several selected training algorithms are compared. In the interest of time, models were trained with a subset of the data which consisted of 20,000 training and 2,000 validation images. Images were randomly selected from the contour fragments dataset and different subsets were chosen for each parameter. Except for the parameter under investigation, all other model and training parameters were set to their default values, as described in Sections 4.1 and 4.3. For each parameter setting, the model was trained for 100 epochs. Mean training and validation IoU scores were recorded after every epoch and parameter values were plotted against peak mean IoU scores across all epochs. Finally, none of the held-out test stimuli (see Section 4.4.2) were considered in the sensitivity analysis.

4.5.1 Model parameters

Within the CI block, interactions between E and I nodes repeat N_{iters} times before outputs are passed to subsequent layers. Figure 4.11A shows the effect of varying N_{iters} on task performance. Mean IoU scores continued to rise even with the highest tested N_{iters}. However, the relative increase above 8 iterations was negligible. Increasing the number of recurrent iterations significantly increases training time as well as memory requirements; RNN models use backpropagation through time (BPTT) [Werbos, 1990] to calculate gradients and this requires storing activations after each iteration across the model. As a compromise between performance and run time complexity, N_{iters} was set to 5 in the default settings.

The spatial extent of lateral kernels, S, defines how far E-I node pairs can communicate with their neighbors. The same size was used for both excitatory and inhibitory connections. The impact of varying S on model performance is shown in Figure 4.11B. Performance peaked for a spatial size of 15×15 before decreasing slightly and plateauing. As a result, S was set to 15×15 by default.

Figure 4.11C shows how internal connections of E and I node pairs affect performance. In this analysis, individual connection strengths, $\sigma(J_{xy})$ and $\sigma(J_{yx})$, of all model E-I node pairs were held constant. By default, these parameters are defined as trainable and are independently learnt by each channel-wise E-I pair. Hence, the analysis here only reflects their influences partially. Behavioural performance was not significantly impacted by fixed $\sigma(J_{xy})$ and $\sigma(J_{yx})$. However, peak validation performance was seen when both were set to ≈ 0.2. In the default settings, both $\sigma(J_{xy})$ and $\sigma(J_{yx})$ were initialized to 0.1 and were allowed to be adjusted by the learning process.

The effects of fixed membrane time constants of E and I nodes on performance are shown in Figure 4.11D. Similar to the analysis of connection strengths, individual time constants $\sigma(a)$ or $\sigma(b)$ were fixed for all E-I pairs but by default are defined as trainable. Behavioural performance was not significantly impacted by $\sigma(a)$ or $\sigma(b)$. Hence, membrane time constants of both E and I nodes were randomly initialized around 0.5 and were trainable.

4.5.2 Training parameters

The performance impact of changing the learning rate is shown in Figure 4.12A. On this smaller dataset, model performance was highest for a learning rate of 10^{-3}, while for the control the best learning rate was 10^{-4}. However, when the model was training on the full dataset, best model performance was achieved by setting the learning rate to 10^{-4}. Consequently, the default learning rate for both the model and the control was set to 10^{-4}.

Sparse lateral connections were modelled by including a sparsity constraint in the total loss function. The sparsity constraint was defined as L1 loss modulated by an inverted Gaussian mask (IGM L1 Loss) on lateral kernels, W_e and W_i (see Section 4.3). The same mask was used for both excitatory and inhibitory lateral connections. The relative weight of the sparsity constraint to the criterion loss in the total loss function was set to the default value of 10^{-4}. The effects of varying the width of the Gaussian mask on performance are shown in Figure 4.12B. Validation IoU scores plateaued at a mask width of 8, while training IoU scores continued to increase with mask width. However, the relative increase above a width of 10 was small. Consequently, mask width was set to 10 by default.

Figure 4.12C (blue curves) shows the performance impact of sparsity loss weight relative to criterion loss in the total loss function. The width of the Gaussian mask was set to the default value of 10. Peak validation performance was seen for a sparsity loss weight of 10^{-4}. Higher weights significantly attenuated performance while lower weights, plateaued at a slightly lower value. Consequently, default sparsity loss weight was set to 10^{-4}.

The chosen sparsity constraint was also compared with other more common forms of weight regularization including L1 and L2 loss,

$$L_{sparsity\_L1} = \sum \|W_e\| + \sum \|W_i\|, \tag{4.8}$$

$$L_{sparsity\_L2} = \sum \|W_e\|^2 + \sum \|W_i\|^2, \tag{4.9}$$

where $\|.\|$ and $\|.\|^2$ are the L1 and L2 norms of lateral kernels.

The effect of sparsity loss weight on task performance for these two weight regularization methods are shown in Figure 4.12C. For loss weights below 10^{-4}, the performance of models trained with IGM L1 Loss was always better than models trained with L1 weight loss while the performance of models trained with L2 weight loss sparsity were better than models trained with IGM L1 Loss. However, L2 sparsity loss penalizes large weights and tends to distribute activity amongst many smaller weights. This has an adverse effect on the results on the neurophysiological gains experiments; smaller distributed weights produce small neuronal activations and this results in large contour gain measurements for small changes in output activations. For weights 10^{-5} and above, performance flattened at similar values for all considered sparsity constraints. As a result, IGM L1 loss was used as the default weight sparsity constraint.

In Figure 4.12D, different criterion loss functions are compared. By default, Binary Cross Entropy (BCE) Loss was used (see Section 4.3). The loss function treats the two classes (contour fragment vs. background fragment) equally. However, in each input stimulus, there are many more background fragments than contour fragments. To account for this class imbalance, Xie and Tu [2015] proposed class-balanced BCE Loss,

$$H_p(q) = -\frac{1}{N}\sum_{i=n}^{N}\left[p * \beta \log(q) + (1-p)(1-\beta)\log(1-q)\right], \tag{4.10}$$

where $\beta = {}^{N_{bg}}/_{N_{total}}$ is defined as the ratio of the number of background fragments to total fragments in an image, p is the contour fragment label and q is probability of a contour fragment predicted by the model. β is a dynamically calculated parameter for each image and proportionally scales up the loss associated with contour fragments by the number of background fragments in the image while at the same time scales down the loss associated with background fragments proportional to the number of contour fragments. As can be seen in Figure 4.12D, using class-balanced BCE Loss resulted in worse performance with the default learning settings. Therefore, by default, BCE loss was used.

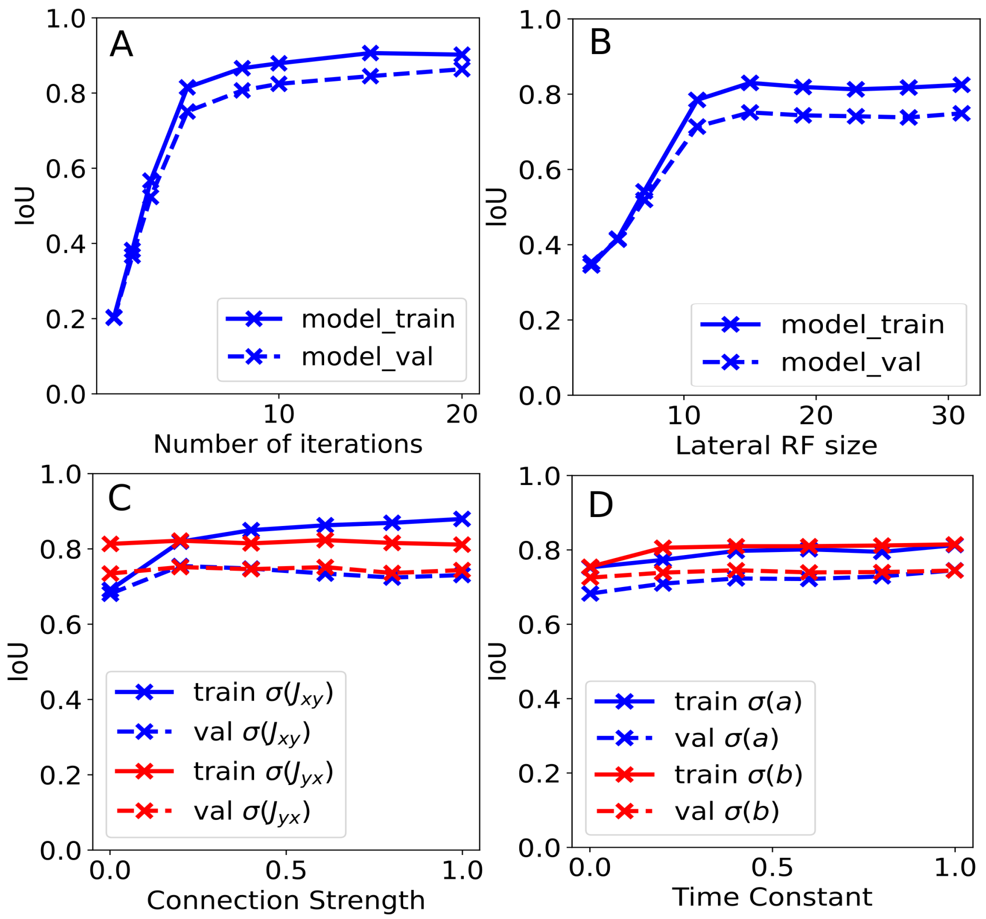

Figure 4.11: **Effects of CI block parameters on performance**. Best viewed in colour. A, Number of recurrent iterations. Performance increased as the number of iterations increased. Increasing the number of iterations above 8 did not significantly improve performance but increased training time substantially. B, The spatial extent of lateral connections. Performance peaked at a size of [15 × 15] before plateauing at a slightly lower IoU score. C, Fixed E $\rightarrow$ I (red) and I $\rightarrow$ E connection strengths. Behavioural performance was not significantly impacted by fixed values of J_{xy} or J_{yx}. D, Fixed membrane time constants of E (blue) and I(red) nodes. Similar to connection strengths, fixed time constant settings did not significantly impact performance.

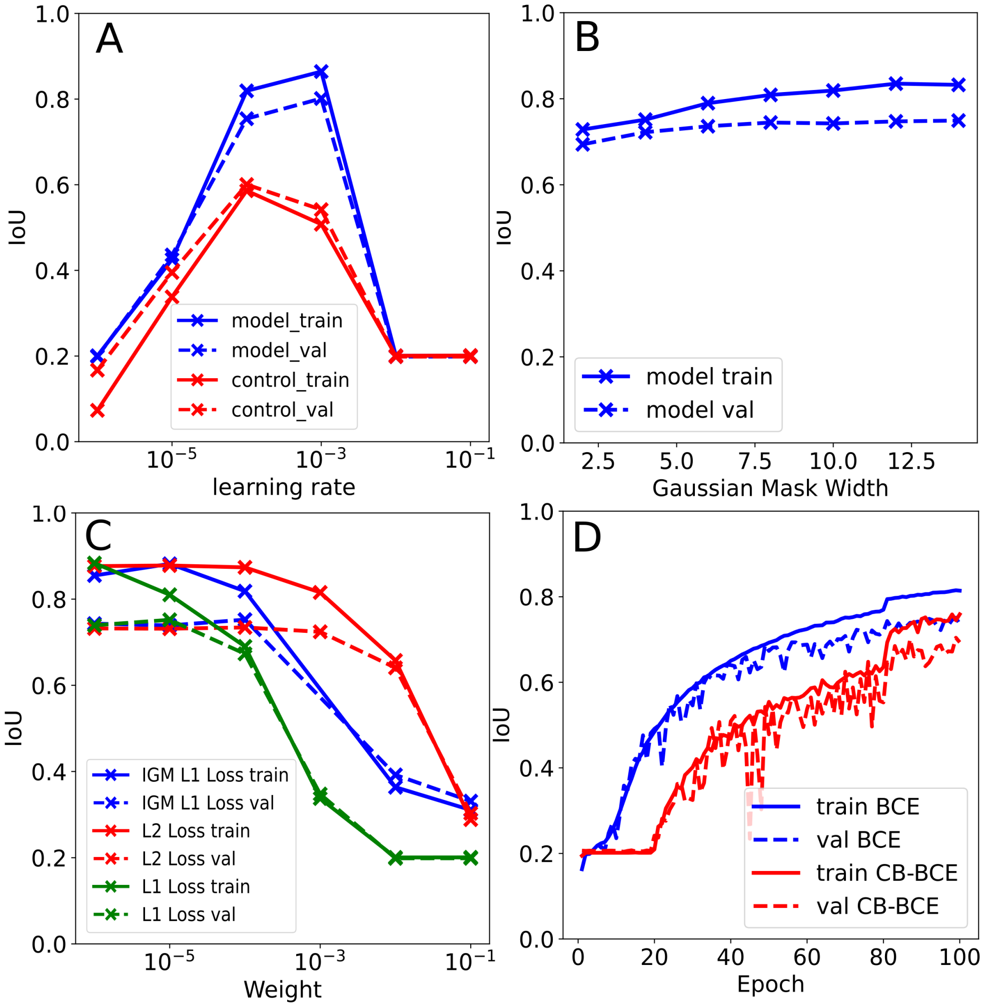

Figure 4.12: **Effects of training parameters on performance.** Best viewed in colour. A, Learning rate. On the smaller dataset used in the sensitivity analysis, peak model performance was seen for a learning rate of 10^{-3} while for the control the best learning rate was 10^{-4}. B, The effect of mask width of the inverted Gaussian masked (IGM) L1 Loss sparsity constraint on performance. Validation performance plateaued at values above 8 while training performance continued to rise with mask width. C, Performance impact of sparsity loss weight with respect to criterion loss. Three different sparsity loss functions were considered. At the default weight setting of 10^{-4}, IGM L1 Loss had the best performance. D, Different criterion loss functions. For the default learning rate of 10^{-3}, Binary Cross Entropy (BCE) outperformed class-balanced BCE.

4.6 CI block variations

In this section, I consider alternative formulations of the CI block to analyze the roles of its individual components as well as some of the constraints imposed during training. All considered variants were trained on the full contour dataset and used the same edge extraction and classification blocks as the original model; only the CI blocks were changed. Trained variants were compared with behavioural and neurophysiological data for straight contours of different lengths and inter-fragment spacing [Li et al., 2006]. Results were also compared with the results of the original model (Section 4.4.2). For each variant, results were averaged across at least 3 independent runs with different starting random seeds. Unless stated otherwise, all variants were trained with the default training settings described in Section 4.3.

4.6.1 Relaxed lateral connections positivity constraint

In the original model, lateral weights were forced to be positive by clipping negative weights to zero after every parameter update step while training. Pruning learnt negative lateral weights interferes with the learning process (by setting some weights to non-recommended values) and significantly slowed down training time. Positive-only lateral weights were necessary to compare learnt lateral connection structures with physical properties of V1 lateral connections. Other methods of encouraging positive lateral weights were also tried including, an additional L2 weight loss on negative lateral weights, convolving with exponential lateral kernels, and convolving with the absolute value of lateral kernels [Minni et al., 2019]. However, no significant improvements were seen and I continued with the simplest approach of clipping negative weights.

Alternatively, I tried a more relaxed positivity constraint on lateral kernels; interactions of lateral connections with the rest of the membrane potential dynamics were restricted to be positive but not the kernels themselves. This was accomplished by using a ReLU non-linearity on the outputs of lateral kernels and is similar to how positivity was enforced on local E-I connection strengths and membrane time constants. With this modification, the membrane potential equations of E and I nodes were defined as,

$$x_t = (1-\sigma(a))x_{t-1} + \sigma(a)\left[-\sigma(J_{xy})f_y(y_{t-1}) + I_{0e} + I + \mathrm{ReLU}(W_e \circledast f_x(X_{t-1}))\right], \tag{4.11}$$

$$y_t = (1-\sigma(b))y_{t-1} + \sigma(b)\left[\sigma(J_{yx})f_x(x_t) + I_{0i} + \mathrm{ReLU}(W_i \circledast f_x(X_t))\right], \tag{4.12}$$

where parameters and operators are identical to how they were defined in Section 4.1.

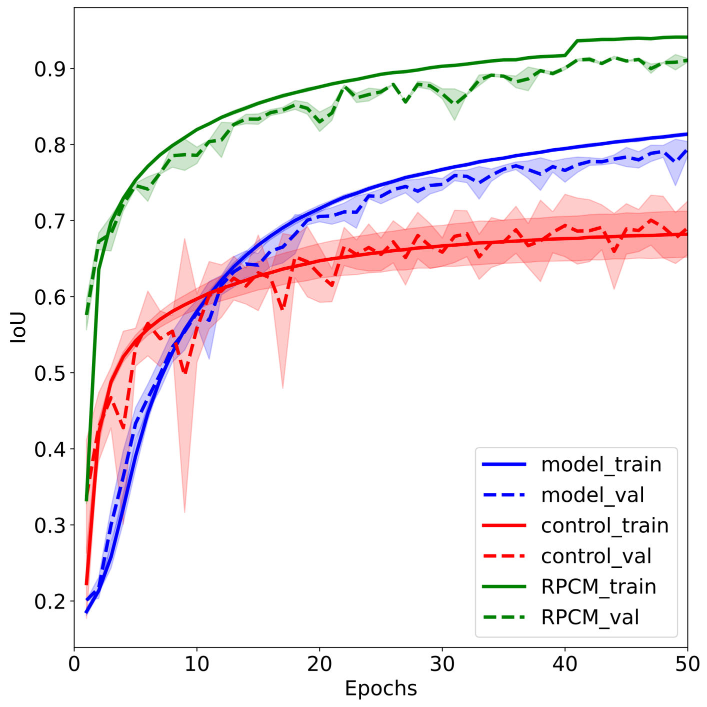

Figure 4.13: **IoU scores of the RPCM variant over the time course of training on the contour fragments dataset.** Best viewed in colour. Mean IoU scores of the model (blue), control (red) and the RPCM variant (green) vs. training time. Dark lines show mean values and the shaded area shown one standard deviation from mean values.

Importantly, this constraint did not interfere with the learning process. This improved the trainability and as a result the task level performance of the model variant. Others have also included this less strict constraint to model biologically plausible lateral interactions [Linsley et al., 2020b].

IoU scores of the relaxed positivity constraint model (RPCM) variant over the time course of training are shown in Figure 4.13. In the interest of time, RPCM networks were trained for 50 epochs only. The default starting learning rate of 10^{-4} was used but was dropped by a factor of 2 after 40 epochs. All other training parameters were set to their default values. As can be seen in the figure, trained networks had fully converged by the end of training. RPCM variants trained more efficiently (higher IoU score for same epoch) and substantially outperformed both the model and the control. Peak IoU scores, averaged across three runs, are shown in Table 4.3. On the complete dataset,

	Train (%)	Validation(%)	Straight Contours Only(%)
Model	$87.33 \pm 0.28\%$	$84.48 \pm 0.30\%$	$85.73 \pm 3.15\%$
Control	$71.62 \pm 0.35\%$	$73.61 \pm 0.38\%$	$82.16 \pm 5.53\%$
RPCM	$94.11 \pm 0.05\%$	$91.40 \pm 0.12\%$	$91.23 \pm 1.64\%$
CurrDi	$86.86 \pm 0.48\%$	$83.81 \pm 0.39\%$	$81.35 \pm 2.45\%$
RCNNM_a	$82.58 \pm 0.31\%$	$81.26 \pm 0.20\%$	$68.94 \pm 2.31\%$
RCNNM_b	$77.65 \pm 0.24\%$	$77.04 \pm 2.09\%$	$84.48 \pm 1.21\%$

Table 4.3: **Peak IoU scores of model variants on the contour fragments dataset.** Each row corresponds to a different model variant. RPCM is the relaxed positivity constraint model (Section 4.6.1), CurrDi is the current-based-divisive-inhibition variant (Section 4.6.2), RCNNM_a and RCNNM_b are two different solutions of the recurrent convolutional neural network model variant (Section 4.6.3).

the RPCM variant performed $\approx 7\%$ better than the model and $\approx 18\%$ better than the control (validation). The performance of the RPCM variant was similar on curved and straight contours. Contrastingly, both the model and the control found it easier to detect straight contours. However, even for straight contours, the RPCM variant outperformed both networks.

The effect of contour length on behavioural performance is shown in Figure 4.14A. For all considered contour lengths, IoU scores of the RPCM variant were higher than those of the model. Apart from the higher performance, trends in the results of the RPCM variant were similar to the model; performance dipped for length 3 contours, but increased monotonically for longer contours. Figure 4.14B shows how average contour integration gains of centrally located neurons changed as the length of contours was increased. Consistent with measured neurophysiological data, gains increased monotonically with contour length. However, measured gains of neurons in the RPCM variant were lower than those of neurons in the model.

The effect of inter-fragment spacing on average neuronal gains of monitored neurons is shown in Figure 4.14C. For inter-fragment spacing of up to 1.5 RCD, gains monotonically decreased with spacing, consistent with observed neurophysiological data. However, for larger fragment spacing, neuronal gains showed a slight increase and were inconsistent with neurophysiological data. The observed gain increases were smaller than those observed in neurons of control networks.

Histograms of gradients of linear regression fits of the outputs of monitored neurons

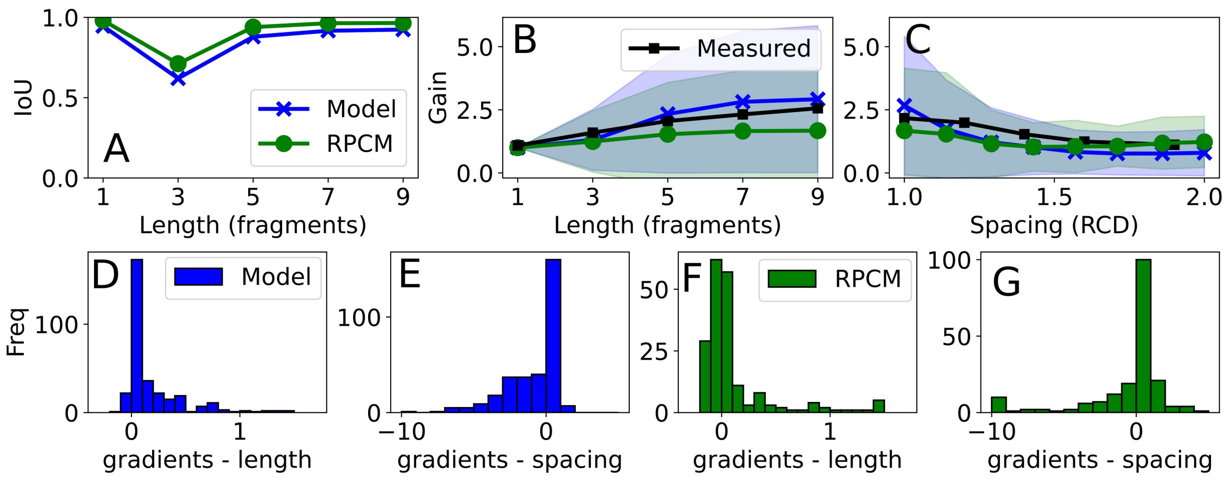

Figure 4.14: **RPCM variant results on the contour fragments task.** The RPCM variant is similar to the original model but did not enforce a strict positive-only constraint on lateral kernels. A, IoU vs. contour length for centrally located straight contours. Behavioural IoU scores of the RPCM variant were higher than the original model for all contour lengths. B and C, Population average contour integration gains vs. length and vs. fragment spacing respectively. Similar to the original model, gains monotonically increased with contour length. However, actual gains were smaller than those of the model. For inter-fragment spacing of up to 1.5 RCD, gains monotonically decreased. For higher inter-fragment spacing, slight increases in gains were seen. F and G, Gradients of linear fits of CI block outputs vs. length and vs. fragment spacing of individual neurons in RPCM networks. Individual neuronal trends were similar to population trends.

as contour lengths and inter-fragment spacing were increased are shown in Figure 4.14 F and G respectively. For the majority of neurons, output activations increased with contour lengths and decreased with fragment spacing, consistent with trends observed in population results. However, there were a few neurons whose outputs increased with spacing. The number of neurons that showed gain increases as inter-fragment spacing increased was larger in the RPCM variant than in the model.

4.6.2 Divisive E-I interactions

Piëch et al. [2013] also defined a current-based-divisive-inhibition (CurrDI) version of their contour integration model (see Section 2.4.3). The main difference between this version and their current-based-subtractive-inhibition (CurrSI) version, from which the original model

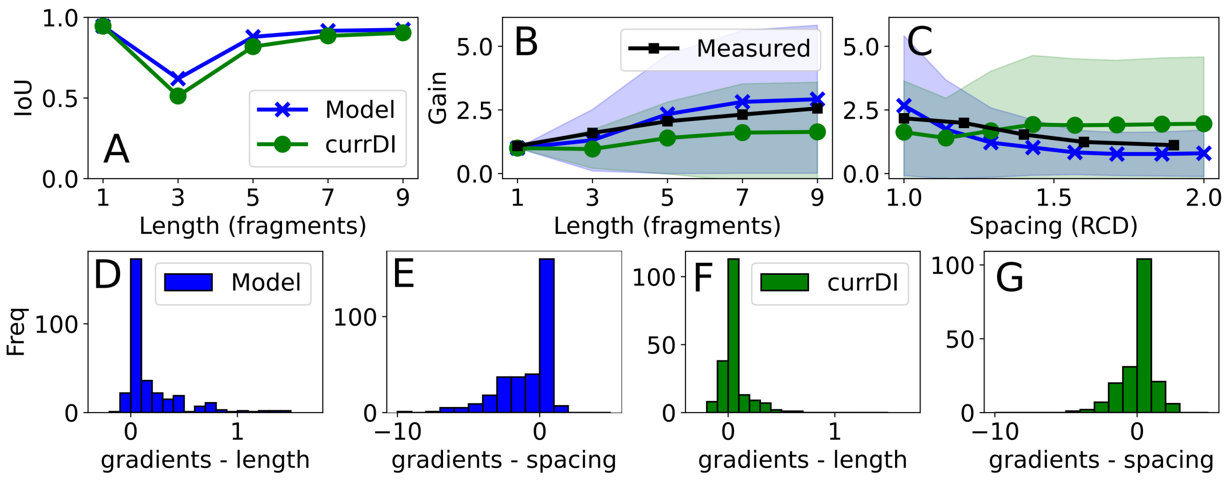

Figure 4.15: **CurrDi variant results on the contour fragments task.** The CurrDi variant is modeled after the current-based-divisive-inhibition model of Piëch et al. [2013]. A, IoU vs. contour length for straight contours. Behavioural classification performance of the currDi variant was similar to the original model. B and C, Population average contour integration gains vs. length and vs. fragment spacing. Contour integration gains increased with contour length. Gains increased with fragment spacing and were inconsistent with observed neurophysiological data. F and G, Gradients of linear fits of CI block outputs vs. length and vs. spacing curves of individual neurons, respectively. The majority of neurons showed increases in output activation as contour lengths increased and decreases in output activations as the spacing between fragments increased. However, a number of neurons showed output increases with fragment spacing. Compared to the model, a larger number of neurons showed outputs increasing with fragment spacing.

is derived, is in the way in which inhibitory neurons interact with excitatory neurons; inhibitory nodes interacted divisively rather than subtractively with excitatory nodes. Using the approach used to convert the CurrSI model into a trainable neural network model (see Appendix A), a trainable neural network model of their CurrDI model was constructed. The final forms of the membrane potentials of E and I node are given by,

$$x_t = (1-\sigma(a))x_{t-1} + \sigma(a)\left[\frac{I_{0e} + I + W_e \circledast f_x(X_{t-1})}{1+\sigma(J_{xy})f_y(y_{t-1})}\right], \tag{4.13}$$

$$y_t = (1-\sigma(b))y_{t-1} + \sigma(b)\left[\sigma(J_{yx})f_x(x_t) + I_{0i} + W_i \circledast f_x(X_t)\right], \tag{4.14}$$

where parameters and operators are identical to how they were defined in Section 4.1.

The CurrDi variant was trained using the default training settings. Similar to the original model, all lateral weights were constrained to be positive.

Peak IoU scores, averaged across 3 independent runs, for the CurrDi variant are shown in Table 4.3. Over the entire dataset, the performance of the currDi variant was similar to the original model. When only centrally located straight contours were tested, the performance of the currDi variant was $\approx 4\%$ worse than the model.

The effect of contour length on behavioural performance is shown in Figure 4.15A. Overall trends were similar to the model; performance dipped for length 3 contours and monotonically increased with larger contour lengths. The dip in performance for length 3 contours was larger for the currDi variant compared with the model. At this length, the model was $\approx 11\%$ better. For all considered lengths, model performance was always better that the currDi variant.

Figure 4.15B shows the effect of contour length on neurophysiological gains of centrally located neurons in the CI block of the CurrDi variant. No gain increase was seen for length 3 contours. For longer contours, gains increased monotonically with contour length. In general, neuronal gains were smaller than those of neurons in the original model. Figure 4.15C shows the effect of inter-fragment spacing on gains. Spacing results were inconsistent with observed neurophysiological data. For small inter-fragment distances (RCD ≤ 1.12), gains dropped with spacing. However, for larger spacing, gains initially increased and then plateaued at a value higher than the gain for the RCD=1 condition.

Histograms of linear fits to the output activation vs. contour length and vs. fragment spacing results of individual neurons of the currDi variant are shown in Figure 4.15F and G respectively. Neurons showed increases in output activations as contour lengths increased. However, gradients of increases were in general less than those of the model. Although the majority of monitored neurons showed decreases in output activations as inter-fragment spacing increased, a sizable proportion of neurons showed increases in gain as spacing increased. The number of neurons that showed increases was larger for the currDi variant then the model as well as the RPCM variant.

4.6.3 Recurrence with no E-I organization

To investigate the significance of the E-I organization of neurons, the model was compared with a variant based on recurrent CNNs [Liang and Hu, 2015, Spoerer et al., 2017]. The recurrent CNN variant (RCNNM) included the same number of recurrent steps and matched the capacity of the model. However, it did not include inhibitory neurons and therefore lacked the organization of component neurons in a 3D grid of E-I pairs. The

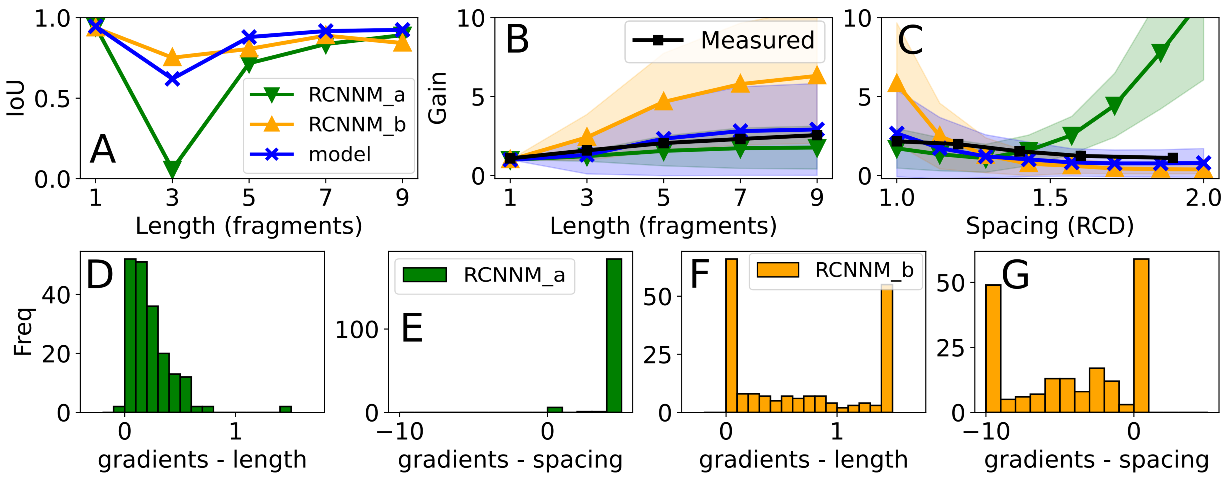

Figure 4.16: **RCNNM variant results on the contour fragments task.** The RCNNM variant is modelled after recurrent CNNs [Liang and Hu, 2015]. The variant had the same capacity and the same number of recurrent steps as the original model, but lacked its biologically inspired connection structure. Trained variants converged to one of two solutions: RCNNM_a and RCNNM_b. A, IoU vs. contour length for straight contours. For short contours, test IoU scores of the RCNNM_a variant was significantly worse than the RCNNM_b variant as well as the original model. B and C, Population average contour integration gains vs. length and vs. fragment spacing. Gains of both solutions increased monotonically with contour length, consistent with neurophysiological data. Gains of the RCNNM_a variant increased with as spacing between fragments increased, inconsistent with neurophysiological data. Gains of the RCNNM_b variant decreased monotonically with inter-fragment spacing. Gain trends in the RCNNM_b variant were consistent with neurophysiological data but were noticeably higher. D and E, Gradients of linear fits of outputs vs. length and vs. inter-fragment of individual channels of the CI block for the RCNNM_a variant. F and G, similar plots as D and E but for the RCNNM_b variant.

main purpose of this variant was to disentangle the role of recurrence from the biologically inspired organization of connections of the model.

The CI block of the RCNNM variant consisted of two convolutional layers and was defined as,

$$h_t = \sigma \left[\mathrm{BN} \left(W_x \circledast x + W_h \circledast h_{t-1} \right) \right], \tag{4.15}$$

where h_{t-1} is the hidden state activation at iteration $t-1$, x is the feedforward input from the edge extraction block, W_x, W_h are learnt convolutional kernels that act on the output activations of the preceding edge extraction block and hidden state activations respectively,

$\circledast$ is the convolutional operator, BN is a batch normalization layer and σ is a non-linear activation function. Similar to Liang and Hu [2015], lateral connections were modeled by the weight matrix acting over hidden activations.

Feedforward inputs propagated N_{iter} times through the CI block before hidden layer activations were passed to subsequent layers. Component convolutional layers were identical to those used in the model. Similar to the model, all convolutional layers of the CI block were constrained to be positive only. To ensure a fair comparison with the model, N_{iter} was set to 5 and the default training settings were used. A ReLU function was used as the non-linear activation function.

Peak IoU scores of the RCNNM variant are shown in Table 4.3. Trained RCNNM variants converged to one of two solutions: RCNNM_a, RCNNM_b. RCNNM_a occurred $\approx$ 62.5% of the time while RCNNM_b occurred $\approx$ 37.5% of the time. For each solution, results were averaged over three independent runs. Over the entire dataset, the RCNNM_a variant outperformed the RCNNM_b variant by $\approx$ 4% (validation IoU). Compared to the model, the performance of the RCNNM_a variant was $\approx$ 3% worse, while the RCNNM_b variant was $\approx$ 7% worse. When the two solutions were tested on straight contours that optimally stimulated centrally located neurons, the performance of RCNNM_a variants dropped dramatically ($\approx$ 12%). RCNNM_b variants fared much better on similar stimuli, and showed a performance improvement of $\approx$ 7% compared to its performance on the validation dataset. Finally, even on straight contours, the model showed higher performance than the RCNNM_b variant.

The effect of contour length on behavioural IoU performance is shown in Figure 4.16A. The RCNNM_a variant found it difficult to detect short (length 3 and 5) contours. IoU scores for length 3 contours were particularly bad (0.58 %) as the RCNNM_a variant failed to detect almost all component fragments. In contrast, RCNNM_b variants performed particularly well on length 3 contours and even outperformed the model by $\approx$ 11%. For all other contour lengths, the model outperformed both solutions of the RCNNM variant.

Figure 4.16B shows the effect of contour length on neurophysiological gains of centrally located neurons of the CI block. Both solutions showed monotonically increasing gains with contour length. However, neuronal gains in the RCNNM_b variant were significantly higher than those of the RCNNM_a variant, the model and observed neurophysiological data. A sharper contrast between the the two solutions was seen when the effects of fragment spacing were analyzed. Gains for the RCNNM_a variant increased with spacing and were inconsistent with neurophysiological data. Neuronal gains of the RCNNM_b variant decreased monotonically with inter-fragment spacing. Trends in the gain changes with spacing for the RCNNM_b variant were consistent with neurophysiological data but were noticeably higher.

Figure 4.16D and E show histograms of gradients of linear regression fits to output activations of individual neurons as contour lengths and inter-fragment spacing were increased for the RCNNM_a variant. Figure 4.16F and G show similar plots for the RCNNM_b variant. For both solutions, most neurons showed increases in output activations as contour lengths increased. Increases in outputs of neurons in the RCNNM_b model were larger. Almost all neurons in the RCNNM_a variant showed positive slopes as the spacing between fragments increased and were inconsistent with neurophysiological data. Moreover, increases in outputs were higher than those observed for neurons in the control. Furthermore, gains continued to rise even at the largest inter-fragment spacing. Neurons of the RCNNM_b variant, showed decreases in output activations as inter-fragment spacing increased. Observed decreases were larger than those observed in neurons of the model.

In summary, the most probable solution of the RCNNM variant, RCNNM_a, had comparable performance to the model on the task it was trained for. However, when it was tested with novel test stimuli, its performance dropped dramatically. An analysis of learnt contour integration gains show inconsistencies with observed neurophysiological data, especially when the spacing between fragments were increased. A second less probable solution, RCNNM_b, was also learnt. RCNNM_b variants had comparatively worse performance on the training task, but was much more robust to test stimuli at the behavioural level. RCNNM_b variants showed trends that were largely consistent with trends in the neurophysiology. However, observed gains were noticeably higher.

4.7 Comparison with existing models

In this section, I discuss similarities and differences between my proposed model and the hGRU model of Linsley et al. [2018] (see Section 3.4.3 for a review). In terms of methodology, the hGRU model is the most similar model; it is derived from an existing computational model of a V1 neural phenomenon and uses ANNs to learn lateral connections between component units. However, there are a number of differences that separate the two models in their objectives, architectures and the experiments that were carried out.

First, at a high-level, the objectives of the two models are different. The hGRU model was designed to address the inefficient detection of long-range spatial dependencies in CNNs. Consequently, its emphasis is on task-level performance. Many biological constraints were relaxed to achieve higher performance. On the other hand, the objective of my model was on modeling brain-like contour integration in ANNs. It more strictly adheres to biological constraints which allows it to be more directly compared with the brain and to be used to explore the brain's mechanism of contour integration.

Second, the hGRU model is derived from the computational model of Mély et al. [2018], while my model is derived from the model of Piëch et al. [2013]. The computational model of Mély et al. [2018] is in fact a model of surround modulation (see Section 2.3.5) rather than contour integration. Surround modulation is a neural phenomenon that focuses more on the suppressive impacts of stimuli in the extra classical receptive field (e-cRF) and typically involves using stimuli that excite the entire e-cRF simultaneously [Angelucci et al., 2017]. In comparison, contour integration stimuli selectively stimulate parts of the e-CRF to see how within cRF stimuli responses are enhanced.

Third, although the hGRU model includes inhibitory influences, it does not incorporate any inhibitory neurons. Inhibitory influences are generated from the outputs of surrounding units and affect a unit's input, while excitatory influences are generating from the inputs of surrounding units and affects a unit's output. Hence, individual units have both positive and negative influences on each other. The separation of inhibitory and excitatory neurons is a fundamental principle of the brain (Dale's principle); each neuron can be excitatory or inhibitory based on the set of neurotransmitters it releases [Dale, 1935, Eccles et al., 1954]. In contrast, inhibitory influences in my model are generated by separate inhibitory units which is more aligned with Dale's principle. It is also similar to how inhibitory influences are typically modelled in contour integration models (see Section 2.4).

Fourth, recurrent interactions in the hGRU model are inspired from complex GRU networks (see Section 3.3.2). Compared to vanilla RNNs, GRU networks are more trainable and expressive. While performance was not directly compared, it is likely that the hGRU model will achieve comparatively higher task-level performance. However, the hGRU model forgoes several biological constraints to improve performance. In particular, time constants of component units are replaced with learnable gates. These gates access the hidden state of other units (at the same location) to determine how to modulate the unit's activity. Not only is this conceptually different from the time constants (internal property of a neuron) that were replaced, but it also introduces multiple parallel paths for information to flow between units. These multiple parallel paths also make it difficult to compare component connections with those in the brain. In contrast, all connections in my model can be directly mapped back to the brain.

Fifth, in my model, a strict positive-only constraint is enforced on lateral kernels during training. This ensures that each learnt excitatory and inhibitory connection always operates in its intended direction. In the hGRU model, no similar constraint was imposed on lateral connections. If the ANN training process learns negative weights, inhibitory connections may turn into excitatory connections and visa versa. In later revisions [Linsley et al., 2020b], Linsley et al. force non-negativity on excitatory and inhibitory interactions by using ReLU non-linear activation functions on the combined outputs of lateral interactions. This is similar to the RPCM model variant (see Section 4.6.1) where excitatory and inhibitory lateral influences are forced to be as intended but not the weights themselves. Although this more relaxed constraint was found have have better task-level performance compared to forcing individual weights to be positive, it does not strictly conform to Dale's principle. Individual neurons can still have positive and negative influences on other neurons, depending on the polarity of the weights learnt during training.

Sixth, because of the different objectives, different experiments were carried out for the two models. In my analysis, the model was compared with the brain at the behavioural, neurophysiological and neuroanatomical levels. On the other hand, the hGRU was only analyzed at the behavioural and neuroanatomical levels. Moreover, even though Linsley et al. analyzed learnt lateral kernels, I conducted a deeper analysis of lateral kernels whereby axes of elongation of lateral connections were compared with the preferred orientation of source V1 neuron. The association field is a directed structure that spreads out more in the preferred direction of the cRF of neurons. Hence, to see if the model learnt lateral connections that are consistent with the association field, it is necessary to look at feedforward and lateral kernels together.

Finally, there was differences in the lateral kernels learnt by the two models. Based only on the structure of lateral kernels, Linsley et al. categorized them into three groups: (1) a near e-cRF region with excitatory connections and a far e-cRF with inhibitory connections (consistent with connection patterns hypothesized by surround modulation models), (2) association field like structures with co-linear excitation and orthogonal inhibition and (3) more complex excitatory and inhibitory regions that are extensions of on-off regions within the cRF of V1 neurons [Tanaka and Ohzawa, 2009]. In my analysis of learnt lateral kernels, I found many more examples of connectivity patterns as proposed in the updated model of the association field [Field et al., 2013]. In this model, inhibitory connections are omnidirectional and are much shorter than excitatory connections. The differences in learnt lateral connections may have resulted from different training tasks or the different approaches used to visualize lateral kernels.

4.8 Discussion

In this chapter, I proposed a new model of contour integration that is based on recurrent Convolutional Neural Networks (rCNNs). Compared to standard rCNNs [Liang and Hu, 2015, Spoerer et al., 2017], its internal architecture and lateral connections are more strictly aligned with those of the V1 cortex. The model also differs from existing computational models of contour integration as its parameters are learnt instead of being fixed at start up. The model establishes a measurable link between the low-level neural phenomenon and high-level goals of the larger system that it is a part of. This link can be used to quantitatively measure the role of contour integration on various high-level tasks, an area which existing computational models have not addressed in detail.

On synthetic data composed of contours formed from co-aligned fragments embedded in sea of identical but randomly oriented fragments, I show that the model can learn to integrate contours in a manner similar to the brain. This was validated by comparing the model with observed data at multiple points including at the behavioural, physiological and the anatomical levels.

At a behavioural level, the model outperformed a feedforward control of similar capacity by $\approx 11\%$ (peak IoU score). Moreover, for contours composed of 3 or more fragments, behavioural performance monotonically increased with contour length, consistent with observed data [Li et al., 2006]. At a neurophysiological level, neurons responding to contour fragments showed elevated responses. The amount of enhancement gains (compared to when they receive optimal stimuli in their cRFs only) seen by these neurons monotonically increased with contour lengths and monotonically decreased with spacing between fragments, also consistent with observed data [Li et al., 2006].

In contrast, the control feedforward network displayed similar behavioural trends, but differed markedly at the neurophysiological level; responses of neurons were largely indifferent to changes in contour length and surprisingly increased as the spacing between fragments increased. This demonstrates that it is important to compare ANN models at multiple levels to accurately equate models with systems in the brain [Lindsay, 2020].

Comparisons between ANNs and neuronal data are typically done using representation similarity analysis [Kriegeskorte et al., 2008] or by expressing individual neuron responses as a linear combinations of the activities of multiple units in a particular layer of an ANN [Yamins et al., 2014, Cadieu et al., 2014]. Both of these approaches can be considered population level comparisons; they do not compare individual units in the two systems directly. At a neurophysiological level, I was interested in seeing if individual units in the CI block behaved similar to neurons. However, exact quantitative correspondence between units of one complex system with those of another are difficult. It is especially difficult for ANNs as a majority of their parameters are learnt through training rather than being fixed at initialization. Hence, in this work, responses of networks units were compared with recorded data at a macroscopic level, where trends in responses rather than exact values were compared. Using this approach, it was shown that the proposed model matched observed neurophysiological data while the feedforward control did not. Finally, it is likely that better quantitative matches between model and neuronal responses may be possible by using existing computational models. These models have many tunable parameters that can be handcrafted to match neural data. Although this has not been done for the considered neurophysiological data [Li et al., 2006].

The number of parameters of the feedforward control network was made similar to the model by matching the number of convolutional layers and the number and size of kernels used in each such layer. Compared to standard feedforward models, the control network used relatively larger kernels. Many other feedforward architectures with a matching number of parameters are also possible. It is likely some of these may have better task-level performance. Spoerer et al. [2020] compared their rCNN model with three feedforward networks that have similar capacities but different architectures. In their control networks, parameters were matching by increasing the spatial size of kernels, increasing the number of kernels in convolutional layers (network width) and increasing the number of layers (network depth) respectively. They found that performance improved the most when network depth was increased and the lowest when the size of kernels was increased (consistent with previous results [Simonyan and Zisserman, 2014]). Moreover, even greater performance may be realized if network width, depth and input image resolution are increased proportionally under a constraint [Tan and Le, 2019]. However, the aim of the contour integration block was to model V1 lateral connections, which are known to spread out up to eight times the cRF of V1 neurons [Stettler et al., 2002]. Consequently, the model was only compared with a feedforward control with kernels of similar size.

Because the model uses recurrence, it also performs a larger number of computations per input as compared to the feedforward control. The model contained x1.0002 more parameters that the feedforward control. By default, the model was set to use five recurrent steps. Hence, compared to the feedforward control, it performs ≈ 5.001 more computations per image. Relatedly, it has a larger inference run time as well. However, this is inline with how contour integration operates in the brain. Contour integration affects late-phase responses of neurons rather than their initial responses (see Section 2.3.2).

Next, after analyzing the model at the behavioural and neurophysiological levels, learnt lateral connections of trained models were compared with observed properties of lateral connections of V1 neurons. It was found that excitatory lateral connections spread out anisotropically and are densest in the preferred direction of source V1 neurons, consistent with observed neuroanatomical data [Sincich and Blasdel, 2001]. The analysis was extended to lateral inhibitory connections as well. Lateral inhibitory connections were found to be shorter than excitatory connections and were mostly omnidirectional. Their structures differed from those assumed by most existing models of contour integration. In these models, inhibitory lateral connections are densest in the orthogonal-to-the preferred direction of source V1 neurons and have a spatial extent that is similar to excitatory lateral connections. Overall, the lateral connectivity suggested by my model is congruent with the hypothesized updated association field model proposed by Field et al. [2013].

To investigate the role of individual components of the model, different permutations of the model were also analyzed. In particular, the RPCM variant, which relaxes the strict positive-only constraint on lateral connections but still enforces their interactions with the rest of the model to be positive, was found to be easier to train and reached higher peak IoU scores compared to the model. Moreover, it was generally consistent with behavioural and neurophysiological data. However, when spacing between fragments increased beyond a certain value, it showed slight increases in contour enhancement gains, which was inconsistent with observed data. Although, these increases were substantially less than those shown by the control network.

The model was also compared to standard rCNN with matching capacity. These networks use recurrence but lack the organization of component neurons into E-I pairs. Trained networks settled into one of two solutions: RCNNM_a and RCNNM_b. Both solutions had lower behavioural performance then the model. RCNNM_a networks had higher behavioural performance compared to RCNNM_b networks. However, their neurophysiological comparisons showed inconsistencies with observed data. RCNNM_b variants on the other hand, were consistent at both the behavioural and neurophysiological level. In summary, both recurrence as well as the biological inspired architecture of the model were necessary to reliably replicate results across all three points of analysis.

Chapter 5

Tasks on natural images

In the previous chapter, it was seen that with synthetic fragmented contours stimuli, the proposed model learns to integrate contours similar to the brain; in a manner consistent with many observed behavioural and neurophysiological properties. Moreover, learnt lateral connections were consistent with observed V1 lateral connection patterns. However, these stimuli are unnatural and are not frequently encountered in our natural viewing environment. Yet V1 lateral connections exist in the brain. Hence, it must be possible to learn them directly from natural scenes. It has also not been established which high-level, frequently used tasks in our natural viewing environment, utilize and benefit from contour integration.

In this chapter, I explore different tasks involving natural images to see if the model can learn to integrate contours and what benefits it may provide. In particular I train the proposed model on two tasks, detecting all edges and tracing contours in natural images.

5.1 Edge detection

To investigate how contour integration may be learnt from our natural viewing environment, I first tried the task of edge detection in natural images. Edge detection involves detecting *all edges* in an image. In the literature, edge detection is often (and more commonly) used to refer to the task of object contour/outer boundary detection which focuses on detecting *object edges* only [Martin et al., 2001, Hariharan et al., 2011]. The distinction between the two tasks is important because contour integration is a low-level phenomenon, whereas object awareness relies on more abstract representations that are typically learnt in deeper layers. In the literature, only two datasets specifically target all-edge detection: the Multi-cue Boundary Detection Dataset [Mély et al., 2016] and the Barcelona Images for Perceptual Edge Detection (BIPED) Dataset [Poma et al., 2020]. For the analysis of the usefulness of contour integration for the task of edge detection, the BIPED Dataset was selected as contains more images.

5.1.1 BIPED Dataset

The Barcelona Images for Perceptual Edge Detection (BIPED) Dataset [Poma et al., 2020] is composed of a base set of 200 train and 50 validation (image, edge map) pairs. Input images consist of outdoor natural scenes and have a fixed size of 1280 $\times$ 720 pixels each. Edge maps were manually annotated by computer vision specialists. The training dataset was further expanded (by the authors) by using several data augmentation methods including: random image crops and rotations, horizontal flips and gamma corrections. The resultant augmented training dataset contains 57,600 images. Sample images and ground-truth edge maps are shown in Figure 5.1A and 5.1B. The complete dataset and scripts for generating the augmented dataset are available at `https://github.com/xavysp/MBIPED`.

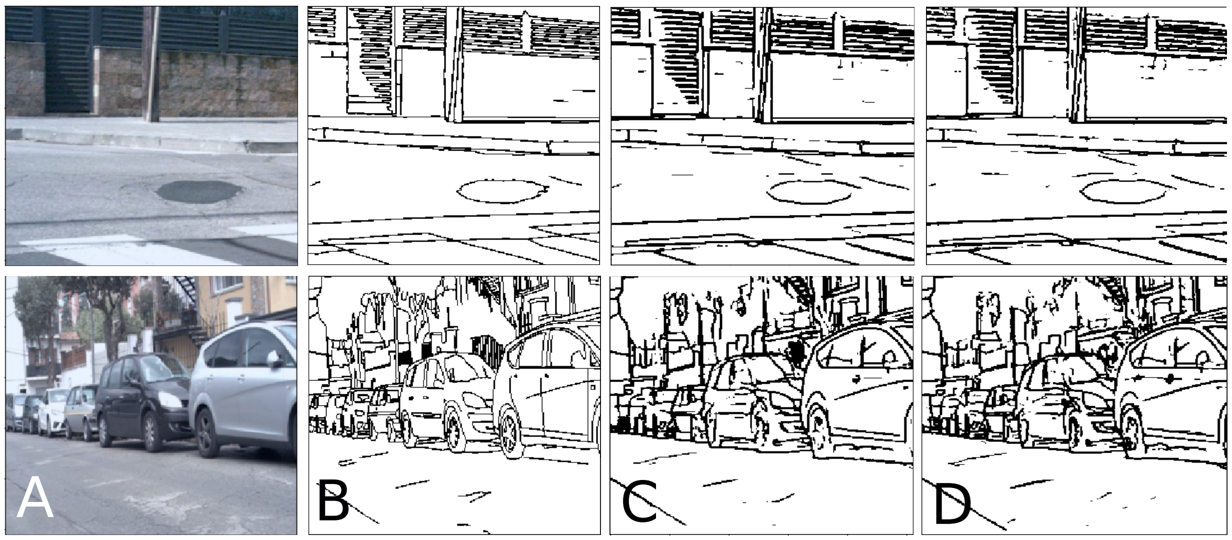

Figure 5.1: **Edge detection in natural images stimuli**. A, Example images from the BIPED [Poma et al., 2020] dataset. Each row show a different image. B, Ground truth edge maps for input images shown in column A. C and D, Corresponding predictions of the control and model, respectively.

5.1.2 Task

The task involved detecting all contours in input images. The expected output of networks was a prediction map, the same size as input images, where each pixel expresses the probability of whether it is an edge or not. Network performance were evaluated using mean IoU scores (see Section 4.3) between network predictions and ground-truth labels over all pixels in an image and all images in the dataset. For computing IoU scores, binary prediction maps of the model were generated by thresholding its outputs. Multiple thresholds, ranging from 0.1 to 0.8 with a step size of 0.1, were experimented with.

An edge detection block was used to map CI block outputs to the same dimensions as labels. The architecture of the edge detection block is shown in Figure 4.2. CI block outputs were upsampled by a factor of 4, using bi-linear interpolation, to resize them back to input sizes. Up-sampled activations were passed through two convolutional layers before prediction maps were generated. The first convolutional layer contained 8 kernels of size of 3×3 and used a stride length of 1. A batch normalization layer was added after the first convolutional layer. The last convolutional layer, contained a single kernel of size 1×1, and was used to flatten activations to a single channel dimension. Outputs of the final convolutional layer were passed though a sigmoid non-linearity to generate prediction maps. The capacity of the edge detection block was intentionally kept to a minimum to allow the CI block to do most of the work.

5.1.3 Training

Networks were trained using Binary Cross Entropy criterion loss as well as an inverted Gaussian masked L1 regularization loss over lateral kernels. Details of the loss function can be found in Section 4.3. The weight of the lateral sparsity loss with respect to the criterion loss was set to 10^{-4} and a Gaussian mask width of 10 was used. For the model, lateral kernel weights were forced to be positive by clipping all negative weights to zero after each parameter update. The control model was not similarly constrained and was free to choose weights that resulted in the best performance. All networks were trained with the Adam optimizer for at least 50 epochs. A batch size of 32 images were used. The learning rate was set to 10^{-3} and was reduced by a factor of 2 after 80 epochs.

Input images were pre-processed to have zero mean and unit standard deviation (SD) on average by normalizing using ImageNet channel means and SDs. All input images as well as ground-truth labels were resized to a fixed size of 256×256 pixels.

5.1.4 Results

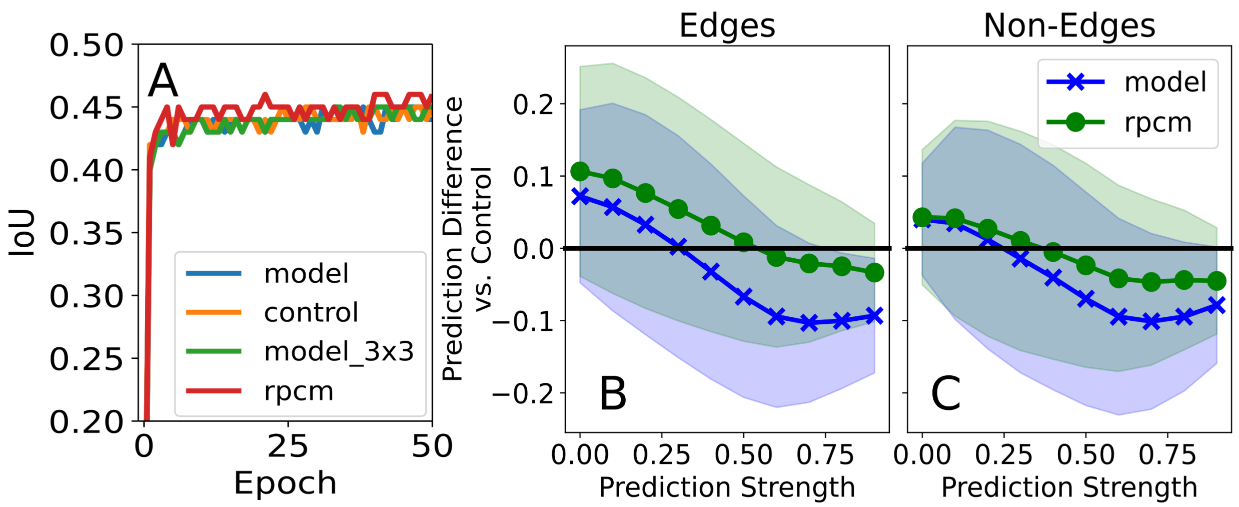

Figure 5.2: **Edge detection in natural images results**. A, Validation IoU scores of the model, RPCM variant and control networks over training. The model and control had similar performance. The RPCM variant showed slightly better performance. B and C, Average prediction difference between model and control over the validation dataset as a function of prediction strength. The sigmoid outputs of the model were compared with control outputs over the full validation dataset. Using a sliding window of width 0.2 and a step size of 0.1, control predictions within the window were highlighted and compared with corresponding model predictions. The average difference between the model and the control for edge pixels is plotted in B. In C average differences between the model and the control for non-edges is shown. Positive values indicate higher model predictions compared to control. Solid line shows mean differences and the shaded area shows unit standard deviation around the mean.

Sample predictions of trained control and model networks are shown in Figure 5.2C and D respectively. Visually, it was difficult to notice differences between their predictions.

Validation IoU scores over the time course of training, for a detection threshold of 0.3, are shown in Figure 5.2A. Both networks achieved their highest mean IoU scores at this threshold. IoU scores for the other thresholds dropped off monotonically as detection threshold deviated away from 0.3.

Peak validation IoU scores for both the model and the control was 0.45. Runs of the model and control were also trained for 100 Epochs (default is 50 Epochs). For both these networks IoU scores did not improve over 0.45. Even when peak IoU scores are compared, the two networks behave quite similarly. The contour integration block has little impact on overall performance, and it appears that the model does not need to integrate contours to detect them, at least on the considered dataset. To further confirm this, I trained a version of the model with a much reduced spatial extent; lateral kernels of size 3×3 (default is 15×15, see Table 4.1). The model reached the same peak performance (model_3x3 in Figure 5.2A). When a similar modification was made to the model on the synthetic contour fragments dataset, its performance dropped substantially, see Figure 4.11B.

In the synthetic contour fragments dataset, all input fragments were identical and the edge detection block did not provide any cues about which fragments were part of the contour. Hence, it was necessary to observe neighboring neurons to determine contour fragments. In comparison, for the task of edge detection in natural images, feedforward edge extraction provides sufficient information to detect contours and there is little need to integrate contextual information from neighbors. Relatedly, even for synthetic fragmented contours, Li et al. [2006] and Chen et al. [2017] observed that V1 neurons do not show enhancement gains when background fragments are absent, when edge extraction activations are clear and fragments can easily be distinguished from the background.

Weak vs. strong edge pixel detection

In the synthetic contour fragments dataset, all input image fragments are of the same strength. In natural images, contour pixels have non-uniform strengths and some contour parts are easier to detect than others. The models of Li [1998] and Piëch et al. [2013] showed that contour integration can potentially enhance weak contours in natural images. However, results were only qualitatively analyzed with a single image. It is not clear if this is a general tendency nor how it quantitatively affects behavioural performance.

To investigate whether the CI block of the model learned to enhance weak contours, model and control outputs were compared at different prediction strengths, pixel-by-pixel. It was necessary to compare model and control outputs directly as binary edge labels do not provide information on the detectability of individual pixels.

To compare predictions of the model and the control at different edge strengths, first pre-threshold control and model outputs over the entire BIPED validation dataset were collected. Second, a sliding window of size 0.2 was run over control outputs to highlight pixels whose predictions lay within the desired range. Third, corresponding predictions of the model were found. Fourth, the average difference between model and control predictions were recorded. The process was repeated over the full range of predictions (0, 1) by sliding the window at intervals of 0.1.

Both edge pixels and non-edge pixels were separately analyzed. To extract edge predictions, network outputs were multiplied with the ground truth mask. While to separate non-edge pixels, network outputs were multiplied with the inverted ground truth mask. Considering edge pixels, if the mean difference is above zero, this suggests that the model is better at detecting pixels of the corresponding strength. Considering non-edge pixels, if the mean difference is below zero, then the model has lower tendency for false positives. Code for carrying out this analysis is available at `https://github.com/salkhan23/contour_integration_pytorch`.

The results of these analyses are shown in Figure 5.2B for edge pixels and in Figure 5.2C for non-edge pixels. Compared to the control, on average the model had higher edge predictions up to edge strengths of 0.3. For higher edge strengths, the control was on average better. Although the difference between the model and the control plateaued above edge strengths of 0.6. For non-edge pixels, model outputs were on average lower than control outputs for all control predictions above 0.2. This shows the model has a lower false positive detection rate. Similar analyses were also carried out between the RPCM model variant (see Section 4.6.1) and the control network. The RPCM variant was on average better than than the control at detecting edges up to edge strengths of 0.55. Moreover, even at high edge strengths, the predictions of the RPCM variant were only slightly lower. For non-edge pixels, predictions of the RPCM model were lower than the control for all prediction strengths above 0.3.

Both the model and the RPCM variant are better at detecting weaker edges compared to the control. These results support the claim of Li [1998] and Piëch et al. [2013] that contour integration enhances weak contours. However, this enhancement had little impact on natural contour detection overall (IoU scores were the almost the same).

Effect of contour length on edge detectability

On the synthetic contours dataset, it was found that that longer contours are easier to detect than shorter ones. To investigate if contour length affects their detectability in natural images, the impact of contour lengths were analyzed. This required creating a new dataset of contours of known lengths in natural images. First, random smooth contours of various lengths were selected from the BIPED validation dataset. The procedure for selecting a smooth contour from an input image using its corresponding edge map is described in Section 5.2.1. Extracted contours were grouped into bins of length [20-49, 50-99, 100-149, 150-199, 200+] pixels. Each bin contained at least 50,000 edge pixels. At inference time, networks were not aware of selected contours and were fed-in original images from where selected contours were extracted. For each selected contour, a new ground-truth label was constructed that only highlighted the selected contour. Next, sigmoid predictions of networks corresponding to edge pixels were compared. For each contour length bin the mean difference between the model and control was calculated. Code for constructing the dataset and conducting the analysis is available at `https://github.com/salkhan23/contour_integration_pytorch`.

Results of this experiment were repeated over three different datasets where contours were independently collected. The control network's predictions were compared with the model and the RPCM variant. When the results of this experiment were analyzed, no discernible trend with contour length was seen for either network (results not shown). It appears contour length is not a determining factor for the detectability of contours in natural images, at least in the BIPED dataset.

5.2 Contour tracing in natural images

In Section 5.1, it was found that for clear natural images, it is not necessary to recurrently integrate contours in order to detect them as feedforward edge extraction outputs provide sufficient information. To further investigate how contour integration may be learnt in our natural viewing environment, a new task was created where edge extraction outputs are less helpful. The new task involved tracing image contours to determine if two appended markers were connected by a smooth contour. In some images, markers were placed on the same contour and were connected, while in others they were placed on different contours and were not connected. Markers were added directly to contours in input natural images, and networks were given no information about the selected contours. Finally, contours in the image were fragmented by puncturing images with occlusion bubbles. The task is inspired by the pathfinder task [Houtkamp and Roelfsema, 2010] which is typically used to explore behavioural properties of contour integration, but uses natural images rather than synthetic contour fragments.

5.2.1 Dataset

To construct the training dataset, natural images from the BIPED [Poma et al., 2020] dataset were used as a starting point. Next, modified input images and new labels were generated using the process described below.

Extracting smooth contours from natural images

The first step in constructing input images required the extraction of smooth contours from a natural image. A random smooth contour, C_1, was extracted from an (image, edge map label) pair in the BIPED dataset. Contours were extracted by first selecting a random starting edge pixel from the edge map. Valid starting pixels had to be part of a straight contour in their 3×3 pixel vicinity, either vertically, horizontally or diagonally. Next, this starting contour was extended at both ends by adding contiguous edge pixels that were at most $\pm\pi/4$ radians from the local direction of the contour. The local direction of the contour was defined as the angular difference between the last two points of the contour. If there were more than one candidate edge pixels, the candidate with the smallest offset from the contour direction was selected. The process was repeated until there were no more edge pixels at candidate positions or if the selected candidate pixel was already a part of C_1 (circular contour). Additionally, once contour length was greater than 8 pixels,

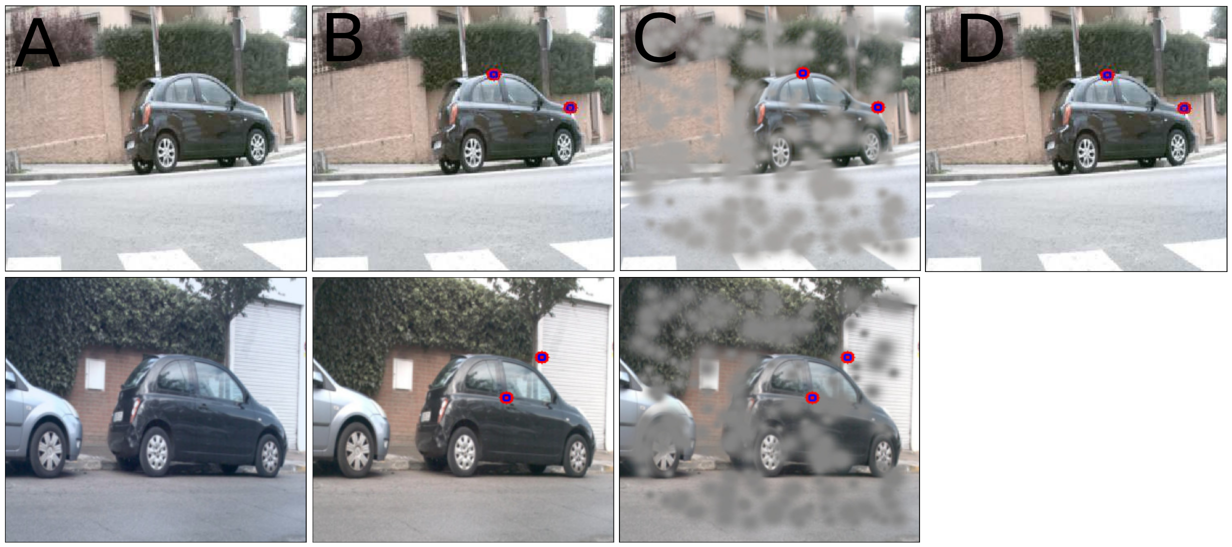

Figure 5.3: **Contour tracing in natural images stimuli.** Best viewed in colour. A, Sample image from the BIPED dataset [Poma et al., 2020]. B, Using its edges label, two markers were randomly placed on edge pixels. C, During training, images were punctured with occlusion bubbles to randomly fragment all image contours. D, After training, the impact of fragment spacing was analyzed using test contours with equidistant occlusion bubbles that were placed along contours with various inter-fragment spacing. The top row shows an example connected class stimulus, while the bottom row shows an example unconnected class stimulus. Only connected stimuli were used for analyzing the impact of inter-fragment spacing.

a large-scale smooth curvature constraint was applied to check that the angle difference between $(n, n-4)$ and $(n-4, n-8)$ contour points was not greater than $\pi/4$ radians, where, n is the last point on the contour. Contour extraction was also stopped if the large-scale curvature constraint was not met.

Stimulus construction

After extracting C_1, one of its endpoints was chosen as the position of the first marker, M_1. Next, a second edge pixel that did not lie on C_1 was randomly selected. To ensure that connected and unconnected stimuli had similar separation distances, the selection process used a non-uniform probability distribution to favor edge pixels that were equidistant with the unselected endpoint of C_1. The probability distribution was constructed by calculating the distance of all edge pixels in the image from M_1. Next, the absolute difference between edge pixels distances and the distance to the unselected endpoint of C_1 was calculated. A Softmax function was used to convert negative distances differences to probabilities. Edge pixels that were of a similar distance to the unselected end point of C_1 had distance differences close to zero and were more likely to be selected, while edge pixels that were at a different distance had large negative distance differences and were less probable.

Given the location of the second edge pixel, a second contour, C_2, was extended from it. If any point on C_2 overlapped with C_1, a new starting edge pixel was selected and the process was repeated until a non-overlapping pair of contours was found. The location of the second marker, M_2, was determined by the type of stimulus. For connected stimuli, the opposite end of C_1 was selected as M_2, while for unconnected stimuli, one of the endpoints of C_2 was chosen. Once marker positions were determined, markers were placed at corresponding positions in the input image. Each marker consisted of a bulls-eye of alternating red and blue concentric circles (see Figure 5.3B).

To fragment contours, occlusion bubbles were added to input images. Following Gosselin and Schyns [2001], bubbles with a 2D Gaussian profile were used to reduce the impact of bubble edges. Each image was punctured using 200 bubbles of multiple sizes. Bubble sizes were specified by the full-width-half-maximum (FWHM) of 2D Gaussian functions and were chosen to correspond to bubble sizes used to explore the effects of fragment spacing on neurophysiological gains (see Section 5.2.4). Individual bubbles were defined over a 2×FWHM square area. After randomly selecting bubble sizes and locations, bubbles were placed in a mask which was used to blend the input image with image channel mean values using,

$$\text{img}_{punc} = \text{mask} \times \text{img} + (1 - \text{mask}) \times \text{mean}_{ch}. \tag{5.1}$$

Within a mask, bubbles were allowed to overlap and a different mask was used for each image. Values in the bubble mask ranged between (0, 1). Sample input training images for the contour tracing task are shown in Figure 5.3C.

Complete Dataset

The train dataset contained 50,000 contours that were extracted from BIPED train images while the validation dataset contained 5,000 contours and were extracted from BIPED test images. Since the BIPED test dataset contains only 50 images, multiple contours per image were extracted. Care was taken to ensure duplicate contours were not selected. All contour containing input images were resized to a fixed size of 256 × 256 and stored with markers appended. Puncturing of input images was done as a pre-processing step. Consequently, each exposure of an image to a network was unique. Equal probabilities were used for generating connected and unconnected stimuli and the distribution of distances between the two markers for both classes was similar (see Figure 5.4). Code for generating this dataset from the BIPED dataset is freely available at `https://github.com/salkhan23/contour_integration_pytorch`.

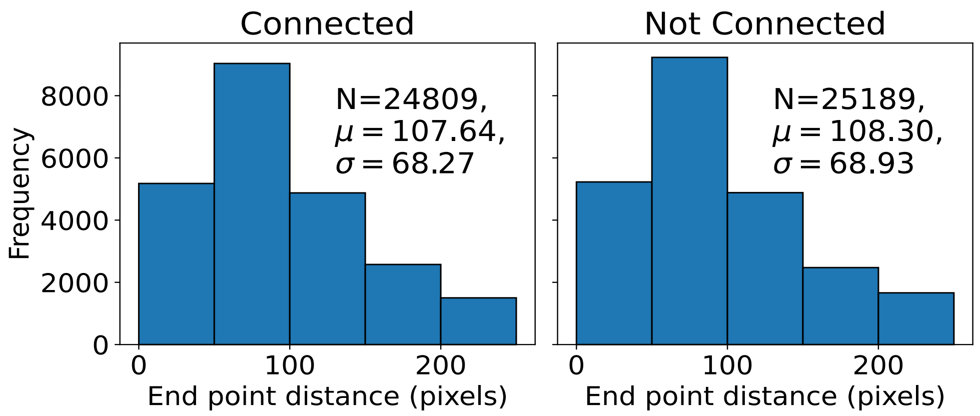

Figure 5.4: **Distribution of marker distances in the task of contour tracing in natural images task** Histograms of the distances between the markers for the connected and non-connected classes in the contour tracing in natural images training dataset.

5.2.2 Task

Networks were tasked with determining whether the two markers added to input images were connected by a smooth contour or not. Network performance was measured by comparing the accuracy of network predictions with true labels. To obtain binary predictions, network outputs were passed though a sigmoid activation function and were then thresholded at 0.5.

A binary classifier block was used to map CI block outputs to class labels. The architecture of the binary classifier block is shown in Figure 4.2. It consisted of 2 convolutional layers. The first convolutional layer consisted of 8 kernels of size 3×3 and used a stride length of 3. A batch normalization layer was added after the first convolutional layer. The final convolutional layer used a single kernel of size 1×1 and used a stride length of 1. Finally, a global average pooling layer [Lin et al., 2013] was added to map output activations to a single value that could be compared with image labels. The capacity of the classifier block was intentionally kept low to allow the contour integration layer to do most of the work.

5.2.3 Training

Networks were trained using the same loss function used on the synthetic contour fragments task (see Section 4.3) and included a Binary Cross Entropy criterion loss as well as the inverted Gaussian masked L1 loss on lateral kernels. The weight of the lateral sparsity loss with respect to the criterion loss was set to 10^{-4} and a Gaussian mask width of 10 was used. For the model, lateral kernel weights were forced to be positive by clipping all negative weights to zero after each parameter update. The control model was not similarly constrained and was free to choose weights that resulted in the best performance. Networks were trained with the Adam optimizer for 100 epochs. A batch size of 32 images was used. The learning rate was set to 10^{-3} and was reduced by a factor of 2 after 80 epochs.

Images were pre-processed to have zero mean and unit SD on average by normalizing using ImageNet channel means and SD. Pre-processing also included puncturing input images with occlusion bubbles (see Section 5.2.1).

5.2.4 Effects of fragment spacing

After training, the consistency of networks with trends in primate behaviour and neurophysiology were explored. Not many studies focus on neurophysiological properties of contour integration while viewing natural images. In this analysis, I try to generalize results from neurophysiological studies on synthetic fragmented contours [Li et al., 2006]. Test stimuli used in the analysis were different from those used in training. This is a form of out-of-distribution testing of networks and gives a better picture of model generalizability compared to testing with held-out test images that are generated from the same distribution as training images [Geirhos et al., 2020].

Test stimuli used contours with connected endpoints that were fragmented in an orderly manner; bubbles were evenly spaced along contours to fragment contours with fixed inter-fragment spacing. An example test stimulus is shown in Figure 5.3D. Contours were punctured with bubbles of sizes 7, 9, 11, 13, 15, 17 pixels, corresponding to fragment spacing of [7, 9, 11, 13, 15, 17]/7 RCD (see Section 4.4.2).

Binary classification accuracy was used to quantify behavioural performance while neurophysiological responses were quantified by the contour integration gain for natural images,

$$G_{NI}(rcd) = \frac{\text{CI Output @ RCD} = \text{rcd}}{\text{CI Output @ RCD} = 1}. \tag{5.2}$$

Here, the relative co-linear distance (RCD) is the ratio of inter-fragment spacing to fragment length in pixels, CI Output @ RCD = 1 is the output activation of an individual neuron responding to its optimal stimulus within the cRF and with the contour fragmented with gaps the same size as the stimulus, while CI Output @ RCD = rcd is the response of the neuron when the spacing between fragments was changed.

Test Stimuli Construction

The first step in measuring G_{NI} of individual channels of the CI block required finding their optimal stimuli. In the synthetic contour fragments dataset, input images were altered to find the optimal stimuli of monitored neurons. However, for natural images, inputs cannot be similarly changed. Therefore, a new procedure was devised. To find optimal stimuli of an individual channel, multiple unoccluded connected contours were presented to networks (Figure 5.3B). For each image, the position of the most active neuron of each channel in the CI block was found. If it was within 3 pixels (the same as the stride length of the subsequent convolutional layer) of the contour, the image as well as the position of most active neuron were stored. The process was repeated over 5,000 contours and the top 50 (contour, most active neuron) pairs were retained for each channel. New random contours were selected from the augmented BIPED train dataset. The train dataset, as opposed to the test dataset, was used as it contained more images and a larger variety of contours.

Given the optimal stimulus for a channel, each input contour was fragmented by inserted occlusion bubbles at specific positions along the contour. Different bubble sizes were used to fragment contours with different inter-fragment spacing. A fixed fragment length of 7 pixels, the same size as the cRF, was used. To ensure the cRF of the most active neuron was unaffected by bubbles, first, the position of the closest point on the contour was found. Bubbles were then inserted along the contour at offsets of $\pm\ ^{(l_{frag}+l_{bubble})}/_{2}$, $\pm^{3(l_{frag}+l_{bubble})}/_{2}$, $\pm^{5(l_{frag}+l_{bubble})}/_{2}$, ... until the ends of the contour. Finally, the blending-in area of bubbles was restricted to FWHM pixels to ensure visible contour fragments were unaffected. Code for conducting this experiment is available at https://github.com/salkhan23/contour_integration_pytorch.

5.2.5 Results

Behavioural (task level) performance

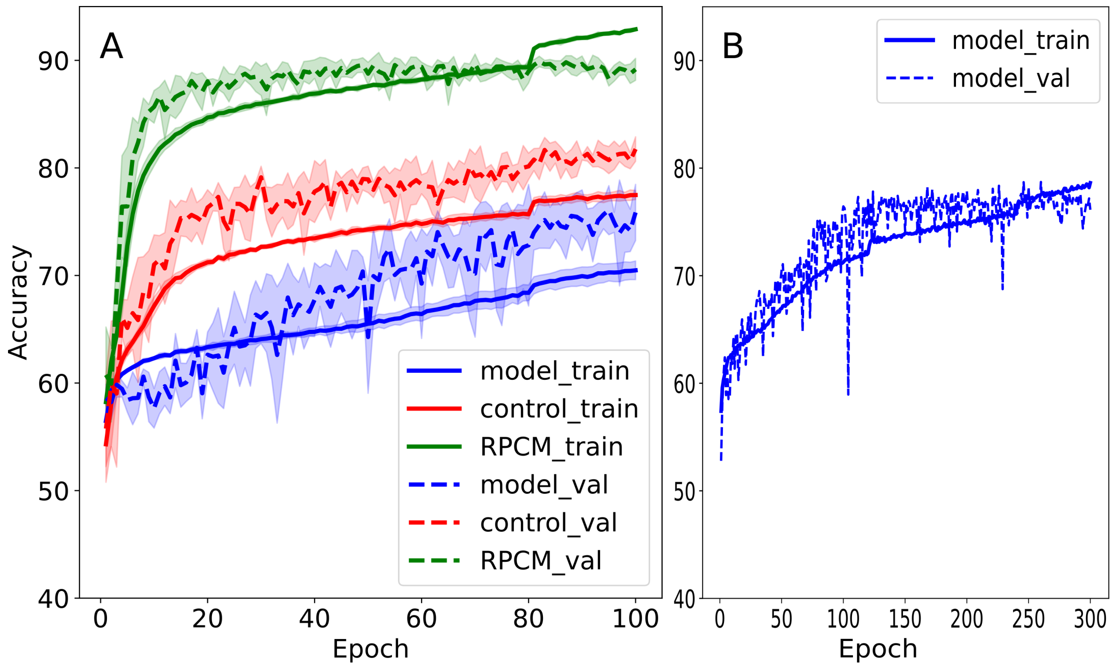

Figure 5.5: **Network accuracy vs. time on the contour tracing in natural images task.** Best viewed in colour. A, Model (blue), control (red), and the RPCM model variant (green) classification accuracies vs training time. Results were averaged over 5 separate runs. Lines show mean values across individual runs and shaded areas show unit standard deviation from mean values. By default all networks were trained for 100 Epochs. B, One run of the model was trained for 300 Epochs. Training accuracy continued to improve but validation accuracy plateaued at the same value as when models were trained for 100 Epochs.

Classification accuracies over the time course of training for the model, control and the RPCM model variant are shown in Figure 5.5A. For each network, results were averaged over 5 independent runs with different random seeds. Classification accuracies of the control and RPCM variant rose quickly and converged within the 100 epochs used for training. In contrast, model performance gradually improved and did not fully converge by the end of training. As previously noted (Section 4.6.1), the strict positive-only lateral weights constraint enforced on the model significantly impacts its training. To see if its performance

Name	Train (%)	Validation (%)	Test (%)
Model	70.52 ± 0.95	77.27 ± 1.55	70.39
Control	77.54 ± 0.44	82.67 ± 0.53	65.65
RPCM variant	92.90 ± 0.14	90.62 ± 0.21	84.36

Table 5.1: **Peak classification accuracies on the contour tracing in natural images task.** Networks were trained for at least 100 Epochs each.

would improve given more training time, a trial of the model was trained for 300 epochs (see Figure 5.5B). Training IoU continued to improve with training time, however validation IoU scores plateaued at 78%. As model performance did not significantly improve but training time increased substantially, multiple runs of the model were trained for 100 epochs, similar to other networks. Table 5.1 shows peak classification accuracies averaged across multiple runs of all considered networks. Over the whole dataset, the model performed $\approx 5\%$ worse than the control (validation IoU). The RPCM variant, which only differs from the model in the use of the strict positivity constraint, trained much more efficiently (higher performance at same epoch) and significantly outperformed the control by $\approx 8\%$.

When occlusion bubbles were added along contours (test data, RCD=1), classification accuracies of all considered networks dropped even for the smallest bubble size (Table 5.1 Test column). However the relative drop in performance for the model and the RPCM model variant ($\approx 6\%$) was significantly less than that of the control ($\approx 17\%$), showing that the strategy employed by the model generalized better. Figure 5.6A shows the results of fragment spacing on the behavioural performance of networks. From the least to the most spacing configuration, model performance dropped by $\approx 4\%$ while RPCM variant performance dropped by $\approx 3\%$, consistent with observed behavioural trends. Contrastingly, performance of the control was unaffected by the spacing between fragments.

Neurophysiological results

Figure 5.6B shows population averaged contour integration gains, G_{NI}, as the spacing between fragments increased. For each network, population average G_{NI} was found by averaging gains of individual channels for whom the optimal stimuli were found across all trained models. Model results were averaged across 293 neurons, RPCM variant results were averaged across 257 neurons while control results were averaged across 120 neurons. Neurophysiological gains of all networks dropped as spacing increased, consistent with observed neurophysiological trends. Gains of the model and the RPCM variant dropped

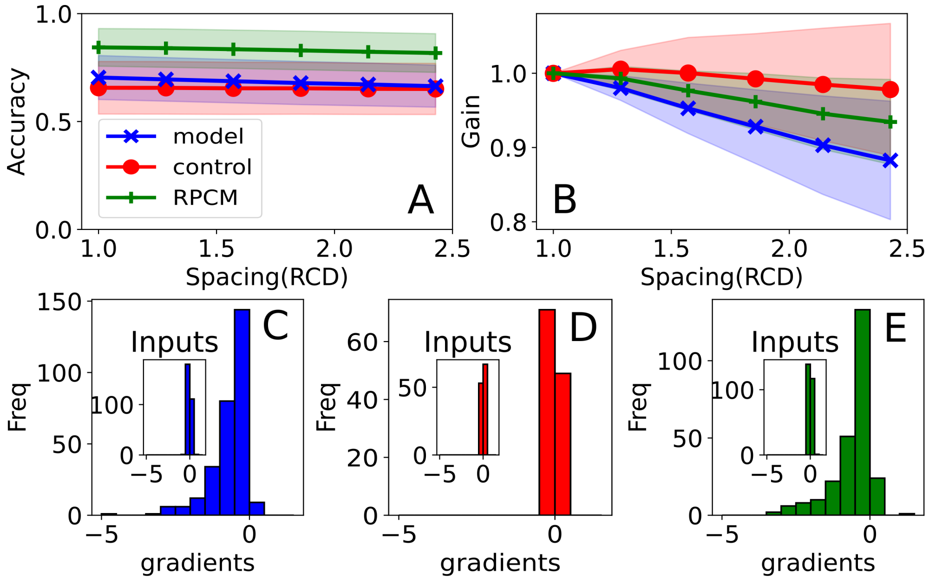

Figure 5.6: **Contour tracing in natural images results.** A, Classification accuracy of the model (blue), control (red) and RPCM model variant (green) vs. fragment spacing. Dark lines show mean accuracies and shaded areas shows unit standard deviation around means. Performances of the model and the RPCM variant dropped as spacing increased, consistent with observed behavioural trends. B, Population average G_{NI} vs. spacing. Gains of all networks dropped with spacing. C, D and E, Histograms of gradients of linear fits of gain vs. spacing results for the model, control and RPCM variant. For all networks, gains decreased as spacing increased. The model and the RPCM variant were more sensitive to inter-fragment spacing. Insets show histograms of gradients of CI block input activations vs. fragment spacing.

more sharply with spacing compared with the control.

The impact of fragment spacing on output activations was analyzed by doing linear regression fits on output activations vs. fragment spacing results for all monitored neurons. Histograms of gradients of the fits are plotted in Figure 5.6C, D and E for the model, control and the RPCM variant respectively. Model and RPCM variant output activations dropped sharply with spacing (consistent with neurophysiological responses to more artificial stimuli). While control output activations only dropped slightly as spacing increased. The results are consistent with trends observed in population averages. Inset plots in Figure 5.6C, D and E show gradients of linear fits of input activations with spacing. For all networks input gradients did not change significantly with spacing, showing that observed trends were learnt by CI blocks.

5.3 Discussion

In this chapter, the contour integration model was evaluated on two natural images tasks. The first task involved detecting all-edges, as opposed to object-only edges, in natural images. Over the BIPED dataset [Poma et al., 2020], model and control networks had similar performance and attained the same mean peak IoU score of 0.45 on the validation dataset. The RPCM variant (see Section 4.6.1) fared slightly better and achieved a peak IoU score of 0.46. The results suggest that in clear natural images (like the ones in the BIPED dataset), it is not necessary to recurrently integrate contours in order to detect them. At least not in the manner that is possible with lateral connections alone whereby all smooth contours are enhanced. This was further confirmed by testing a version of the model with severely curtailed lateral connections; the spatial extent of lateral connections was reduced to 3×3 (Default is 15×15). This version of the model also reached the same peak IoU score of 0.45. When a similar modification was made on the synthetic contour fragments dataset, the performance of the model dropped dramatically (See Figure 4.11B).

In synthetic contour fragment stimuli, all input fragments are identical and the edge extraction block did not provide any information about which fragments were part of the contour and which were not. Consequently, it was necessary to observe neighboring neurons to determine contour fragments. In contrast, in clear natural images, outputs of the edge extraction block contain sufficient information about the location of contours; there is little need to integrate contextual information from outside the cRF of neurons to determine if they contain contours or not.

It has been suggested that contour integration may strengthen weaker contours in natural images [Li, 1998, Piëch et al., 2013]. To investigate, I compared the pre-threshold predictions of networks at different strengths over the entire BIPED validation dataset. It was found that indeed, model and RPCM networks did enhance weaker contour pixels compared to the control network. However, this did not affect their overall IoU scores, at least not on images in the BIPED dataset.

The second task networks were evaluated on was a novel task in which edge extraction outputs were designed to be less helpful. The task involved detecting if two markers added into natural images were connected via smooth contour or not. Image contours were also fragmented by randomly puncturing inputs with occlusion bubbles of different sizes [Gosselin and Schyns, 2001] to simulate variable inter-fragment spacings. It was found that the control trained more efficiently (higher mean classification accuracy at the same epoch) than the model. Moreover, its peak IoU accuracy was also $\approx 5\%$ higher. RPCM variants were also analyzed. These networks trained even more efficiently than the control and outperformed them by $\approx 8\%$ in peak IoU scores.

Next, trained networks were tested on slightly different stimuli where occlusion bubbles were strategically placed along contours to simulate fixed inter-fragment spacings. The performance of all networks dropped even for the smallest bubble sizes. However, the drop in performance for the control ($\approx 17\%$) was significantly higher than the model ($\approx 6\%$) and the RPCM variant ($\approx 6\%$). This shows that the strategy learnt by the model generalized better.

Furthermore an analysis on the effect of spacing on the classification accuracies of networks showed that performance of the model and the RPCM variant dropped as the spacing between fragments increased. Conversely, the performance of the control was indifferent to the spacing between fragments. When neurophysiological gains were analyzed, neuronal outputs of all networks dropped as spacing increased. The drop was sharpest for the model, followed by RPCM variant and then the control. The results of the model were most consistent with neurophysiological responses to more artificial stimuli.

Chapter 6

Conclusion

6.1 Summary of contributions

In this thesis, a new biologically-inspired, recurrent convolutional neural network based model of contour integration is proposed. The model sits on top of an edge (low-level features) extraction block and models lateral interactions between component neurons. It is based on the lateral connections of neurons in the primary visual cortex of the brain. Unlike most existing contour integration models, connections were not pre-configured but were learnt by optimizing performance on high-level behavioural tasks.

The model was trained with synthetic stimuli composed of contours formed from co-aligned fragments embedded in a sea of identical but randomly oriented fragments. Similar stimuli are widely used to study the properties of contour integration in the brain. On datasets composed of these stimuli, the model out-performed feedforward CNN control models of similar capacities.

The trained model replicated many observed behavioural and neurophysiological properties of contour integration. Specifically, at a behavioural level, the model's ability to detect contours increased monotonically with contour length (see Figure 4.5A). At a neuronal level, optimally stimulated neurons that were responding to contour fragments showed elevated outputs in response to aligned contextual stimuli that were outside their classical receptive fields. Enhancement gains monotonically increased as contour length increased (see Figure 4.5B) and monotonically decreased as the spacing between fragments increased (see Figure 4.5C). Overall, model results were consistent with behavioural and neurophysiological properties of V1 neurons observed by Li et al. [2006]. In contrast, the feedforward control did not show similar neurophysiological trends.

Through an extensive sensitivity analysis and experimenting with model permutations, it was found that both recurrence as well as the biologically inspired architecture of the model were necessary to reliably reproduce results.

Strict constraints imposed on the model allowed learnt lateral connections to be directly observed and compared with lateral connections of V1 neurons. Recently, other researchers have highlighted a need for such a detailed analysis of learnt lateral connections in similar R-CNN models [Spoerer et al., 2020].

It was found that learnt lateral connections were non-uniformly distributed and spread out more densely in the direction of the preferred orientation of their source edge extraction neurons. Excitatory connections showed a higher degree of anisotropy and spread out further than inhibitory ones (see Figure 4.9). The results were in contrast with the predefined lateral connectivity structures used in existing models of contour integration. Most models assume a similar spread for inhibitory and excitatory connections. Moreover, the model's inhibitory connections, although less directed than excitatory connections, also spread out most densely in the direction of the preferred orientation of source V1 neurons. Most existing contour integration models assume inhibitory connections spread out more densely in the orthogonal-to-the-preferred direction of source V1 neurons. Overall, learnt connectivity patterns were consistent with the updated Association Field Model proposed by Field et al. [2013].

Next, the model was used to explore how contour integration may benefit the processing of natural images. In natural images, edge pixels have variable strengths and some contour parts are harder to detect compared to others. When the model was trained on the task of edge detection in natural images, it learnt to enhance responses to weaker contours, as previously suggested by Piëch et al. [2013] and Li [1998] (see Figure 5.2). However, this did not translate to an increase in overall behavioural performance. At least on the dataset used for training, feedforward responses appear to provide sufficient information to detect contours, and the model did not need to recurrently integrate contours in order to detect them.

To further investigate the role of contour integration in natural images, a new task where edge extraction outputs are less helpful was introduced. The task involved tracing contours between two markers to determine if they are connected via a smooth contour. The trained model learned a robust strategy that translated well to stimuli that were slightly different from training data. This was not the case for the feedforward control which saw a dramatic drop in performance, even though it had a higher task-level performance over the training dataset. Moreover, the model's responses varied in the same direction as in the task with synthetic fragmented contours; task-level performance as well as contour integration gains of component neurons dropped monotonically as the spacing between fragments increased (see Figure 5.6). In contrast, the behavioural responses of the control were unaffected by fragment spacing, and its component neurons showed lower drops in gains as the spacing between neurons was increased.

The work presented here should be of interest to neuroscientists as it offers a way of quantitatively analyzing the role of low-level neural phenomena in high-level behavioural tasks. Although I investigated the role of contour integration on edge detection and contour tracing in natural images only, the model can easily be incorporated in larger ANNs and used on many other tasks, including object classification and segmentation. Moreover, the ability to quantify performance on high-level tasks can be used to compare alternative mechanistic models of neuron phenomena. Furthermore, the approach of learning parameters using DNNs can challenge assumptions used in predetermined computational models. Finally, trained models provide full access to their internals and can be used to simultaneously analyze multiple properties of contour integration more easily than can be done in the brain [Olshausen and Field, 2006].

This work should also be of interest to the machine learning community. Recently, it has been highlighted that ANNs are prone to excessively relying on the simplest discriminative feature they can find in training data [Funke et al., 2021, Geirhos et al., 2020]. The performance of these networks falls dramatically when tested with inputs that are from outside the distribution of training data [Hendrycks and Dietterich, 2019]. Often even the smallest changes in the data can cause these collapses. It is important to add additional constraints on models or remove these features from the data [Geirhos et al., 2018] to guide the model to learn desired solutions over shortcut ones. In complex problems, it can be difficult to identify and separate out these less reliable features. Adding architectural constraints, as done here, is one way of providing these restrictions. By doing so, I have shown that ANNs can learn more resilient features that generalize well to out-of-distribution data, at least for the task of tracing contours in natural images.

6.2 Future work

The work presented here can be expanded in several different directions. In the following section I highlight a few interesting directions.

Alternative mechanisms of contour integration

The model's contour integration mechanism only involved lateral connections in shallow layers of a visual systems. However, the V1 cortex also contains dense feedback connections from multiple higher layers. An alternative theory postulates that the brain mostly uses these feedback connections for contour integration (see Section 2.3.4). Particularly interesting are the neurophysiological properties observed by Chen et al. [2014]. Here, they propose that contour integration is mediated by a recurrent loop between the V1 and V4 cortices. Mechanistically, there are not many models of contour integration that utilize feedback as their main conduit for contour integration. It would be interesting to develop such a model and quantitatively compare it with this model. The work present here provides a framework for carrying out such a comparison.

Moreover, the brain most likely uses both mechanisms for contour integration. A model that includes both can be used to elucidate how the two mechanisms interact on different high-level visual tasks. Such a model could also be used to analyze other properties of contour integration, such as selective object boundaries enhancement and what role contour integration plays in the perception of non-luminance defined contours (see Section 2.2.3). Finally, V4 is strongly interconnected with the MT cortex of the dorsal stream [Markov et al., 2014]. It contains many neurons that are sensitive to motion defined shapes [Handa and Mikami, 2018]. Potentially, such a model can be used to analyze the interplay between V4 detected kinetic contours and luminance defined contours detected by V1.

Positive-only constraint

The strict positive-only constraint imposed on lateral connection weights of the model severely impacted training time and task-level performance. The method of clipping negative weights during training interfered with the training process. When the constraint was removed, in the Relaxed Positive Constraint Model (RPCM) (Section 4.6.1) variant, the model trained faster and achieved the highest task-level performance on multiple tasks. However, the constraint conforms the model with Dale's principle [Dale, 1935, Eccles et al., 1954] and makes it more biologically plausible. Moreover, restricting excitatory and inhibitory lateral kernels to be positive only allowed them to be directly compared with connections in the brain. Finally, it was found that the model was the only network that was reliably consistent with the brain at the behavioral, neurophysiological and neuroanatomical levels. It would interesting to explore other methods of incorporating this constraint that have a less dramatic affect on training time and performance.

Preliminary analysis of the RPCM model's dynamic equations show that at steady state and with certain reasonable parameter values, both lateral excitatory and inhibitory connections affect neuronal responses in the same way. Hence, negative connections in the excitatory connection matrix can be moved to the inhibitory connection matrix and visa versa without affecting performance. Alternatively, Tripp and Eliasmith [2016], Parisien et al. [2008] present a method that can convert any idealized network (with units that form positive and negative connections to other units) into another more physiologically plausible (with units that only form positive connections) one that is nearly functionally equivalent. In this new network, source units project to target units with positive-only connections, as well as to a small population of inhibitory units that also form positive-only connections with target units. This method can potentially be used to convert the better performing RPCM network into a more biologically plausible one. Finally, Cornford et al. [2021] recently proposed an initialization method and weight update mechanism for inhibitory neurons in a sign constraint network that can help Dale's principle compliant ANNs reach the same performance as regular ANNs.

Alternative learning rules

The Back Propagation Through Time learning algorithm was used to train the model. A major limitation of this algorithm is it's large memory requirement; activations of every recurrent step need to be retained in memory to calculate gradients and update network parameters. This limits the number of recurrent steps that can be modelled. Recently, Linsley et al. [2020a] presented a new learning algorithm, the Contractor Recurrent Back Propagation algorithm that has the same memory requirements regardless of the number of modelled recurrent steps. Figure 4.11A shows that increasing the number of recurrent steps improved performance. It would be interesting to see if incorporating this learning rule can similarly improve the performance of the proposed model. Moreover, it may potentially improve model stability by allowing the underling differential equations to be modelled up to their steady states.

Neurophysiological properties of contour integration in natural images

The work presented here hypothesizes several behavioural and neurophysiological trends of contour integration in natural images. As far as I am aware, similar behavioural and neurophysiological results for contour integration in natural images do not exist. It would be interesting to conduct similar experiments neurophysiologically on V1 neurons and check the validity of these predictions.

More shape reliant visual ANN models

Geirhos et al. [2018] found that CNN models trained on ImageNet show a strong bias towards texture over shape when recognizing objects. Humans, on the other hand, rely mostly on object edges [Landau et al., 1988]. They constructed a new dataset, Stylized ImageNet, where texture cues were made inconsistent with object identities. Specifically, content (high-level activations) from natural images was separated and mixed in with style (low-level activations) from many art paintings. These were then used to generate new stylized images.

When networks were jointly trained on the two datasets, they learnt to rely on object edges more than textures compared to networks trained only on ImageNet. These networks also demonstrated higher performance on the tasks of object classification and detection. Since contour integration is a mechanism to enhance object edges, it would be interesting to see if incorporating the contour integration model into these networks could lead models to learn more shape based features directly from natural images. It would also be interesting to see if this would further improve performance on these tasks.

Robustness to common ANN pitfalls

Finally, neurophysiological analyses of the model indicated that the model learns to solve tasks using different strategies compared to feedforward models. The strategies employed by the model showed several similarities with contour integration in the brain. It was found that the strategies learnt by the model generalized better to stimuli outside the training distributions. It would be interesting to see if the mechanism employed by the model also lead to better robustness against adversarial images [Goodfellow et al., 2014, Madry et al., 2017] and other common image perturbations [Hendrycks and Dietterich, 2019]. Using an older version of the model, I showed how contour integration offered greater resilience from One (few) Pixel Adversarial Attacks, [Khan et al., 2018]. It would be interesting to repeat the analysis using the updated model proposed here and to expand to other more severe adversarial attacks.

References

Thomas D Albright and Gene R Stoner. Contextual influences on visual processing. *Annual review of neuroscience*, 25(1):339–379, 2002.

Alessandra Angelucci, Jonathan B Levitt, Emma JS Walton, Jean-Michel Hupe, Jean Bullier, and Jennifer S Lund. Circuits for local and global signal integration in primary visual cortex. *Journal of Neuroscience*, 22(19):8633–8646, 2002.

Alessandra Angelucci, Maryam Bijanzadeh, Lauri Nurminen, Frederick Federer, Sam Merlin, and Paul C Bressloff. Circuits and mechanisms for surround modulation in visual cortex. *Annual review of neuroscience*, 40:425–451, 2017.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Nicholas Baker, Hongjing Lu, Gennady Erlikhman, and Philip J Kellman. Deep convolutional networks do not classify based on global object shape. *PLoS computational biology*, 14(12):e1006613, 2018.

David GT Barrett, Ari S Morcos, and Jakob H Macke. Analyzing biological and artificial neural networks: challenges with opportunities for synergy? *Current opinion in neurobiology*, 55:55–64, 2019.

Ohad Ben-Shahar and Steven Zucker. Geometrical computations explain projection patterns of long-range horizontal connections in visual cortex. *Neural computation*, 16(3): 445–476, 2004.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Maryam Bijanzadeh, Lauri Nurminen, Sam Merlin, Andrew M Clark, and Alessandra Angelucci. Distinct laminar processing of local and global context in primate primary visual cortex. *Neuron*, 100(1):259–274, 2018.

William H Bosking, Ying Zhang, Brett Schofield, and David Fitzpatrick. Orientation selectivity and the arrangement of horizontal connections in tree shrew striate cortex. *Journal of neuroscience*, 17(6):2112–2127, 1997.

Wieland Brendel and Matthias Bethge. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. *arXiv preprint arXiv:1904.00760*, 2019.

Andrew Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. *arXiv preprint arXiv:2102.06171*, 2021.

Cameron Buckner. The comparative psychology of artificial intelligences, May 2019. URL http://philsci-archive.pitt.edu/16128/.

Charles F Cadieu, Ha Hong, Daniel LK Yamins, Nicolas Pinto, Diego Ardila, Ethan A Solomon, Najib J Majaj, and James J DiCarlo. Deep neural networks rival the representation of primate IT cortex for core visual object recognition. *PLoS Comput Biol*, 10 (12):e1003963, 2014.

Edward M Callaway. Local circuits in primary visual cortex of the macaque monkey. *Annual review of neuroscience*, 21(1):47–74, 1998.

Edward M Callaway. Feedforward, feedback and inhibitory connections in primate visual cortex. *Neural Networks*, 17(5-6):625–632, 2004.

Matteo Carandini and David J Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62, 2012.

Matteo Carandini, David J Heeger, and J Anthony Movshon. Linearity and normalization in simple cells of the macaque primary visual cortex. *Journal of Neuroscience*, 17(21): 8621–8644, 1997.

Leo M Chalupa and John S Werner. *The visual neurosciences, Vols. 1 & 2.* MIT press, 2004.

Minggui Chen, Yin Yan, Xiajing Gong, Charles D Gilbert, Hualou Liang, and Wu Li. Incremental integration of global contours through interplay between visual cortical areas. *Neuron*, 82(3):682–694, 2014.

Rujia Chen, Feng Wang, Hualou Liang, and Wu Li. Synergistic processing of visual contours across cortical layers in V1 and V2. *Neuron*, 96(6):1388–1402, 2017.

Heather J Chisum, François Mooser, and David Fitzpatrick. Emergent properties of layer 2/3 neurons reflect the collinear arrangement of horizontal connections in tree shrew visual cortex. *Journal of Neuroscience*, 23(7):2947–2960, 2003.

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.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Radoslaw M Cichy and Daniel Kaiser. Deep neural networks as scientific models. *Trends in cognitive sciences*, 23(4):305–317, 2019.

Tim Cooijmans, Nicolas Ballas, César Laurent, Çağlar Gülçehre, and Aaron Courville. Recurrent batch normalization. *arXiv preprint arXiv:1603.09025*, 2016.

Jonathan Cornford, Damjan Kalajdzievski, Marco Leite, Amélie Lamarquette, Dimitri Michael Kullmann, and Blake Aaron Richards. Learning to live with dale's principle: ANNs with separate excitatory and inhibitory units. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=eU776ZYxEpz.

Rui Ponte Costa, Yannis M Assael, Brendan Shillingford, Nando de Freitas, and Tim P Vogels. Cortical microcircuits as gated-recurrent neural networks. *arXiv preprint arXiv:1711.02448*, 2017.

Henry Dale. Pharmacology and nerve-endings, 1935.

Aniruddha Das and Charles D Gilbert. Topography of contextual modulations mediated by short-range interactions in primary visual cortex. *Nature*, 399(6737):655–661, 1999.

Peter Dayan and Laurence F Abbott. *Theoretical neuroscience: computational and mathematical modeling of neural systems.* Computational Neuroscience Series, 2001.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009. URL http://www.image-net.org/.

James J DiCarlo, Davide Zoccolan, and Nicole C Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

John C Eccles, P Fatt, and K Koketsu. Cholinergic and inhibitory synapses in a pathway from motor-axon collaterals to motoneurones. *The Journal of physiology*, 126(3):524–562, 1954.

Reinhard Eckhorn, Roman Bauer, Wolfgang Jordan, Michael Brosch, Wolfgang Kruse, Matthias Munk, and HJ Reitboeck. Coherent oscillations: A mechanism of feature linking in the visual cortex? *Biological cybernetics*, 60(2):121–130, 1988.

Thomas Euler, Silke Haverkamp, Timm Schubert, and Tom Baden. Retinal bipolar cells: elementary building blocks of vision. *Nature Reviews Neuroscience*, 15(8):507–519, 2014.

Jerome Feldman. The neural binding problem(s). *Cognitive neurodynamics*, 7(1):1–11, 2013.

Daniel J Felleman and David C Van Essen. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral cortex (New York, NY: 1991)*, 1(1):1–47, 1991.

David Ferster and Kenneth D Miller. Neural mechanisms of orientation selectivity in the visual cortex. *Annual review of neuroscience*, 23(1):441–471, 2000.

D J Field, J R Golden, and Hayes A. Contour integration and the association field. In L M Chalupa and J S Werner, editors, *The new visual neurosciences*, pages 627–638. MIT Press, 2013.

David J Field, Anthony Hayes, and Robert F Hess. Contour integration by the human visual system: evidence for a local "association field". *Vision Research*, 33(2):173–193, 1993.

Jeremy Freeman and Eero P Simoncelli. Metamers of the ventral stream. *Nature Neuroscience*, 14(9):1195–1201, 2011.

Kunihiko Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural networks*, 1(2):119–130, 1988.

Christina M Funke, Judy Borowski, Karolina Stosio, Wieland Brendel, Thomas SA Wallis, and Matthias Bethge. Five points to check when comparing visual perception in humans and machines. *Journal of Vision*, 21(3):16–16, 2021.

Anupam K Garg, Peichao Li, Mohammad S Rashid, and Edward M Callaway. Color and orientation are jointly coded and spatially organized in primate primary visual cortex. *Science*, 364(6447):1275–1279, 2019.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

Robert Geirhos, Carlos R Medina Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. *arXiv preprint arXiv:1808.08750*, 2018.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Wilson S Geisler, Jeffrey S Perry, BJ Super, and DP Gallogly. Edge co-occurrence in natural images predicts contour grouping performance. *Vision Research*, 41(6):711–724, 2001.

Charles D Gilbert. Horizontal integration and cortical dynamics. *Neuron*, 9(1):1–13, 1992.

Charles D Gilbert and Wu Li. Top-down influences on visual processing. *Nature Reviews Neuroscience*, 14(5):350–363, 2013.

Charles D Gilbert and Torsten N Wiesel. Clustered intrinsic connections in cat visual cortex. *Journal of Neuroscience*, 3(5):1116–1133, 1983.

P Girard, JM Hupé, and J Bullier. Feedforward and feedback connections between areas V1 and V2 of the monkey have similar rapid conduction velocities. *Journal of neurophysiology*, 85(3):1328–1331, 2001.

Melvyn A Goodale, John Paul Meenan, Heinrich H Bülthoff, David A Nicolle, Kelly J Murphy, and Carolynn I Racicot. Separate neural pathways for the visual analysis of object shape in perception and prehension. *Current Biology*, 4(7):604–610, 1994.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Frédéric Gosselin and Philippe G Schyns. Bubbles: a technique to reveal the use of information in recognition tasks. *Vision Research*, 41(17):2261–2271, 2001.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

Alexander Grunewald and Evelyn K Skoumbourdis. The integration of multiple stimulus features by V1 neurons. *Journal of Neuroscience*, 24(41):9185–9194, 2004.

Umut Guclu and Marcel AJ van Gerven. Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream. *Journal of Neuroscience*, 35(27):10005–10014, 2015.

Takashi Handa and Akichika Mikami. Neuronal correlates of motion-defined shape perception in primate dorsal and ventral streams. *European Journal of Neuroscience*, 48(10): 3171–3185, 2018.

Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *2011 International Conference on Computer Vision*, pages 991–998. IEEE, 2011. URL http://home.bharathh.info/pubs/codes/SBD/download.html.

Jianzhong He, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. Bi-directional cascade network for perceptual edge detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3828–3837, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

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.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Robert F Hess, Keith A May, and Serge O Dumoulin. Contour integration: Psychophysical, neurophysiological, and computational perspectives. In *The Oxford Handbook of Perceptual Organization*, chapter 5. Oxford University Press, 2014.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

Judith A Hirsch and Charles D Gilbert. Synaptic physiology of horizontal connections in the cat's visual cortex. *Journal of Neuroscience*, 11(6):1800–1809, 1991.

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

Roos Houtkamp and Pieter R Roelfsema. Parallel and serial grouping of image elements in visual perception. *Journal of Experimental Psychology: Human Perception and Performance*, 36(6):1443, 2010.

Brian Hu and Ernst Niebur. A recurrent neural model for proto-object based contour integration and figure-ground segregation. *Journal of computational neuroscience*, 43 (3):227–242, 2017.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

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):106–154, 1962.

Chou P Hung, Gabriel Kreiman, Tomaso Poggio, and James J DiCarlo. Fast readout of object identity from macaque inferior temporal cortex. *Science*, 310(5749):863–866, 2005.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics*, 26(11):3365–3385, 2019.

Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven A. Siegelbaum, and A James Hudspeth. *Principles of neural science*. McGraw-hill New York, 5 edition, 2013.

Mitesh K Kapadia, Minami Ito, Charles D Gilbert, and Gerald Westheimer. Improvement in visual sensitivity by changes in local context: parallel studies in human observers and in V1 of alert monkeys. *Neuron*, 15(4):843–856, 1995.

Mitesh K Kapadia, Gerald Westheimer, and Charles D Gilbert. Spatial distribution of contextual interactions in primary visual cortex and in visual perception. *Journal of neurophysiology*, 84(4):2048–2062, 2000.

Kohitij Kar, Jonas Kubilius, Kailyn Schmidt, Elias B Issa, and James J DiCarlo. Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior. *Nature Neuroscience*, 22(6):974–983, 2019.

Aaron M Kerlin, Mark L Andermann, Vladimir K Berezovskii, and R Clay Reid. Broadly tuned response properties of diverse inhibitory neuron subtypes in mouse visual cortex. *Neuron*, 67(5):858–871, 2010.

Seyed-Mahdi Khaligh-Razavi and Nikolaus Kriegeskorte. Deep supervised, but not unsupervised, models may explain IT cortical representation. *PLoS computational biology*, 10(11):e1003915, 2014.

Salman Khan and Bryan Tripp. An empirical model of activity in macaque inferior temporal cortex. *Neural Networks*, 87:8–21, 2017.

Salman Khan, Alexander Wong, and Bryan Tripp. Guarding against adversarial attacks using biologically inspired contour integration. *Journal of Computational Vision and Imaging Systems*, 4(1):3–3, 2018.

Salman Khan, Alexander Wong, and Bryan P Tripp. Task-driven learning of contour integration responses in a V1 model. In *Shared Visual Representations in Humans & Machines NeurIPS Workshop*, 2020.

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

ZF Kisvarday and U Th Eysel. Cellular organization of reciprocal patchy networks in layer iii of cat visual cortex (area 17). *Neuroscience*, 46(2):275–286, 1992.

Zoltán F Kisvárday, E Toth, Martin Rausch, and Ulf T Eysel. Orientation-specific relationship between populations of excitatory and inhibitory lateral connections in the visual cortex of the cat. *Cerebral cortex (New York, NY: 1991)*, 7(7):605–618, 1997.

Ilona Kovacs and Bela Julesz. A closed curve is much more than an incomplete one: Effect of closure in figure-ground segmentation. *Proceedings of the National Academy of Sciences*, 90(16):7495–7497, 1993.

Dwight J Kravitz, Kadharbatcha S Saleem, Chris I Baker, Leslie G Ungerleider, and Mortimer Mishkin. The ventral visual pathway: an expanded neural framework for the processing of object quality. *Trends in cognitive sciences*, 17(1):26–49, 2013.

Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:4, 2008.

MF Kritzer, A Cowey, and P Somogyi. Patterns of inter- and intralaminar gabaergic connections distinguish striate (V1) and extrastriate (V2, V4) visual cortices and their functionally specialized subdivisions in the rhesus monkey. *Journal of Neuroscience*, 12 (11):4545–4564, 1992.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Norbert Kruger, Peter Janssen, Sinan Kalkan, Markus Lappe, Ales Leonardis, Justus Piater, Antonio J Rodriguez-Sanchez, and Laurenz Wiskott. Deep hierarchies in the primate visual cortex: What can we learn for computer vision? *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1847–1871, 2012.

Jonas Kubilius, Martin Schrimpf, Kohitij Kar, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Pouya Bashivan, Jonathan Prescott-Roy, Kailyn Schmidt, et al. Brain-like object recognition with high-performing shallow recurrent ANNs. *arXiv preprint arXiv:1909.06161*, 2019.

Yoshiyuki Kubota. Untangling gabaergic wiring in the cortical microcircuit. *Current opinion in neurobiology*, 26:7–14, 2014.

Ron Kupers and Maurice Ptito. Compensatory plasticity and cross-modal reorganization following early visual deprivation. *Neuroscience & Biobehavioral Reviews*, 41:36–52, 2014.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Victor AF Lamme and Pieter R Roelfsema. The distinct modes of vision offered by feedforward and recurrent processing. *Trends in neurosciences*, 23(11):571–579, 2000.

Barbara Landau, Linda B Smith, and Susan S Jones. The importance of shape in early lexical learning. *Cognitive development*, 3(3):299–321, 1988.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015.

Wu Li and Charles D Gilbert. Global contour saliency and local colinear interactions. *Journal of neurophysiology*, 88(5):2846–2856, 2002.

Wu Li, Valentin Piëch, and Charles D Gilbert. Contour saliency in primary visual cortex. *Neuron*, 50(6):951–962, 2006.

Wu Li, Valentin Piëch, and Charles D Gilbert. Learning to link visual contours. *Neuron*, 57(3):442–451, 2008.

Zhaoping Li. A neural model of contour integration in the primary visual cortex. *Neural computation*, 10(4):903–940, 1998.

Hualou Liang, Xiajing Gong, Minggui Chen, Yin Yan, Wu Li, and Charles D Gilbert. Interactions between feedback and lateral connections in the primary visual cortex. *Proceedings of the National Academy of Sciences*, 114(32):8637–8642, 2017.

Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3367–3375, 2015.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. URL http://https://cocodataset.org/#home.

Grace W Lindsay. Convolutional neural networks as a model of the visual system: Past, present, and future. *Journal of cognitive neuroscience*, pages 1–15, 2020.

Drew Linsley, Junkyung Kim, Vijay Veerabadran, Charles Windolf, and Thomas Serre. Learning long-range spatial dependencies with horizontal gated recurrent units. In *Advances in neural information processing systems*, pages 152–164, 2018.

Drew Linsley, Alekh Karkada Ashok, Lakshmi Narasimhan Govindarajan, Rex Liu, and Thomas Serre. Stable and expressive recurrent vision models. In *Advances in Neural Information Processing Systems*, volume 33, pages 10456–10467, 2020a. URL https://proceedings.neurips.cc/paper/2020/file/766d856ef1a6b02f93d894415e6bfa0e-Paper.pdf.

Drew Linsley, Junkyung Kim, Alekh Ashok, and Thomas Serre. Recurrent neural circuits for contour detection. In *International Conference on Learning Representations*, 2020b. URL https://openreview.net/forum?id=H1gB4RVKvB.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

Margaret Livingstone and David Hubel. Segregation of form, color, movement, and depth: anatomy, physiology, and perception. *Science*, 240(4853):740–749, 1988.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

Jennifer S Lund and Charles Q Wu. Local circuit neurons of macaque monkey striate cortex: IV. neurons of laminae 1-3a. *Journal of Comparative Neurology*, 384(1):109–126, 1997.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Rafael Malach, Y Amir, M Harel, and Amiram Grinvald. Relationship between intrinsic connections and functional architecture revealed by optical imaging and in vivo targeted biocytin injections in primate striate cortex. *Proceedings of the National Academy of Sciences*, 90(22):10469–10473, 1993.

Adam H Marblestone, Greg Wayne, and Konrad P Kording. Toward an integration of deep learning and neuroscience. *Frontiers in computational neuroscience*, 10:94, 2016.

Nikola T Markov, Maria M Ercsey-Ravasz, AR Ribeiro Gomes, Camille Lamy, Loic Magrou, Julien Vezoli, Pierre Misery, Arnaud Falchier, Rene Quilodran, Marie-Alice Gariel, et al. A weighted and directed interareal connectivity matrix for macaque cerebral cortex. *Cerebral cortex*, 24(1):17–36, 2014.

David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001. URL https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/.

Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International conference on artificial neural networks*, pages 52–59. Springer, 2011.

John HR Maunsell, Geoffrey M Ghose, John A Assad, Carrie J Mcadams, Christen Elizabeth Boudreau, and Brett D Noerager. Visual response latencies of magnocellular and parvocellular lgn neurons in macaque monkeys. *Visual neuroscience*, 16(1):1–14, 1999.

Barbara A McGuire, Charles D Gilbert, Patricia K Rivlin, and Torsten N Wiesel. Targets of horizontal connections in macaque primary visual cortex. *Journal of Comparative Neurology*, 305(3):370–392, 1991.

Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113, 2014.

David A Mély, Junkyung Kim, Mason McGill, Yuliang Guo, and Thomas Serre. A systematic comparison between visual cues for boundary detection. *Vision Research*, 120: 93–107, 2016.

David A Mély, Drew Linsley, and Thomas Serre. Complementary surrounds explain diverse contextual phenomena across visual modalities. *Psychological review*, 125(5):769, 2018.

Sun Minni, Li Ji-An, Theodore Moskovitz, Grace Lindsay, Kenneth Miller, Mario Dipoppa, and Guangyu Robert Yang. Understanding the functional and structural differences across excitatory and inhibitory neurons. *bioRxiv*, 2019. URL `https://www.biorxiv.org/content/early/2019/06/25/680439`.

Mortimer Mishkin, Leslie G Ungerleider, and Kathleen A Macko. Object vision and spatial vision: two cortical pathways. *Trends in neurosciences*, 6:414–417, 1983.

Vernon B Mountcastle, Philip W Davies, and Alvin L Berman. Response properties of neurons of cat's somatic sensory cortex to peripheral stimuli. *Journal of neurophysiology*, 20(4):374–407, 1957.

Javier R Movellan. Tutorial on gabor filters. [Online], 2002. URL `https://inc.ucsd.edu/mplab/tutorials/gabor.pdf`.

Aran Nayebi, Daniel Bear, Jonas Kubilius, Kohitij Kar, Surya Ganguli, David Sussillo, James J DiCarlo, and Daniel LK Yamins. Task-driven convolutional recurrent models of the visual system. *arXiv preprint arXiv:1807.00053*, 2018.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL `http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf`.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

Bruno A Olshausen and David J Field. Contour integration and the association field. In J Leo Van Hemmen and Terrence J Sejnowski, editors, *23 Problems in Systems Neuroscience*, chapter 10, pages 182–211. Oxford University Press, 2006.

Guy A Orban. Higher order visual processing in macaque extrastriate cortex. *Physiological reviews*, 2008.

Brian O'Toole and Peter Wenderoth. The tilt illusion: Repulsion and attraction effects in the oblique meridian. *Vision Research*, 17(3):367–374, 1977.

Christopher Parisien, Charles H Anderson, and Chris Eliasmith. Solving the problem of negative synaptic weights in cortical models. *Neural computation*, 20(6):1473–1494, 2008.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

Valentin Piëch, Wu Li, George N Reeke, and Charles D Gilbert. Network model of top-down influences on local gain and contextual interactions in visual cortex. *Proceedings of the National Academy of Sciences*, 110(43):E4108–E4117, 2013.

Panayiota Poirazi, Terrence Brannon, and Bartlett W Mel. Arithmetic of subthreshold synaptic summation in a model ca1 pyramidal cell. *Neuron*, 37(6):977–987, 2003.

Xavier Soria Poma, Edgar Riba, and Angel Sappa. Dense extreme inception network: Towards a robust CNN model for edge detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1923–1932, 2020.

Karim Rajaei, Yalda Mohsenzadeh, Reza Ebrahimpour, and Seyed-Mahdi Khaligh-Razavi. Beyond core object recognition: Recurrent processes account for object recognition under occlusion. *PLoS computational biology*, 15(5):e1007001, 2019.

Rishi Rajalingham, Elias B Issa, Pouya Bashivan, Kohitij Kar, Kailyn Schmidt, and James J DiCarlo. Large-scale, high-resolution comparison of the core visual object recognition behavior of humans, monkeys, and state-of-the-art deep artificial neural networks. *Journal of Neuroscience*, 38(33):7255–7269, 2018.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature Neuroscience*, 22(11):1761–1770, 2019.

Kathleen S Rockland and Jennifer S Lund. Intrinsic laminar lattice connections in primate visual cortex. *Journal of Comparative Neurology*, 216(3):303–318, 1983.

Anna W Roe, Leonardo Chelazzi, Charles E Connor, Bevil R Conway, Ichiro Fujita, Jack L Gallant, Haidong Lu, and Wim Vanduffel. Toward a unified theory of visual area V4. *Neuron*, 74(1):12–29, 2012.

Pieter R Roelfsema. Cortical algorithms for perceptual grouping. *Annu. Rev. Neurosci.*, 29:203–227, 2006.

Pieter R Roelfsema, Victor AF Lamme, and Henk Spekreijse. Object-based attention in the primary visual cortex of the macaque monkey. *Nature*, 395(6700):376–381, 1998.

Ankit Rohatgi. Webplotdigitizer: Version 4.4, 2020. URL https://automeris.io/WebPlotDigitizer.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Nicole C Rust and James J DiCarlo. Selectivity and tolerance ("invariance") both increase as visual information propagates from cortical area V4 to IT. *Journal of Neuroscience*, 30(39):12978–12995, 2010.

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Mark M Schira, Alex R Wade, and Christopher W Tyler. Two-dimensional mapping of the central and parafoveal visual field to human visual cortex. *Journal of neurophysiology*, 97(6):4284–4295, 2007.

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.

Thomas Serre. Deep learning: the good, the bad, and the ugly. *Annual review of vision science*, 5:399–426, 2019.

Wei Shen, Xinggang Wang, Yan Wang, Xiang Bai, and Zhijiang Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3982–3991, 2015.

S Shushruth, Lauri Nurminen, Maryam Bijanzadeh, Jennifer M Ichida, Simo Vanni, and Alessandra Angelucci. Different orientation tuning of near-and far-surround suppression in macaque primary visual cortex mirrors their tuning in human perception. *Journal of Neuroscience*, 33(1):106–119, 2013.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Lawrence C Sincich and Gary G Blasdel. Oriented axon projections in primary visual cortex of the monkey. *Journal of Neuroscience*, 21(12):4416–4426, 2001.

Fabian H Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S Tolias. Engineering a less artificial intelligence. *Neuron*, 103(6):967–979, 2019.

David C Somers, Emanuel V Todorov, Athanassios G Siapas, Louis J Toth, Dae-Shik Kim, and Mriganka Sur. A local circuit approach to understanding integration of long-range inputs in primary visual cortex. *Cerebral cortex (New York, NY: 1991)*, 8(3):204–217, 1998.

Courtney J Spoerer, Patrick McClure, and Nikolaus Kriegeskorte. Recurrent convolutional neural networks: a better model of biological object recognition. *Frontiers in psychology*, 8:1551, 2017.

Courtney J Spoerer, Tim C Kietzmann, Johannes Mehrer, Ian Charest, and Nikolaus Kriegeskorte. Recurrent neural networks can explain flexible trading of speed and accuracy in biological vision. *PLoS computational biology*, 16(10):e1008215, 2020.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Dan D Stettler, Aniruddha Das, Jean Bennett, and Charles D Gilbert. Lateral connectivity and contextual interactions in macaque primary visual cortex. *Neuron*, 36(4):739–750, 2002.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Corentin Tallec and Yann Ollivier. Can recurrent neural networks warp time? In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJcKhk-Ab.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

Hiroki Tanaka and Izumi Ohzawa. Surround suppression of V1 neurons mediates orientation-based representation of high-order visual features. *Journal of neurophysiology*, 101(3):1444–1462, 2009.

Anne Treisman. The binding problem. *Current opinion in neurobiology*, 6(2):171–178, 1996.

Bryan Tripp. Approximating the architecture of visual cortex in a convolutional network. *Neural computation*, 31(8):1551–1591, 2019.

Bryan Tripp and Chris Eliasmith. Function approximation in inhibitory networks. *Neural Networks*, 77:95–106, 2016.

Bryan P Tripp. Similarities and differences between stimulus tuning in the inferotemporal visual cortex and convolutional networks. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 3551–3560. IEEE, 2017.

Mauro Ursino and Giuseppe Emiliano La Cara. A model of contextual interactions and contour detection in primary visual cortex. *Neural Networks*, 17(5-6):719–735, 2004a.

Mauro Ursino and Giuseppe-Emiliano La Cara. Comparison of different models of orientation selectivity based on distinct intracortical inhibition rules. *Vision Research*, 44(14): 1641–1658, 2004b.

Ruben S van Bergen and Nikolaus Kriegeskorte. Going in circles is the way forward: the role of recurrence in visual inference. *Current Opinion in Neurobiology*, 65:176–193, 2020.

Simo Vanni, Henri Hokkanen, Francesca Werner, and Alessandra Angelucci. Anatomy and physiology of macaque visual cortical areas V1, V2, and V5/MT: bases for biologically realistic models. *Cerebral Cortex*, 30(6):3483–3517, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

Aaron Voelker, Ivana Kajić, and Chris Eliasmith. Legendre memory units: Continuous-time representation in recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/file/952285b9b7e7a1be5aa7849f32ffff05-Paper.pdf.

Brian A Wandell, Serge O Dumoulin, and Alyssa A Brewer. Visual field maps in human cortex. *Neuron*, 56(2):366–383, 2007.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

John Simon Werner and Leo M Chalupa. *The new visual neurosciences*. MIT Press, 2014.

Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

Daniel LK Yamins and James J DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, 19(3):356–365, 2016.

Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624, 2014.

Raymond Yeh, Chen Chen, Teck Yian Lim, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2(3), 2016.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pages 2528–2535. IEEE, 2010.

Li Zhaoping. Border ownership from intracortical interactions in visual area v2. *Neuron*, 47(1):143–153, 2005.

Hong Zhou, Howard S Friedman, and Rüdiger Von Der Heydt. Coding of border ownership in monkey visual cortex. *Journal of Neuroscience*, 20(17):6594–6611, 2000.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

APPENDICES

Appendix A

Contour Integration Layer Derivation

In the current-based-subtractive-inhibition model of Piëch et al. [2013], the temporal dynamics of a single orientation column are defined as,

$$\frac{dx}{dt} = \frac{1}{\tau_x}\left[-x + J_{xx}f_x(x) - J_{xy}f_y(y) + I_{0e} + I + \sum_{x' \in eCRF} L_{xx'}f_x(x')\right], \tag{A.1}$$

$$\frac{dy}{dt} = \frac{1}{\tau_y}\left[-y + J_{yx}f_x(x) + I_{0i} + \sum_{x' \in eCRF} L_{yx'}f_x(x^{'})\right]. \tag{A.2}$$

Here, x and y are the membrane potential of E and I nodes, J_{xx}, J_{xy}, J_{yx} are E$\rightarrow$E (self), I$\rightarrow$E and E$\rightarrow$I within node-pair connection strengths, respectively, $f_{.}(.)$ is a non-linear activation function that transforms membrane potentials into firing rates, $\tau_{.}$ are membrane time constants, $I_{0.}$ are background currents, I is the external input current to the model, and $L_{xx'}$ and $L_{yx'}$ are lateral connections strengths from E nodes $x^{'}$ in neighboring orientations columns that lie within the extra classical receptive fields (e-cRF) of the target orientation column.

Each spatial location contains multiple orientation columns with similar selectivities but with different orientation preferences. The lateral connections of an E-I node pair are modelled as fixed association field structures. They are different for each orientation column; excitatory connections are directed towards the column's preferred orientation while inhibitory connections are directed orthogonal to it. The same set of orientation columns is repeated at different spatial locations. Lateral connection patterns are shared across different spatial locations. The full model consists of a 2D grid of spatial locations with multiple columns at each location. The dynamics of the full model are realized as the joint activities of all modelled orientation columns.

To incorporate the model into neural networks, first, summations over orientation columns in the e-cRF are replaced with sliding convolutions. The convolutional kernel operates over orientation columns at the same spatial location as well as orientation columns at neighboring locations that lie within the e-cRF. It also operates over the target orientation column and incorporates the excitatory self connection J_{xx}. The convolutional operation also allows modelling of multiple E-I node pairs simultaneously.

The dynamics of a single orientation column can then be expressed as,

$$\frac{dx}{dt} = \frac{1}{\tau_x}\left[-x - J_{xy}f_y(y) + I_{0e} + I + (W_e \circledast f_x(X)\right], \tag{A.3}$$

$$\frac{dy}{dt} = \frac{1}{\tau_y}\left[-y + J_{yx}f_x(x) + I_{0i} + (W_i \circledast f_x(X))\right], \tag{A.4}$$

where, $f_x(X)$ is the output activation of all modeled excitatory neurons.

Using Euler's Method, the approximate solution is given by the following difference equations,

$$x_t = x_{t-1} + \frac{\delta t}{\tau_x}\left[-x_{t-1} - J_{xy}f_y(y_{t-1}) + I_{0e} + I + (W_e \circledast f_x(X_{t-1}))\right], \tag{A.5}$$

$$y_t = y_{t-1} + \frac{\delta t}{\tau_y}\left[-y_{t-1} + J_{yx}f_x(x_{t-1}) + I_{0i} + (W_i \circledast f_x(X_{t-1}))\right]. \tag{A.6}$$

Setting ${}^{\delta t}/_{\tau_x} = a$, ${}^{\delta t}/_{\tau_y} = b$ and collecting all like terms together,

$$x_t = (1-a)x_{t-1} + a\left[-J_{xy}f_y(y_{t-1}) + I_{0e} + I + (W_e \circledast f_x(X_{t-1}))\right], \tag{A.7}$$

$$y_t = (1-b)y_{t-1} + b\left[J_{yx}f_{x_t}(x_t) + I_{0i} + (W_i \circledast f_x(X_t))\right], \tag{A.8}$$

where $x_0 = y_0 = 0$.

This final form resembles a leaky vanilla recurrent neural network [Tallec and Ollivier, 2018] which can be trained using standard neural network training techniques.

In the original model, fixed positive values were used for time constants, local and lateral connection weights. Rather than arbitrarily setting these values, I included them in the parameters of the model and let the neural network learn their optimal values. However, to ensure connections operate in a manner consistent with the original model, positive-only constraints were added to lateral connections of the model. For time constants and local connections strengths (J_{xy}, J_{yx}), a sigmoid non-linearity was added to keep their influences on membrane potentials positive. For lateral connection weights, all negative weights were set to zero after every parameter update. This more strict constraint allowed learnt lateral weight structures to be directly comparable to neurophysiologically observed connections.

With these additional constraints, the activity of an orientation column is expressed as,

$$x_t = (1 - \sigma(a))x_{t-1} + \sigma(a)\left[-\sigma(J_{xy})f_y(y_{t-1}) + I_{0e} + I + W_e \circledast f_x(X_{t-1})\right], \tag{A.9}$$

$$y_t = (1 - \sigma(b))y_{t-1} + \sigma(b)\left[\sigma(J_{yx})f_x(x_t) + I_{0i} + W_i \circledast f_x(X_t)\right], \tag{A.10}$$

where $x_0 = y_0 = 0$.