MODELING INTEGRATED CORTICAL LEARNING: EXPLORATIONS OF CORTICAL MAP DEVELOPMENT, UNIT SELECTIVITY, AND OBJECT RECOGNITION

by

James W. Ryland

APPROVED BY SUPERVISORY COMMITTEE:

Richard M. Golden, Chair

Alice O'Toole

Daniel Krawczyk

Peter Assmann

Thank you very much to my family and partner for keeping me sane during my COVID dissertation, y'all mean the world to me.

MODELING INTEGRATED CORTICAL LEARNING: EXPLORATIONS OF CORTICAL MAP DEVELOPMENT, UNIT SELECTIVITY, AND OBJECT RECOGNITION

by

JAMES W. RYLAND, BS, MS

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN

COGNITITION AND NEUROSCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

December 2021

ACKNOWLEDGMENTS

My progress through the PhD program and in writing this dissertation has been helped by many people. Here I want to highlight a few of them.

First I would like to thank my advisor Richard M. Golden and my co-advisor Alice J. O'Toole. I found their advice and guidance invaluable for shaping and completing my PhD. You taught me to balance my dreams of theoretical advances and new architectures with the demands of the real world and the academic setting.

I would like to thank my colleagues in Richard's COINS lab and Alice's Face Lab. You have let me springboard many ideas off of you, and have been very patient with my ramblings. Special thanks to Gerie, Connor, Matt, and Ivette for taking a look at drafts of my articles and helping me to edit them!

This research was supported in part by a grant from Raytheon Technologies. Special thanks to Phil Sallee and the folks at Raytheon who made this possible.

Finally, I want to thank my partner Talyer Dinh for keeping me motivated during COVID and my dissertation, and my mom and sister for letting us stay in Portland during the last leg of this journey and the beginning of a new one.

September 2021

MODELING INTEGRATED CORTICAL LEARNING: EXPLORATIONS OF CORTICAL MAP DEVELOPMENT, UNIT SELECTIVITY, AND OBJECT RECOGNITION

James W. Ryland, PhD
The University of Texas at Dallas, 2021

Supervising Professor: Richard M. Golden, Chair

One of the most formative theories in neuroscience is the *Hierarchical Theory of Cortex* (HTC), which postulates a hierarchy of simple and complex cells within each cortical visual area. The *Deep Convolutional Neural Networks* (DCNN) architecture is the most computationally successful implementations of HTC, and has been adopted as a tool for linking cognition to neural processes. However, DCNNs are exceedingly abstract models of cortical learning. First, DCNNs use fixed connectivity, whereas cortical connectivity is plastic. Second, DCNNs use convolutional weight-sharing, whereas simple cells in visual cortex learn using local competition rules. Third, DCNNs use fixed pools, whereas complex cells in visual cortex may learn their pooling structure. This means that DCNNs do not develop an analogue to the cortical maps developed by cortex. In addition, differences in feature learning may mean that DCNNs learn very different high-level unit representations compared to the high-level visual cortex. In this dissertation, I introduce a biologically inspired framework for understanding unsupervised visual category learning, called the *Temporal Relation Manifold* TRM framework, which extends the object manifold framework of vision. With this new framework, I develop a model of hierarchical cortical learning that integrates biologically plausible models of axon development, simple cell learning, and complex cell learning,

into a single model called the *Integrated Cortical Learning Model* (ICL). As part of these efforts I also introduce novel methods for incorporating axonal learning and development into artificial neural networks called the Axon Game and the Arbor Layer. I examined the utility of this new cortical model in three main sets of simulation studies. First, I explored its ability to develop high-level cortical maps organized by semantic categories. Second, I explored whether the ICL model would develop functionally specialized unit representations or unspecialized unit representations. Third, I tested the performance of several versions of the model on two image recognition benchmarks (Fashion-MNIST & ImageNette). These simulation studies showed three main results. First, that the ICL model developed continuous topological maps in its upper layers, but these maps were not substantially different from the maps developed in its lower layers in key ways. Second, the ICL model developed unspecialized unit representations similar to those of DCNNs, though this result may be due to propagation of shallow representations. Third, the ICL model performed at a comparable level to similarly-sized DCNNs with very modest tuning (91% accuracy for Fashion-MNIST, and 40% accuracy for ImageNette). Post-hoc analyses suggested that the proposed complex cell model may have been a limiting factor, highlighting an area for future study. The deep ICL model built for this dissertation showed the novel ability to learn hierarchical cortical maps, agreement with DCNN work on unit-level representations, and promising performance, all while using more biologically motivated unsupervised learning rules. In summary, this dissertation introduces a framework (TRM) and bio-inspired model (ICL) as an alternative to DCNNs, and evaluates this new model in terms of cortical map development, unit representations, and classification performance.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Deep Convolutional Neural Networks or (DCNNs) have demonstrated that with a relatively small number of mechanisms, some of which are directly inspired by cortex, models can perform quite well at visual object recognition and classification (Fukushima, 1980; Krizhevsky et al., 2012). Although today's DCNNs are not usually explicitly considered models of visual cortex, their computational success and the fact that they have a similar structure of layers and local connections, has prompted researchers to compare the representations learned by DCNNs and to those learned by high-level ventral visual cortex (Khaligh-Razavi and Kriegeskorte, 2014; Rajalingham et al., 2018; Yamins and DiCarlo, 2016b). Despite their atheoretical usage, the computational success and neural predictive power of DCNNs have made them the de facto model of choice for understanding computation in the ventral stream at both a neural and cognitive level. However, there are critical issues with treating DCNNs as the default model of the ventral stream that need to be addressed.

First Issue: DCNNs Rely on Implausible Architectural and Learning Assumptions. Most computationally successful DCNN models use a strong form of supervised learning and require a massive number of images with high-level semantical labels to train effectively (Yamins and DiCarlo, 2016b). In contrast, humans learn using extensive amounts of unstructured and unsupervised experience, without the need for huge amounts of externally applied labels (Yamins and DiCarlo, 2016b). Many unsupervised DCNN learning algorithms have actually been proposed, but the representations learned by these models are largely inconsistent with our knowledge of neural representation in higher-level visual cortex, and exhibit poor classification performance (Khaligh-Razavi and Kriegeskorte, 2014; Yamins and DiCarlo, 2016b).

The learning issues with DCNNs go beyond reliance on supervised learning. In both supervised and unsupervised learning in DCNNs, the network computes an error signal in

its final layer and propagates this signal back through all of its layers. The results of this error propagation are used to modify the connection weights within each layer. This learning method is called *backpropagation*. Although there are ongoing efforts to find neurally plausible implementations of this scheme (Scellier and Bengio, 2017), backpropagation requires the close coordination of many neural layers that are not directly linked. Even prominent researchers using these techniques acknowledge that it is unlikely that cortex implements a form of backpropagation like the one used with most DCNNs (Yamins and DiCarlo, 2016b).

Further, DCNNs share information in a largely implausible fashion. Each neuron in a DCNN layer's feature map shares its weights with all of the other neurons in that feature map. This mechanism is called *convolutional weight-sharing*. Neuroscience suggests that convolutional weight-sharing is almost certainly implausible (Yamins and DiCarlo, 2016b). Researchers usually tolerate weight-sharing as a computationally efficient stand-in for more local and neurally plausible learning mechanisms that will learn similar features under the right conditions.

Finally, DCNNs use a highly artificial handcrafted fixed connectivity structure between their layers, usually based on the retinotopy of the early visual cortex. Generally, neuroscience suggests that the connectivity between layers in cortex develops and adapts in highly reactive ways due to many forces (Benson et al., 2001). Presumably because of these forces and others, the connectivity and features of the visual cortical hierarchy become progressively less retinotopic in higher layers. Given this, fixed convolutional connectivity may poorly model the structure of higher-level vision on a fundamental level.

It is important from a neuroscience standpoint to develop bio-inspired algorithms with good performance, while avoiding biologically implausible mechanisms like supervised learning and weight sharing (Yamins and DiCarlo, 2016b). These biologically implausible mechanisms may be interpreted as highly abstract biological models, but the degree of abstraction used limits their ability to make specific qualitative predictions. Models with more biologically motivated mechanisms would allow for more specific qualitative predictions, as their

mechanisms would have a more direct neural interpretation. Further, testing more bio-plausible models on image recognition tasks can better help us understand if the proposed mechanisms are computationally powerful enough to address real world visual tasks.

In this dissertation, I introduce a special class of neurally plausible learning architectures which are referred to as neurally conservative. Neurally conservative means the core mechanisms of a model are commonly theorized as plausible or well-accepted from a neuroscience standpoint. For illustration, the learning mechanisms used to train both supervised and unsupervised DCNNs would not generally be considered neurally conservative, as their core mechanisms (convolutional-weight-sharing and fixed pooling) are not supported by neuroscience, and can only be viewed as modeling cortical learning in a highly abstract way. Although DCNNs are not neurally conservative as cortical models, they make specific qualitative predictions. The stronger claims of a more neurally conservative model provide opportunities for making a richer set of qualitative predictions. Towards this end, this dissertation explores the behavioral and functional predictions of a more neurally conservative model and whether this model can achieve strong object recognition performance similar to DCNNs.

Second Issue: DCNN Units may be Selective yet not Specialized. Recent research into DCNNs has shown that they appear to learn functionally unspecialized unit representations in their upper levels even though they demonstrate apparent selectivity using conventional neuroscience measurement methodologies (Parde et al., 2021; Szegedy et al., 2014a). Here, an unspecialized unit representation means that each individual unit would be used in a broad array of category discriminations rather than a few. This contrasts with the long-held view that high-level ventral stream neurons are quite specialized (Grill-Spector and Weiner, 2014). Parde et al. (2021) goes further and suggests that the apparent selectivity of high-level ventral visual neurons may be a methodological artifact given that functional specialization is not required to achieve a high-level of discriminative performance. These

clashing views have given rise to an emerging debate about the level of specialization of high-level ventral visual neurons. However, the idea that functional neural specialization in high-level ventral visual cortex does not exist, and that selectivity could be a mere measurement artifact should be viewed skeptically as work in the field of neuroscience suggests that the extreme version of this hypothesis is almost certainly wrong given that injuries to category selective regions of ventral cortex (such as FFA) lead to specific and reproducible behavioral deficits (Farah, 2004). Further, DCNNs and most other deep learning models lack many well-established biological mechanisms that would promote both selectivity and functionally specialized unit representations in the ventral stream. Likely, the debate around specialized vs. unspecialized representations will be one of degree and whether specialization plays a major role in how the ventral stream functions.

Third Issue: Cortical Maps are not Learned in DCNNs. The responses properties of cortical neurons tend to vary smoothly across the cortical sheet in structures called cortical maps (Bednar and Wilson, 2016). Here "vary smoothly" means that cortical neurons in close physical proximity respond similarly. Researchers consider this aspect of cortical representation important because cortical maps have long been used as a window into understanding the structure of cortical feature spaces (Bednar and Wilson, 2016). DCNNs arrange their neurons in a completely arbitrary physical arrangement that is not influenced by the feature space learned by the model. Because of their arbitrary physical arrangement, researchers cannot directly compare the representations of DCNNs to the cortical map representations found in primate brains. Given the goal of developing computational models of primate visual cortical development, the absence of cortical maps in DCNNs is a serious limitation.

While there are many models for the cortical maps of low-level vision, there is a lack of high-level cortical map models that also demonstrate high performance on behavioral measures. Contemporary DCNNs show high classification performances, but their lack of cortical maps means that they are very limited in making physiological predictions about the

location of neural representation. Given this, there is a need for an alternative to DCNNs that can generate biologically realistic cortical mapping behavior.

1.1 Aims and Research Questions

The overall goal of this study is to develop a neurally conservative computational model of visual cortical learning, as an alternative to DCNNs and to test whether or not it addresses these issues via simulation. Towards this end, a new framework called the Temporal Relation Manifold (TRM) framework and computational mechanism called Temporal Auto-Untangling are introduced for understanding unsupervised category learning in visual cortex. Also, a specific model that implements this theory, called the Integrated Cortical Learning (ICL) model, is introduced and evaluated in a series of simulation studies. These simulation studies in this dissertation are designed to address the three questions listed below.

Question 1: Do the mechanisms proposed in the new ICL model produce high-level cortical maps that are organized by abstract category-like semantic features?

To answer this question, I tested if the model developed physical groupings or regions of neurons that were selective for the same high-level category on its cortical sheet. Further, I tested if these category-selective regions could have been generated due to chance.

Question 2: Do the mechanisms proposed in the new ICL model learn more specialized high-level units than would be predicted by DCNNs?

To answer this question, I used a procedure adapted from Parde et al. (2021) in which units were deleted from the higher levels of DCNNs in order to better understand the functional significance of individual units. Networks that are dependent on specific neurons to encode specific categories (i.e. specialized) will degrade quickly when units are deleted, whereas networks that don't depend on specific units to encode specific categories (i.e. unspecialized) will tend to degrade more gracefully. For this investigation I developed a novel

way to compare a network's actual deletion curve to the expected deletion curve it would have if it had a perfectly unspecialized representation. This comparison allowed for a test to determine if a network's representation was fully unspecialized or not.

Question 3: Can the neurally conservative mechanisms proposed in the ICL model perform visual category learning at a high level without supervision or deep backpropagation?

To answer this question, I trained train the neurally conservative ICL model on multiple image recognition benchmark datasets commonly used to train supervised and unsupervised DCNN models and test if it could perform object classification at an acceptable level. This was done entirely without the use of labels given to the network, or any form of deep error propagation within the model.

1.2 Impact

First, the development of the TRM framework gives direction and guidance for building more biologically inspired models of unsupervised visual learning, such as the ICL model. But it can also be used to generate any number of new models, whether biologically focused or more abstract, as it provides a rich set of theoretical ideas and constructs with which to view unsupervised visual learning in general.

Second, although models of cortical map behavior are commonly used to understand low-level vision, very little modeling work has been done to understand the cortical maps of high-level vision. If successful, the ICL model developed in this project could open the door to predicting the behavior of high-level cortical maps and understanding their functional relevance.

Third, very few models that feature high performance can also make predictions about the relationship between high-level unit selectivity unit specialization besides DCNNs cur-

rently. If successful, the ICL model could give a more biologically relevant counterpoint to predictions made with DCNNs.

Fourth, if successful, the ICL model developed in this project would be the first non-DCNN architecture to demonstrate strong unsupervised learning on real-world datasets. Further, it would demonstrate that more biologically inspired models can achieve high levels of performance using far more neurally conservative learning mechanisms than DCNNs. This would have scientific and practical consequences, as strong unsupervised learning has long been a goal for both computational neuroscience and computation vision.

1.3 Dissertation Overview

The rest of the dissertation is organized in the following manner:

Chapter 2 consists of an unpublished manuscript draft that introduces a new conceptual framework for supporting the development and evaluation of a new class of theories of higher-level visual object recognition in the primate visual cortex. This new conceptual framework is called the Temporal Relation Manifold (TRM) framework and it is designed to explicitly incorporate temporal environment statistics as powerful cues for unsupervised visual category learning using biologically plausible mechanisms. The TRM framework is an extension of the DiCarlo and Cox (DiCarlo and Cox, 2007) object manifold framework. Although the DiCarlo and Cox (2007) object manifold framework provides important insights into how the visual hierarchy can identify objects by performing a series of successive transformations, it does not provide an explicit proposal regarding how "temporal statistics" might support learning these transformations in an unsupervised manner. *Temporal statistics* refers to the general tendencies of how an environment changes in time, such as how visual features and views of objects tend to evolve in time (i.e. a view of cat tends to be followed by another view of a cat, shifted rotated, scaled, or with a changed pose). In theory, these temporal statistics could be a rich source of information about object and category identities. The TRM framework seeks

to extend the object manifold framework with the use of temporal statistics. Within the TRM framework, I developed a theory for unsupervised category learning called *Windowed-TAU*, which can be implemented within more biologically plausible models of cortex.

Chapter 3 is a version of a second published manuscript that provides a new explicit proposal of how known biological mechanisms that guide axonal development and plasticity may be used to support computationally powerful unsupervised connectivity learning in computational models of primate visual cortex. Although Chapter 3 provides an essential in-depth examination of a particular aspect of the ICL model, the discussion in Chapter 3 is not restricted to either the ICL model or the more general TRM framework. Rather the focus in Chapter 3 is primarily on the possible biological and computational consequences of these known mechanisms of axonal development and plasticity under specific implementations.

Chapter 4 introduces a specific biologically plausible model for implementing Windowed - TAU theory within the TRM framework from Chapter 2. I call this model the Integrated Cortical Learning (ICL) model because it integrates axonal, simple cell, and complex cell learning with biologically motivated implementations. Further, the axonal model introduced here called the arbor model is heavily inspired by concepts from the axon game model introduced in Chapter 3. This chapter also gives a more in-depth description of the different ICL models that were explored in the dissertation. The different variations of the ICL model are referenced heavily for disambiguation in the following chapters.

Chapter 5 presents a series of simulation studies designed to develop and evaluate the ICL model introduced in Chapter 4. Simulation Study 1 examines whether the cortical maps in the ICL model tend to organize semantically related features together physically, similar to the cortical maps of the high-level visual cortex. Simulation Study 2 examines whether the cortical units in the ICL model become as specialized as dominant theories of primate visual cortex suggest. Experiment 3 examines if the ICL models can perform unsupervised category learning on real-world image classification tasks using widely accepted biological

mechanisms. Simulation Study 1 showed that the ICL model did in fact learn topologically contiguous cortical maps for category selectivity, but that there were caveats for interpreting the finding. Simulation Study 2 showed that the ICL model actually learned surprisingly unspecialized neural representations similar to DCNNs. Finally, Simulation Study 3 showed that the unsupervised ICL model is already competitive with simple supervised DCNNs on some image recognition tasks, but that it is also quite far from state-of-the-art. Code for running versions of these simulation studies can be found here `https://bitbucket.org/jryland/icl-dissertation/`.

Chapter 6 provides a brief summary and evaluation of the contributions of this dissertation as well as concluding remarks and suggestions regarding future research. One area of future research proposed, focuses on improving the complex cell models used in with the ICL architecture, which appeared to be a limiting factor in the simulation studies.

CHAPTER 2

A FRAMEWORK, THEORY, & MODEL OF CORTICAL LEARNING

2.1 Preface

Chapter 2 is part of a separate manuscript. Chapter 2 includes the development of a new framework and computational theory of cortical learning that extends DiCarlo and Cox (2007) to account for how temporal variation of the environment could be used to support unsupervised category learning compatible to their Object Manifold Framework. The framework is called *Temporal Relation Manifolds* (TRM) and the computational theory is called Windowed Temporal Auto-Untangling (Windowed-TAU). The development and explanation of this new framework and theory are important contributions of this dissertation.

Further, this chapter is relevant to how the ICL model (which attempts to implement the new framework and the new theory) will address all three of the critical issues laid out in the aims section of the paper: unsupervised learning, the unit specialization debate, and the development of cortical maps.

Relevance to Cortical Map Formation. While not explicitly required by the TRM framework or the Windowed-TAU theory, the more efficient and neurally plausible implementations of these concepts will be composed of different types of Self-Organizing Map (SOM) models (e.g. Kohonen (1982)). SOM models are commonly used to make explicit predictions about the structure of actual cortical maps (Bednar and Wilson, 2016). In theory, Windowed-TAU suggests a new type of multi-layer cortical map learning that may be successful at developing high-level cortical maps and successful unsupervised category learning. This new type of multi-layer SOM learning is central to the ICL model's conception.

Relevance to Unit Specialization. Several core mechanisms of TRM-based approaches should induce specialized neural representations. Since ICL seeks to implement the ideas from Windowed-TAU and TRM, I expected that it would produce more specialized unit representations than would be expected from DCNNs. But, I found this was not the

case with the current implementation of ICL, however this result may be due to limitations with the complex cell model used.

Relevance to Unsupervised Learning. TRM framework and Windowed-TAU theory offer an explanation for how a model like ICL can learn category structure hierarchically across layers without labels. First, Temporal Relation Manifold (TRM) framework, which extends the DiCarlo and Cox (2007) Object Manifold framework to more explicitly incorporate the use of temporal statistics in an environment to spontaneously separate categories. In other words this framework explicitly uses the way objects and visual features change over time in order to define and learn the identity of objects and the nature of categories. Second, Windowed Temporal Auto-Untangling (Windowed-TAU) specifies a set of computational behaviors that satisfy the TRM framework's requirements and that can be implemented using commonly theorized and accepted cortical mechanisms. Finally, this chapter introduces a simplified model for illustration of the TRM and Windowed-TAU concepts.

2.2 Abstract

In this paper, we extended the DiCarlo and Cox (2007) tangled object manifold framework of object recognition to better address the unsupervised nature of category learning. We developed a novel Markov chain-based similarity metric that formally connects aspects of manifold untangling with trace learning. Using these developments, we replaced unobservable labels and artificial category boundaries with our observable Markov chain walk-based similarity metric as a theoretically grounded target for unsupervised category untangling. Further, we developed a new rationale for how neuronal input windows should be chosen for an untangling algorithm using this new framework. This new framework for manifold untangling and trace learning allowed us to synthesize aspects of simple cell learning, complex cell learning, and axonal development theories, into a high-level theory of how the visual cortex learns to separate object categories at a computational level. These ideas are relevant to

the development of more powerful unsupervised machine learning methods and the general study of the ventral visual pathway.

2.3 Introduction

DiCarlo and Cox (2007) popularized the concept of *Tangled Object manifolds*, which refers to the idea that the viewing state of a visual object traces a high-dimensional manifold in image space, tangling it with other object manifolds. Under this framework, the primary goal of object recognition is to untangle these object manifolds from one another. This idea has become a touchstone in computer vision, machine learning, and visual neuroscience communities. Despite its popularity, the object manifold framework has changed relatively little since its introduction. In this paper, we will integrate the concept of object manifolds with trace theory to produce a new framework for understanding object recognition. Using this new framework, we will synthesize several theories of cortical learning into a novel computational theory of object recognition.

2.4 Re-Defining the Problem

DiCarlo and Cox (2007) provided a now classic way to understand the problem of object recognition. DiCarlo and Cox (2007) envisioned the visual environment as a series of discrete object categories combined with viewing parameters that produce a visual image. Under this framework, an object at a particular location away from the viewer represents the state of the environment. The environment state maps to a point in a sensory space that is experienced as a visual image. All the possible ways an object or category of objects can vary in the state space project to a tangled high-dimensional sheet of points in the sensory space. These tangled sheets of images are called *Object Manifolds*.

According to DiCarlo and Cox (2007), the central problem of object recognition is that these object category manifolds are tangled with one another in sensory space and no simple

decision boundary will separate them. They suggest the goal of object recognition is to learn a function that projects these manifolds into a new space where they are linearly separable or untangled from one another DiCarlo and Cox (2007). This viewpoint can be easily applied to the current state of the art supervised deep-learning with visual neural network models with observable labels (Krizhevsky et al., 2012; Yamins and DiCarlo, 2016b). However, (DiCarlo and Cox, 2007) are somewhat vague about how concepts from object manifold framework could be applied to unsupervised category learning, though the paper did suggest this possibility.

Many authors, including DiCarlo and Cox, have proposed that the temporal statistics of the environment could be used instead of labels to learn an appropriate transformation from the sensory representation space to support object recognition, but none have integrated these ideas with object manifolds in detail (DiCarlo and Cox, 2007; Einhäuser et al., 2005; Földiák, 1991). In this paper, we will extend the object manifold framework by replacing the unobservable categories and labels as the organizing force in state space with an observable metric of nearness that fits with classic concepts from unsupervised temporal learning such as trace theory. To do this we will develop a new kind of object called a "temporal relation manifold" that combines object manifolds with temporal information.

2.5 Temporal Relation Manifolds

In a philosophical sense, we can define the identity of an event in the universe (a cluster of particles in a specific location and configuration) by the fact that similar events proceed and follow it. To borrow from a well-trodden metaphor, the only reason we say a ship is "Theseus's Ship" is that there was a ship there yesterday called Theseus's Ship that looked very similar. This view is reductive, but useful in a practical sense, as it suggests that object identity and category are derivatives of the temporal nature of a world that has no true boundaries between events.

Picture a simplified model of the visual environment where an observer sees one object category at a time with variable viewing parameters (see Fig 2.1). This environment could be thought of as a simple Markov chain where each unique combination of object category and viewing condition is a state x in the environment state space Ω. For now, we will assume that the state-space is organized topologically by category and viewing conditions as it is in the object manifold framework, but this will change later. The observer cannot directly experience the environment state x. Instead, the observer experiences the environment state through a sensory space such as a retinal image space that severely tangles the environment state space Ω (see Fig. 2.1). More precisely, there is a function $\mathbf{z}$ that projects the environment state x into a real-valued high-dimensional sensory space $\mathcal{Z}$, $\mathbf{z} : \Omega \rightarrow \mathcal{Z}$, in a roughly continuous way. As a complication, there may be noise on the dimensions of the sensory space and nuisance axes of the environment state-space that chaotically perturb the projection from state space to sensory space, making the projection function stochastic $\tilde{\mathbf{z}}$ from a practical perspective.

Right now, this perspective is only a minor modification of the object manifold framework, moving from projecting separate object state spaces into sensory space to projecting one unitary environment state space into sensory space with additional temporal assumptions. We are about to extend the concept of an object manifold much further.

In the object manifold framework, the goal of the observer was to learn a projection from $\mathcal{Z}$ to a new space that linearly separates the object manifolds. In our new conceptualization, the observer will try to learn a projection from the sensory space into a new space where each axis reacts to a region of the state space. We will call this a *region space* $\mathcal{R}$. A region space is composed of a set of overlapping region functions, where a region function $r_i(\mathbf{z})$ takes a point in $\mathcal{Z}$ and estimates whether that a sensory pattern originated from a region of the state space Ω around some point x_i (see Fig. 2.1). An ideal region function $r_i(\mathbf{z}(x))$ is a bell-shaped function on the state space Ω with a range between 0 and some max value.

Further, the level sets of a region function are convex regions on Ω. Fig. 2.1 shows what an approximation of a region function might look in state space.

Because a set of ideal region functions will tend to group semantically related sensory patterns, the region space will promote linear separability between meaningful categories (see Fig. 2.1). In biology, these region functions would be implemented by a cascading series of cortical areas that build towards high-level neurons whose selectivity resembles these region functions.

To learn an ideal set of region functions, an observer will need a target for learning that tells it what regions should look like. This target can be knowledge of the topology of the state space or a measure of nearness that can be directly observed. Up until now, we assumed that the environment state space has an innate idealized semantic organization, but this organization is not directly observable in an unsupervised learning paradigm. Instead, we will be using the fact that the environment generates short sequences, which tend to be of the same or related objects, to help us define a new notion of the local organization on the environment state space Ω that is observable (see Fig. 2.2). In theory, this will reproduce many desirable qualities of the unobservable ideal semantic organization we used previously.

Before defining a nearness metric on Ω we need to introduce some terms and constructs. First, a sequence of states generated by a Markov chain (like our environment) is often called a random walk or just a walk. Second, the probability of transitioning between states in a Markov chain is usually denoted with the matrix $\mathbf{P}$, where P_{ij} is the probability of transitioning from state x_i to x_j. Further, the probability of transitioning between states over multiple steps is denoted with $\mathbf{P}^\tau$, where P^τ_{ij} is the likelihood of transition from x_i to x_j in τ steps.

We define nearness between states x_i and x_j in Ω as the likelihood that they will occur near each other in time. More precisely, we will say there is a similarity metric on Ω, called $M(x_i, x_j)$ (see Eq. 1 and Fig. 2.2), that defines the similarity between two states in Ω as

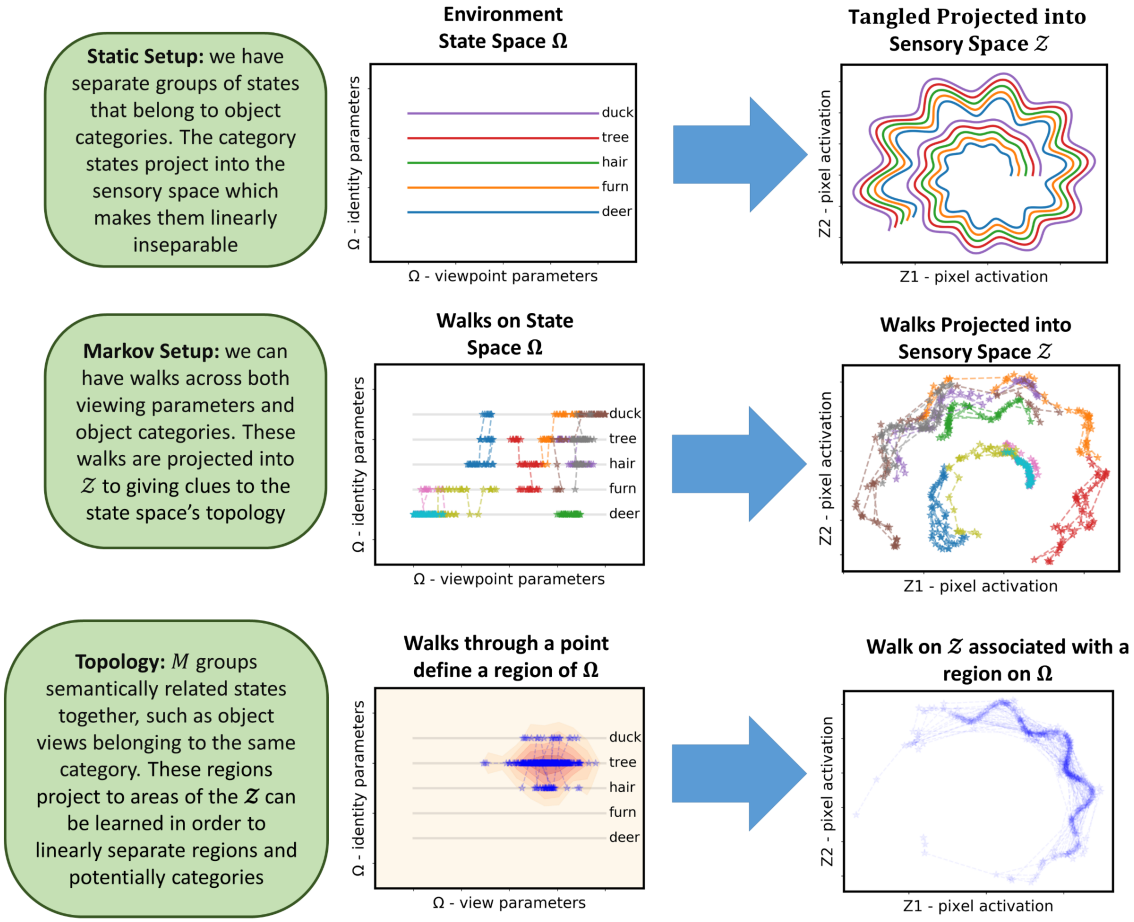

Figure 2.1. Projections from State Space to Sensory Space

This figure shows how the static and dynamic aspects of our state space are constructed for our simplified object recognition problem. (TOP) Shows the state space of separate object categories varied across viewing parameters and how it is projected into a tangled sensory space which makes the categories linearly inseparable. (MIDDLE) Shows how the environment state transitions over time moving across viewing parameters most frequently, but sometimes across category. (BOTTOM) The nearness between points for our states will be defined as the likelihood that a walk will cross between them in a small approximate span of time. This can be used to define contiguous and convex regions of the state space via a metric function called M. Further, topology and regions given by M can be observed via walks in the sensory space. Note that projecting walks associated with a region in state space tends to highlight points in $\mathcal{Z}$ associated with a category.

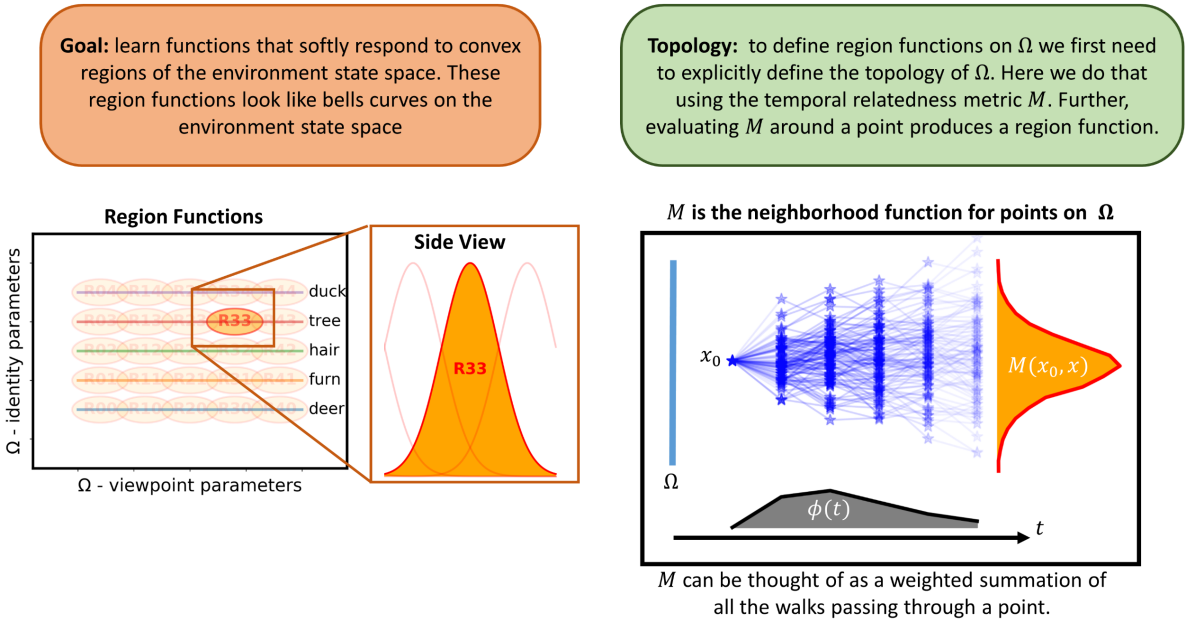

Figure 2.2. Defining Region Functions and the Temporal Relation Metric
This figure gives an overview of what the region functions would look like, and how they can be calculated using an integral over all possible walks. (LEFT) The goal of our newly re-defined problem is to find a series of overlapping region functions on the state-space. Here a region function is a function on the sensory space that react softly to a convex region of state space. Note that the region functions tend to be densely packed on states corresponding to each category. Using a simple linear method each category could be discriminated using only the values of the region functions. (RIGHT) These region functions are defined as the likelihood of crossing between a central point and all other points in an approximate span of time, see Eq. 1 and Eq. 2. This can also loosely be thought of as an integral over all the vertices of all possible paths leaving or coming to a point weighted by the likelihood of that path and by the temporal weighting function ϕ.

the likelihood of a random walk going between x_i and x_j in approximately Ω steps, given that the walk starts at either x_i or x_j. In Eq. 1 the temporal weighting Ω function weights the probability of crossing from x_i to x_j in σ steps highly but gives progressively less weight to the probability of crossings in fewer or greater than Ω steps. The specific choice for ϕ is not important, but ϕ needs to converge to 0 as τ becomes large to avoid a known issue, namely that walk expectations tend to be dominated by longer walks, which may mix to the stationary distribution under common conditions (Von Luxburg et al., 2014). When this happens a metric no longer relates to the concept of similarity in any meaningful way. A convenient choice for ϕ used for demonstrations in this paper is the gamma density function reparametrized to have its peak at σ.

$$\mathrm{M}(x_i, x_j) \sum_{\tau}^{\infty} \left(P_{ij}^{\tau} + P_{ji}^{\tau} \right) \tag{2.1}$$

$$r_i((z)) \approx f(M(x_i, x)) \tag{2.2}$$

Note: f is some strictly positive monotonically increasing function.

This new metric gives a concept of nearness on our state space that can directly guide region learning in several ways. Eq 2.2 shows how M can be used to directly define what a region function r_i centered on x_i should look like. However, we are only using the similarity metric M as a tool for defining the topology of the environment state space in an observable way. But, by simply defining the topology of Ω, the metric M also defines what constitutes a convex region on Ω, the learning of which is the main goal in our framework.

This new similarity metric can apply to points in a sensory space $M(z_i, z_j)$. If two points in sensory space appear similar according to the metric M, they may not look at all similar according to a Euclidean or angular distance metric. Therefore, convex regions defined by M

will look quite different than regions defined using Euclidean or angular metrics in sensory space (see Fig. 2.1).

Similarity and distance metrics defined using walk expectations on Markov chains have been used for finding missing links between items and deleting spurious links in graph-like datasets, like social networks, word relations, and paper citations (Hashimoto et al., 2015). However, using a walk-based similarity metric to theoretically ground the topology of the state space for object views and categories is novel to our knowledge.

Let us recap our new conceptualization of the problem of object recognition (see Fig. 2.3). First, we have an environment state-space called Ω with states like x_i. Further, this state space is organized topologically by a nearness or neighborhood metric called M which relates how close two states are $M(x_i, x_j)$. The state-space $\mathcal{X}$ projects into a sensory space $\mathcal{Z}$ that makes it difficult to tell which part of the state space generated an observation in $\mathcal{Z}$, such as $\mathbf{z}(x_0)$. The goal will be to find a transformation from sensory space into a region space $\mathcal{R}$ that allows us to approximate the region of state space that generates each observation in sensory space using only linear decision boundaries.

These new definitions make the relationship between temporal statistics and manifolds more explicit than was stated in (DiCarlo and Cox, 2007). Under our extension of the object manifold framework, the sensory space embeds not just the state-space, but also its temporal statistics via its topology. Further, we are now defining the environment state space's topology using the similarity metric $M(x_i, x_j)$, rather than idealized semantic axes. We will refer to this topologically organized state space embedded in a sensory space as a *temporal relation manifold*.

The temporal relatedness metric M allows us to bridge the concept of regions on the temporal relation manifold, with the target of "trace learning". Trace rules attempt to teach a layer of cells to become less sensitive to change in their inputs over time, and hopefully category and view variability, by making their target for learning a temporal average of past

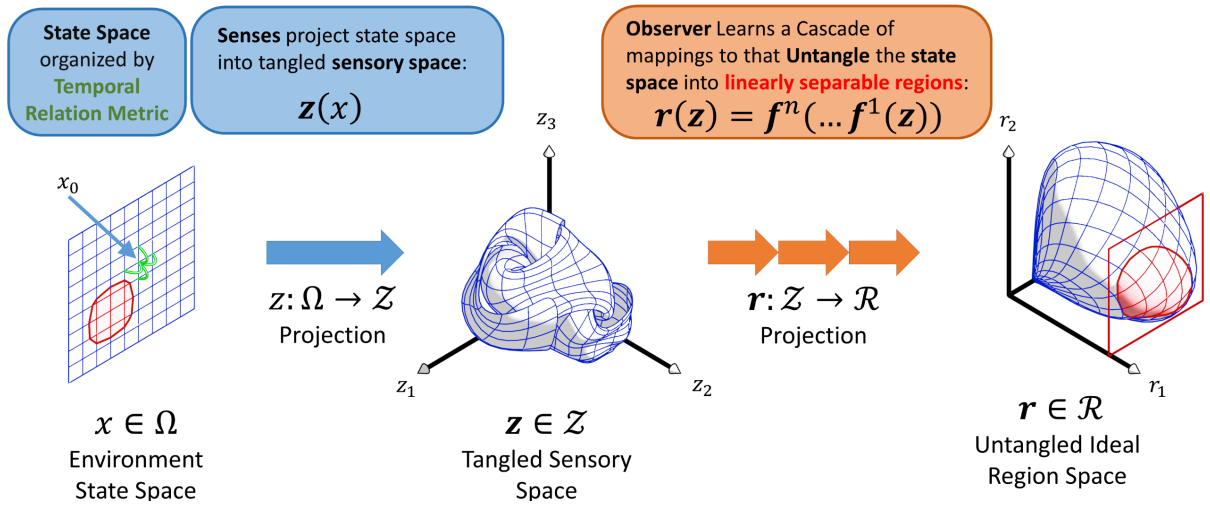

Figure 2.3. The Goal of Untangling the Temporal Relation Manifold
This figure gives an overview how the various spaces project to one another and what they do. Reading from left to right, we start out with the state space Ω that projects into the tangled sensory space $\mathcal{Z}$ through the function $\mathbf{z}(x)$. The goal of our hypothetical observer is to learn another projection function $\mathbf{r}(\mathbf{z})$ that untangles the environment state space into linearly separable regions in what we call region space $\mathcal{R}$. This transformation from sensory space to region space would be implemented as a cascade of neural layers. Importantly, the concept of nearness between states is defined by the temporal relation metric.

activations (Földiák, 1991; Wallis, 1996). In practice, this means a layer of cells will broaden their receptive fields so that each cell will respond to events that are happening and events that are likely to happen soon in the same manner. This should sound familiar because the receptive fields learned by cells using a trace rule will model the temporal similarity metric M on a sensory space. In effect when a trace rule pushes a cell to learn to respond invariantly to a temporally related group of inputs, it is attempting to learn a region of a temporal relation manifold defined by a specific M and ϕ function.

Our goal is still to find a projection that differentiates single stimuli or snapshots in time from one another in a useful fashion (i.e. we are still just doing category learning, not sequence learning), despite introducing temporal statistics.

Later we will discuss how temporal relation manifolds can lead to a new view of how simple and complex cells untangle categories from one another (called Temporal Auto-Untangling), but first, we need to discuss how the real world prevents us from learning untangling functions directly on the sensory space and how to address these issues.

2.5.1 Folding on Window Spaces

If a sensory space has few dimensions (i.e. few pixels or receptors), and the temporal relation manifold in sensory space is not too convoluted, then learning a projection to a region space is somewhat trivial. For human vision, this is rarely the case, as our senses embed the temporal relation manifold in a high dimensional space in a convoluted way, which makes learning a direct untangling function intractable. For this reason, many techniques such as convolutional networks (Krizhevsky et al., 2012), instead untangle on small groupings of the original sensory dimensions or windows making the task locally tractable and then stack these solutions in hierarchical layers. However, these window spaces "fold" the manifold. We introduce the term folding here to mean that many stimuli have become indistinguishable in a window space because of the loss of dimensionality compared to the full sensory space (see Fig. 2.4).

Because of the folding problem, a windowed untangling algorithm must integrate information across windows, usually via layering. But first, the untangling algorithm needs to choose good windows, as some window choices will fold the temporal relation manifold too severely, while others will make portions of the manifold fold in a useful way. A good window structure can make an algorithm more efficient at untangling the temporal relation manifold by requiring less representation and fewer training examples.

2.5.2 Choosing Appropriate Windows

It turns out that sparsely activating neurons that are highly predictive of one another make better windows than sparsely activating neurons that are independent of one another. By

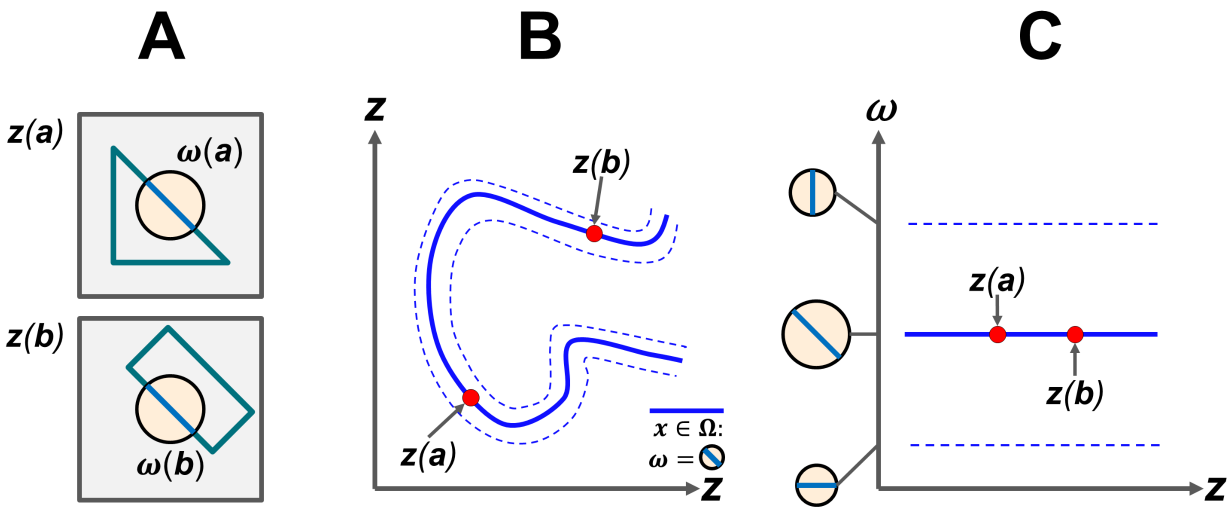

Figure 2.4. Folding (Dimesnionality Loss) on Windows
This figure illustrates the general concept of folding, whereby when a full sensory space is projected into a window space, many stimuli that were differentiable in the full space become inseparable in the window space. (A) Shows two sensory images **z**(**a**) and **z**(**b**) of the objects **a** and **b** **(shown in blue/green)**, while also highlighting the window image on dimensions ω. (B) Shows how in the full space **z** the objects **a** and **b** are easily separable despite tangling. (C) Shows how the window dimensions ω fold the sensory space **z** such that objects **a** and **b** are indistinguishable. Once stimuli have been folded together in the window space nothing can separate them except information from other windows.

"sparse activity", we mean that a given neuronal unit will rarely be active. We will call groups of sparsely activating units that are mutually predictive s*parse predictive windows*, and we will call groups of sparsely activating units that are independent *sparse independent windows*. As sparse units are the default, we will often omit the term when describing windows.

To better understand why window predictiveness influences untangling it is useful to use a geometric view (see Fig. 2.5). The neurons of sparse independent windows rarely activate at the same time, and for the majority of the time, none are active. This implies that most of the sensory patterns cluster near the origin or the axes of the window's activation space. These regions compress many patterns to the same location in window space, as they have

a low dimensionality. As such, these regions of the window's activation space severely fold the temporal relation manifold. We will label these highly folded regions *origin folds* and *axis folds* respectively (see Fig. 2.5).

In a sparse predictive window, there will be a tendency for many neurons within the window to fire at various levels of activation. This implies that a subset of sensory patterns will spread out into the middle range of the window space. Because this middle region of window space has a much higher dimensionality than the origin or the axis fold (roughly the same dimensionality as the window itself), fewer patterns will project to the same location. As such, the temporal relation manifold is far less folded there than in an origin or axis folds. We will call this region of a predictive window space that comparatively expands parts of the temporal relation manifold the *expanded fold* (see Fig. 2.5).

Because the expanded fold only represents parts of the full manifold and in a less folded manner, it is generally more computationally tractable to ignore everything outside of the expanded fold, when performing local untangling on a window. We will call this ignored region the *null response region*.

When the right units are chosen to be part of a window, sparse predictive windows and folding synergize to increase the power of untangling algorithms. To see how this happens, we need to talk about how the temporal relation manifold is convoluted in the sensory space. Real temporal relation manifolds twist in ways that are often self-similar, meaning many parts of the manifold have the same shape along certain dimensions (see Fig. 2.6). This self-similarity comes about because objects share features that themselves transform in similar ways across sensory dimensions. Window spaces will often compress these self-similar portions of the temporal relation manifold together, creating what we term an *apparent manifold*. An untangling function on this apparent manifold untangles multiple self-similar portions of the manifold in parallel (see Fig. 2.6), while the folding can be undone by integrating information from other windows later. This parallelized untangling operation

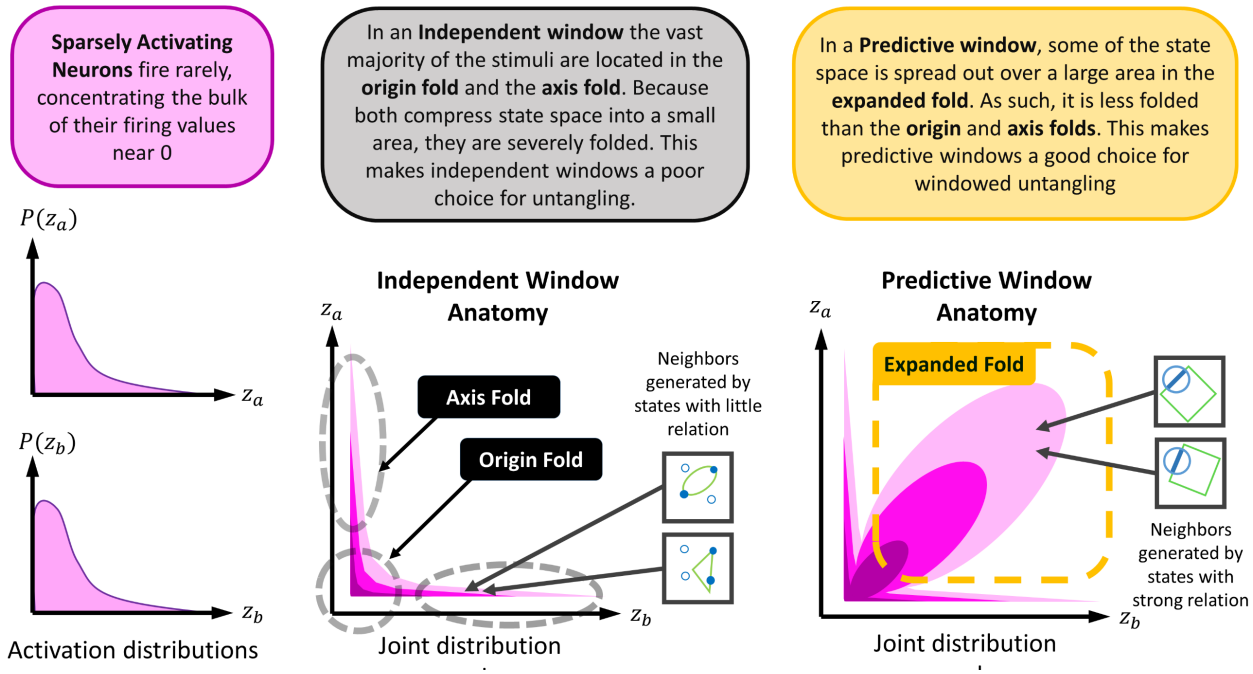

Figure 2.5. Anatomy of Folding in Sparsely Active Windows

Gives an overview of the parts of sparsely activating independent and predictive window probability distributions. (LEFT) Shows how sparsely activating neurons distribute their activation levels mostly at 0. (MIDDLE) Shows how an independent window formed from two sparsely activating neurons creates a specific joint PDF, that has two prominent parts, namely the origin fold which resides at the origin and the axis folds which reside along the axes of the window space. The null fold and the axis folds are severely tangled because they compress the entire state space into small slivers of the window space. Intuitive, neighbors in the expanded fold share little in common as it compresses virtually all of the state space to a point, and the neighbors on the axis fold tend to share little in common as well as they are related by only a single sensory unit activation. (RIGHT) Shows how predictive windows have another prominent part, namely the expanded fold which is a small subset of the state space which is flares into a larger portion of the available window space. The expanded fold is low density and neighbors in it tend to be generated by states with some strong relation (such as sharing an edge or high-level feature in a common position). Further, neighbors in the expanded fold will often be generated by neighbors on the temporal relation manifold. Because of these qualities, the expanded fold is far less folded than either the axis folds or the origin fold, making it ideal to perform untangling on.

is the main advantage of using a windowed form of untangling, as it can vastly reduce the computational requirements and number of learning examples needed to develop effective representations.

In short, predictive window learning takes advantage of statistical redundancies in real-world to vastly simplify the goal of progressively untangling temporal relation manifolds and object manifolds.

2.5.3 Untangling Across Windows and Layers

To better understand how information is integrated across windows we need to discuss how biological sensory systems are organized. A hallmark of low-level biological sensory systems is that they tend to densely sample on some set of physical axes. For example, early human vision densely samples across position and scale using features that are sparsely sensitive to contrasts at different positions and spatial frequency. Because of this sampling, small clusters of sensory dimensions or neurons will tend to be highly predictive of one another. This tends to be true at every level of representation in the human ventral stream. We can use this predictive structure to understand how information is hierarchically integrated.

We can imagine each sensory dimension or neuron z_i of the sensory space $\mathcal{Z}$ as a node on a lattice or a graph, where each sensory dimension z_i acts as a random variable. Neighbors on this graph are highly predictive of one another, and distant variables on the lattice are approximately independent of one another. We call this lattice structure a *predictive lattice*. Given this, we can interpret learning predictive windows as learning neighborhoods on the predictive lattice.

The evolution of the predictive lattice across multiple layers of representation is what drives integration across windows. A windowed untangling algorithm takes inputs organized by one predictive lattice and produces outputs organized by a new predictive lattice. This new predictive lattice tends to add new feature axes and contract old feature axes. For a

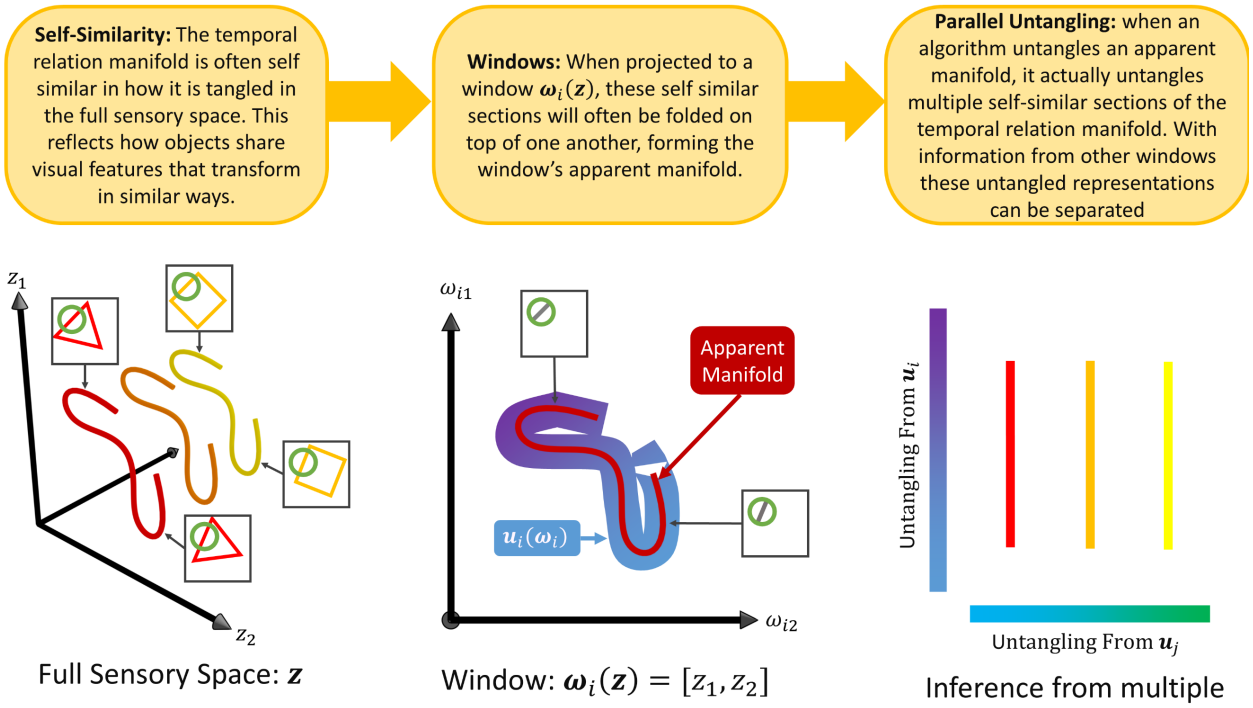

Figure 2.6. Parallel Untangling as Windows

This figure gives a brief overview of the parallel untangling property of predictive windows. The full temporal relation manifold for real sensory spaces and real sensory environments will tend to have large amounts of self-similarity in its shape in $\mathcal{Z}$. As such, when the temporal relation manifold is projected into a window space it will often cause many sections of the temporal relation manifold to overlap in a parallel manner, producing an *apparent manifold*. This means that untangling the apparent manifold can untangle the self-similar portions of the full temporal relation manifold that have been overlaid on one another. In practice, information from other untangled window representations will be necessary to split these untangled representations apart again. This property means windows can enhance the efficiency and generalization potential of an untangling algorithm.

visual neuroscience example, V1 introduces the feature axis of edge orientation and reduces the sensitivity to position inherited from the inputs it receives from LGN (see Fig. 2.7). This example shows how outputs of the new predictive lattice constructed from distant regions of the original sensory lattice can become neighbors, and outputs derived from neighbors on the original sensory lattice can become distant. The creation and contraction of feature axes across multiple stacked layers allow the top-level neurons of a windowed-untangling algorithm to integrate information across original sensory lattice (see Fig. 2.7).

While the predictive lattice is mostly a theoretical concept meant to characterize how systems of predictive windows will tend to be learned on an input representation, it can be used to more clearly conceptualize and guide discussions about how information is integrated across layers of representation that perform some kind of predictive window learning. Modeling the predictive lattice of sensory representations directly could also be a major component of predictive window learning architectures or a diagnostic tool for better understanding the behavior of such systems.

2.5.4 Context on Windowed Connectivity

While the concept of the temporal relation manifold is a straightforward extension of the existing object manifold framework, predictive window and predictive lattice learning are major extensions to the existing view of object recognition. Unfortunately, window choice is a neglected problem in the field of artificial neural networks (Quinlan, 1998). This lack of development may be blocking progress in modeling human ventral visual pathways.

Currently, the state of the art is to use Deep-Convolutional Neural Networks as proxies for deep cortical representation, however, these use fixed windows based solely on the retinotopy of low-level visual cortex (Fukushima, 1980). Researchers believe that experience during critical periods and adulthood refines the axonal windows of higher-level cortical areas (Innocenti and Price, 2005; Price et al., 2006). Further, research shows consistently that

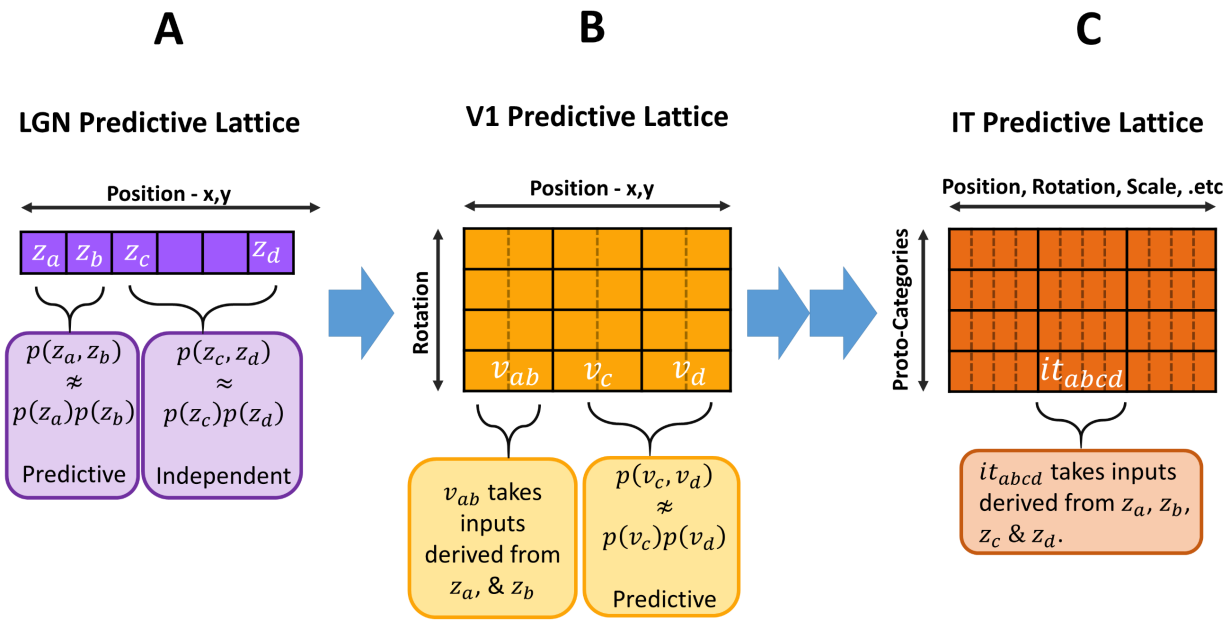

Figure 2.7. Predictive Lattice Development

This figure illustrates how predictive lattices evolve across layers, using human visual cortex as an example. (A) shows an illustration of the predictive lattice for LGN to V1. Note that neighbors are highly predictive but inputs that are distant on the lattice are not. (B) Illustrates how the outputs of V1 add new feature axes of organization to their predictive lattice and compress feature axes that were present in the input, note that output dimensions derived from distant inputs may now be neighbors on the new predictive lattice. (C) Shows how if this trend was continued out across multiple layers of cortex, IT level units would take inputs derived from inputs spread across the original sensory lattice. Each step of integration and window building is guided by the predictive lattice of the preceding layer's output, progressing from a space whose predictive lattice resembles the physical position of sensing units to a space with a predictive lattice that resembles a map of the temporal relation manifold.

high-level cortical areas become progressively less retinotopic (Grill-Spector and Malach, 2004). This suggests that high-level connectivity is probably not well described using simple convolutional windows. Because of this, the next generation of models should explore dynamics that develop learned window structures motivated by neuroscience. The concepts of sparse predictive windows and predictive lattices can guide these developments.

As discussed, a full-fledged windowed untangling algorithm needs to learn windows and integrate information across them. Later we will discuss a theory for how this may be accomplished (called Windowed Temporal Auto-Untangling) that integrates the problems of window choice and manifold untangling together and integrates information across windows via layering.

2.6 Synthesizing a Solution from Theories of Cortex

2.6.1 The Hierarchical Theory of Cortex as Temporal Auto-Untangling

Early studies of primary visual cortex suggested the existence of a hierarchy between two types of cells called simple cells and complex cells (Hubel and Wiesel, 1962). This theory came to be called the Hierarchical Theory of Cortex (HTC)(Hubel and Wiesel, 1962). Under this theory, simple cells respond to specific activation patterns while complex cells respond to multiple simple cell inputs generating invariance to mild changes in the stimulus. This theory has inspired many computational models such as neo-cognitron, the H-MAX models, and Deep-convolutional neural networks (Fukushima, 1980; Krizhevsky et al., 2012; Riesenhuber and Poggio, 1999).

In the following sections, we will detail how this hierarchy of simple and complex cells can be interpreted as performing a limited untangling operation on window folded temporal relation manifolds. Under our interpretation, this is done in a two-step process. First, the simple cells divide the input window space into small pieces, and then the complex cells bind

these small pieces together into groupings. These groupings are sensitive to regions of the temporal relatedness manifold (see Fig. 2.8 and 2.10 for an overview). We call this two-step process *Temporal Auto-Untangling* (TAU).

To understand these roles, let us first examine our goal. The ideal for untangling the temporal relation manifold would be to learn how every point in the sensory space $\mathcal{Z}$ is related to every other point according to the temporal relatedness metric on the environment state space Ω so that we can learn a projection into a region space. In practice, doing this individually for every point is intractable, given that there may be a practically infinite number of observable points in the sensory space $\mathcal{Z}$ and only a finite number of observations with which to relate them. However, the projection is roughly continuous on a small scale, meaning we can bin observations for small regions of the sensory space $\mathcal{Z}$. With this setup, we can instead relate the finite set of bins to one another, which is computationally tractable with finite observations. We will call the process of finding these bins *z-binning*. The next step is relating the bins together.

We call the process of relating the bins together *z-gluing*. In the z-gluing process, a set of units learn to respond to a temporally related group of bins. These groups of temporally related bins attempt to be sensitive to specific regions of the temporal relation manifold. Together, these overlapping groups of bins create a model of the topology of the temporal relation manifold in $\mathcal{Z}$ by learning to respond to a web of overlapping region functions spread across the temporal relation manifold.

At a high level, z-binning and z-gluing conform to the behavior of simple and complex cells. The z-binning process learns narrow receptive fields similar to the receptive fields of simple cells. The z-gluing process learns receptive fields that tolerate minor variations similar to the receptive fields commonly proposed for complex cells.

Actual simple and complex cells work in a windowed context. Meaning that a proper theory for how cortex implements Temporal Auto-Untangling would need to address how

neuronal windows are chosen. Conveniently, current theories and models of axonal development fit neatly with the concept of predictive window learning.

In the following sections, we will go into more detail about z-binning, z-gluing, predictive window learning, and how they relate to theories of simple cell learning, complex cell learning, and axonal development.

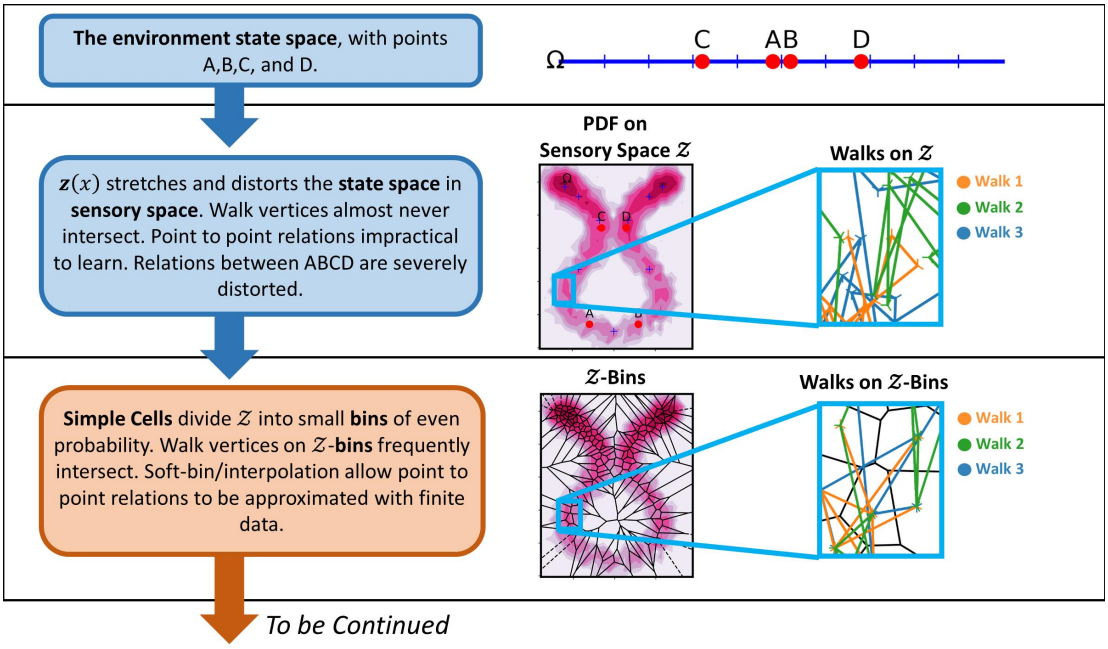

Figure 2.8. Z-Binning

This figure shows an overview of what a TAU untangling algorithm need to accomplish in a simple low-dimensional environment. Starting at the top, we have a state space Ω on a temporal relation manifold, with four points A, B, C, and D, highlighted in red. Next, the projection into sensory space warps and adds noise to the state space producing a PDF on the sensory space, causing the neighborial relationships between A, B, C, and D to change significantly. Also note that walks observed on the sensory space almost never intersect (i.e. the same exact stimulus will never be seen twice). Next the simple cells bin nearby locations on the sensory space together, which if small enough will tend to contain observation generated nearby regions of the state space. Walks on the bins of the sensory space will frequently converge and intersect.

2.6.2 SOM Models of Simple Cells as Z-Binning

For half a century, research has shown that the sensitivities of cortical neurons tend to vary continuously across the cortical surface in maps known as *cortical maps* (Hubel and Wiesel, 1974; Obermayer et al., 1990). These cortical maps tend to be representative of the stimulus distribution living on their input space. For instance, V1 representatively samples all the possible edge positions, orientations, and scales that tend to occur in its input space with more representation for more common combinations (Carreira-Perpiñán and Goodhill, 2002; Swindale, 2004).

Kohonen (1982) introduced the concept of *Self Organizing Maps* (SOM) models, many of which make excellent models of simple cell learning (Bednar and Wilson, 2016). In a SOM model, physically neighboring neurons constrain each other to have similar response properties and competitive learning forces these neurons to spread their response selectivity's across the input distribution in a representative manner (Bednar and Wilson, 2016; Kohonen, 1982; Swindale, 1996). As a direct consequence of these constraints, these models produce topographical maps very similar to cortical maps.

It turns out that many of the fundamental behaviors of SOM theories and models of simple cell learning are exactly those required for a strong z-binning algorithm.

As a reminder, the binning divides the input window space into small pieces, and gluing connects these pieces to better reflect the temporal relation manifold. In a simple analogy, TAU learning can be thought of as a connect the dots puzzle, where z-binning finds the dots, and the z-gluing finds the lines that connect them. If the z-binning process learns well-spaced dots, the z-gluing process can connect them into a coherent picture. On the other hand, if the z-binning process learns badly spaced dots, then the z-gluing process will have a hard time discovering useful connections between them. Next, we will discuss necessary behaviors for a z-binning process to find well-spaced dots and relate these behaviors to theories of simple cell learning.

The spreading behavior of SOM models of simple cells is the central behavior that a z-binning algorithm needs. This behavior causes the units of a SOM model to spread their receptive fields across the entire input distribution. To be effective, a z-binning algorithm also needs to learn a set of bins spread across all of the portions of $\mathcal{Z}$ generated by the environment. Without this, the bin representation could become functionally blind to many stimuli generated by the environment. As an exception to this, certain regions of window spaces should be ignored.

The distribution matching behavior of SOM models of simple cells is another critical behavior for the z-binning process. Distribution matching refers to the tendency for simple cells to distribute their receptive fields in a manner in proportion to the occurrence of input patterns. As a consequence, simple cells represent more frequently generated patterns with a higher sampling resolution. The z-binning process needs to perform distribution matching, as the density of bins will need to follow the density of the temporal relation manifold in sensory space. Essentially, areas with more of the temporal relation manifold will tend to be more highly convoluted, needing more bins to represent. On the other hand, areas with less of the temporal relation manifold will tend to be less convoluted, needing fewer bins to represent.

As a simple strategy to guarantee that the bin density follows the density of the temporal relation manifold, each bin can change its size and location to have the same overall likelihood of catching input patterns as every other bin (see Fig. 2.8). This also has theoretical benefits, as guaranteeing that the bins are roughly equal in likelihood makes it more conceptually consistent to use the temporal relation metric as a target for learning as it would otherwise need to incorporate the stationary distribution. The temporal relation metric M intentionally does not use the stationary distribution, because this would severely weaken it as a similarity metric. For an infinite continuous state space, the stationary distribution can be avoided simply by using an arbitrary measure that gives each point a uniform probability density,

and as a theoretical construct, this has little practical consequence. However, in the sensory space, an unequal stationary distribution across stimulus locations will need to be addressed. Bin size is an effective way to combat this issue.

In many SOM theories of simple cell learning, the activation of multiple neurons can be used to interpolate the position of a stimulus in sensory space more precisely. This also has a practical benefit for z-binning, as making the bins too small to increase accuracy will lead to an unwieldy number of bins per window and making the bins too large will make it difficult to approximate relationships between points accurately.

A simple way to implement bin interpolation is to use soft bins. Soft bins are bins that have degrees of membership for observations and with borders that fade gradually into one another. This means that an attribute of a point in sensory space can be approximated or interpolated by using a weighted average of that attribute for bins the point belongs to.

Bin interpolation has a second benefit. An interpolated bin function can preserve more information than a simple bin function, because a bin function is a piecewise discontinuous projection, while a bin interpolation function can be a continuous projection. This preservation of information is important because, in a multi-layer approach, the expectation is that the first layer cannot reach the desired transformation, and thus preserving information that may eventually be critical to the full untangling function is key.

While z-binning could in theory bin locations of a window or sensory space using something like competitive radial basis units, most models of simple cells assume that the direction of stimulus patterns is the chief concern. This can be accomplished using the typical weighted summation followed by non-linearity unit type found in most neural networks today. However, to make the output of such units act like directional bins, we need to apply some sort of competition between them to reduce the activity of distant bins and enhance the activity nearby bins. This can be accomplished using soft-max like operations or lateral inhibition and recurrent activation schemes. Generally, all the logic developed so far concerning z-

binning and z-gluing still applies even when the bins are spread on directions rather than locations within a space.

Some models seek to reduce their sensitivity to the length of vectors explicitly by soft normalizing the window activations. When using soft normalization on a window space, many activation vectors will explicitly be pushed to the unit hypersphere, while a large majority of patterns will still be folded at the origin (see Fig. 2.9). Because of this it makes sense for a Z-binning algorithm to treat the region near the origin as a dead zone as it is too compressed to learn useful bins on (see Fig. 2.9). Under a soft normalization, scheme the expanded fold will be mostly compressed to the surface of a hypersphere (see Fig.2.9). All of the logic developed so far also applies to binning near the surface of a hypersphere, or more complicated hypersurfaces.

To wrap-up the discussion of z-binning, it would be useful to give an example of how a specific simple cell model can implement z-binning. As an example, consider the LISSOM model (Sirosh and Miikkulainen, 1994) on acting on single input window. The LISSOM or Laterally interconnected synergistically self-organizing map model, was originally designed to be a more biologically plausible alternative to the more widely known Kohonen SOM model (Kohonen, 1982). LISSOM will learn to distribute its units' receptive fields according to the distribution of input patterns, satisfying the dynamic resolution requirement of a z-binning algorithm. LISSOM does this by picking a subset of the neurons in a 2D map to learn to respond more strongly to the current input activation pattern, after every training stimulus. These neurons are chosen through competitive interactions mediated by use of lateral inhibition, excitation, and recurrent activation. Further, when presented a stimulus, only a small number of neurons whose weights are near the stimulus will be activated due to this lateral inhibition, this can be thought of as a soft-bin representation for the direction of the stimulus in sensory space. Given these behaviors, models of simple cell learning like LISSOM are powerful z-binning mechanisms.

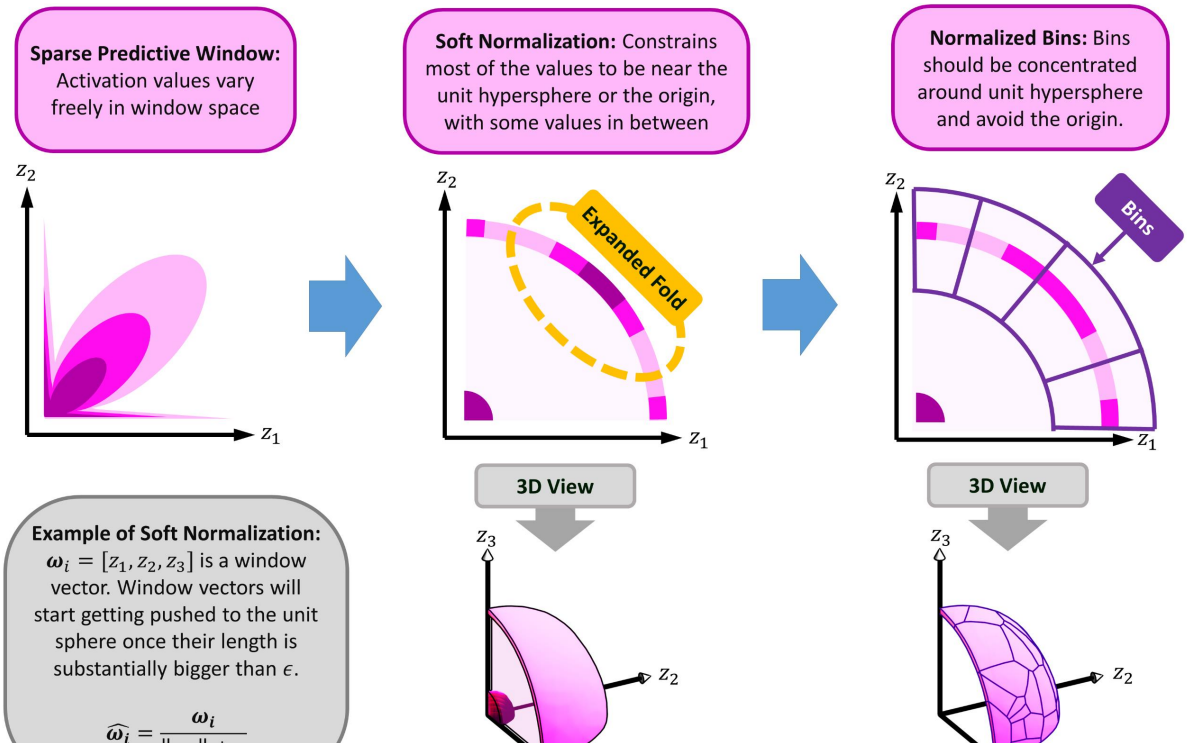

Figure 2.9. Z-Gluing

This figure show how soft normalization distorts the activation distribution of a window space. Soft normalization will often push most of the non-zero activation values to the surface of the unit hypersphere (Note, that in a sparse predictive window the vast majority of values will still be close to zero). Note that for the volume containing non-zero value this is only a small drop in dimensionality, and thus the expanded fold of a predictive window will still fold the temporal relation manifold far less than the origin or axis fold. Given this, the bins should be restricted to responding to the area near the unit hypersphere's surface. As an interesting side note, the bins on the unit sphere's surface can be regarded as a type of Voronoi cell system.

To improve the dynamic resolution aspect of LISSOM, we could go a step further and use mechanics from the GCAL extension of LISSOM (Stevens et al., 2013). GCAL features a homeostatic adaptation that causes each of the units on its map to attempt to have the same average firing rate, by adjusting each unit's pre-threshold bias term. This can be interpreted as units widening or narrowing the size of their bins in order to have approximately the same probability of being entered, which is another desirable quality for a Z-binning mechanism. GCAL also includes something called gain control, which loosely approximates a soft normalization scheme for the window spaces, although it distorts the activation distribution in much more complicated manner.

2.6.3 Trace Rule Theories of Complex Cells as Z-Glueing

There have been many theories proposed for what complex cells do in the visual hierarchy and how they learn, but here we will focus on trace theories. Trace theories of complex cell learning assume that images which appear next to each other in time tend to come from the same object or event, and that this can teach complex cells a limited form of invariance (Földiák, 1991; Wallis, 1996). Trace theories accomplish this by having a complex cell attempt to predict a running average of or temporally smoothed version of its recent activations using only its current input (Michler et al., 2009; Rolls and Stringer, 2001; Wallis, 1996). This causes complex cells to learn receptive fields that are less sensitive to time than their simple cell inputs, and in theory less sensitive to natural image variation within categories and for common view transformations.

To understand how trace theories of complex cell learning connect to z-gluing it would be useful to recall where the simple cells left off. The simple cell layer divides the sensory space $\mathcal{Z}$ into many bins on a sensory or window space, but we do not know which of these bins represent neighboring regions on the temporal relation manifold. We can learn this by observing how temporal sequences generated by the environment travel across the bins.

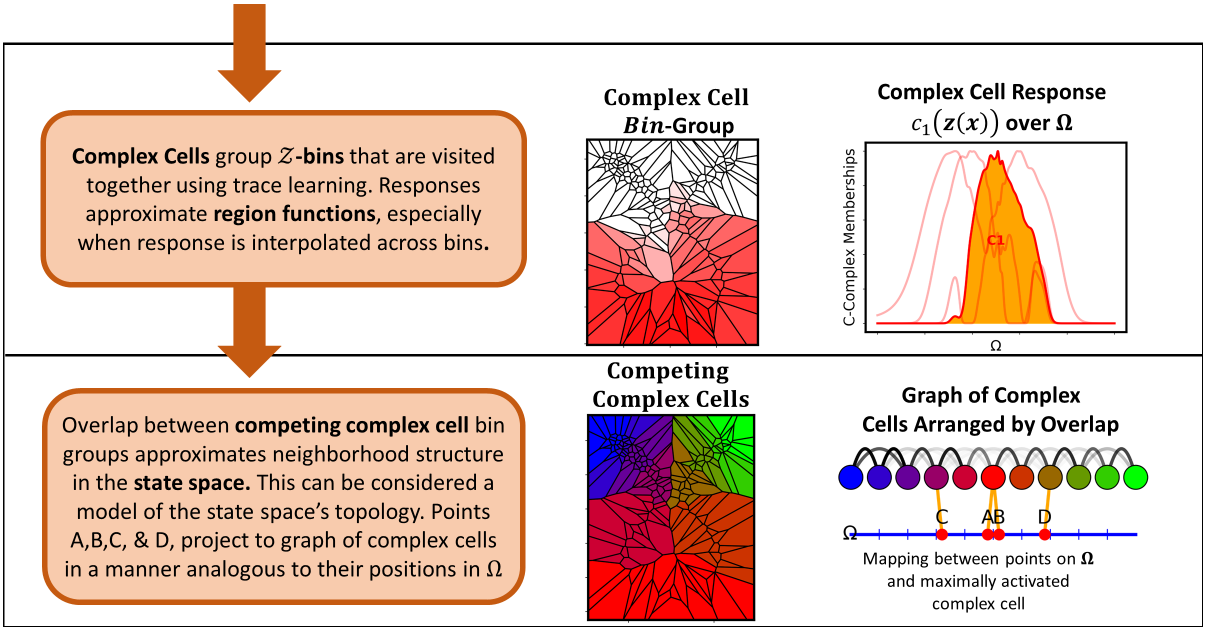

Figure 2.10. Z-Gluing continued from Z-Binning

Continuing the process from the z-binning figure, the complex cells group sensory space bins together that tend to be visited close together in time. The red gradient indicates the membership for a single complex cell, note that that complex cell's response approximates a region function over the state space. Altogether, the competing complex cells will spread their approximate regions functions across the visible portions of the state space projected into the sensory space. This system of complex cells and their membership overlaps can be used to approximate the topology of the environment state space and the temporal relatedness metric. While this example shows a simple temporal relation manifold embedding without the need for windowing, it also demonstrates how powerful a TAU algorithm is at linearly untangling state space regions.

These temporal sequences or walks will trace the structure of the temporal relation manifold linking bins that represent neighboring areas of the temporal relation manifold. We can use trace learning to encode the linkages or glue between the bins.

Trace learning models the temporal relation manifold in a straightforward manner. In learning to make their activity more stable, a complex cell will distribute its weights across all of the simple cells or bins that are likely to activate in a short time span from its own activation (Michler et al., 2009; Rolls and Stringer, 2001; Wallis, 1996). The way trace learning distributes the weights of complex cells, is directly analogous to the way the temporal relatedness metric M evaluated around a point highlights points in state space or sensory space that are likely to occur within short time spans (see Fig. 2.1 & Fig. 2.10). Given this, complex cells can be understood as learning to respond to regions of the temporal relation manifold defined by M. Further, competitive dynamics will cause complex cells to distribute their sensitivities across the available regions of the temporal relation manifold defined by M (see Fig. 2.10). As a result of their coverage and overlap, the regions learned be used to approximate the nearness between any two stimuli Z on the temporal relation manifold (see Fig. 2.10).

To give an explicit example of how a trace rule can be used to implement z-gluing, we will consider a simple LISSOM model where its Hebbian update rule has been modified to use a "delayed trace rule" and its inputs are z-bin membership values. Here delayed trace rule refers to a trace rule where the temporal average or trace of a layer's activity is dominated by a small time period before the current time step. This has the impact of reducing a complex cell's bias towards reinforcing only its most preferred input pattern (as the standard trace rule does) and corresponds to modeling a more general temporal relatedness metric M with a nonzero σ delay parameter. Similar delayed trace rules have been examined and proposed in the past, and researchers have shown they bear strong relations to temporal difference learning and error correction (Rolls and Milward, 2000; Rolls and Stringer, 2001). A complex

cell modeled by such a delayed trace LISSOM model would learn to activate in response to a subset of z-bins that were highly temporally related to one another according to an M like metric, and the competitive interactions of the LISSOM model would push each complex cell to learn a unique group of temporally related z-bins. Together, these mechanisms would implement z-gluing quite efficiently.

Researchers have long proposed that complex cells may perform a Max like operation (Riesenhuber and Poggio, 1999). So, as an additional modification of our implementation it might be useful to change the activation function of the complex cells from a weighted sum to a weighted max function. Here meaning weighted by each bin's membership in a complex cell's group. This would also make a z-gluing algorithm correspond more to the pooling layers of contemporary convolutional neural networks which typically use spatial maxpooling operations (Krizhevsky et al., 2012; Yamins and DiCarlo, 2016b).

2.6.4 Axonal Plasticity as Predictive Window Learning

The role of axonal plasticity in understanding the cortical algorithm has mostly been ignored in neural network modeling circles, even though axonal plasticity has a huge impact on what neurons can represent.

For decades, research has shown that axon tracts connecting to cortical areas and between cortical areas tend to project in orderly maps that often preserve the projecting area's topology, while also being sensitive to the activation dynamics between the areas (Benson et al., 2001). When long-range primary axons reach their destination tissue they branch out into axonal arbors, think a literal tree of axon branchlets. The branchlets of these arbors will grow and shrink in specific directions while maintaining the same overall size of the arbor (Gogolla et al., 2007; Meyer and Smith, 2006; Portera-Cailliau et al., 2005; Ruthazer et al., 2003). This means the axonal arbors effectively move across their target tissue in response to certain cues. For our purposes, we will focus on the activity-dependent mechanism. Due

to the activity-dependent mechanism axonal arbors migrate towards other axonal arbors and possibly towards the dendritic arbors of neurons that tend to fire together (Jamann et al., 2018; Katz and Shatz, 1996). This mechanism organizes the small-scale wiring of cortical projection maps with a high precision on top of other forces which tend to organize projection maps on a larger scale.

Axonal plasticity with emphasis on the *activity-dependent* mechanism appears to implement a sophisticated form of predictive window learning. When axonal arbors group with other co-activating axonal or dendritic arbors it changes the structure of neuronal windows for the receiving tissue. When axonal arbors group this way, a receiving dendritic arbor will only be able to synapse with axons that are highly predictive of one another forming a predictive window. At a higher level, the collection of these windows will model the predictive lattice for a set of inputs. This is because other forces in axonal learning, such as competition for dendritic real estate, will force a model to learn windows sampled from across the predictive lattice in a manner that is representative of the available neighborhoods.

We can implement predictive window learning in neuro-plausible manner using a modified multi-factor axonal development model. The Fraser and Perkel (1990) multi-factor model was the first to integrate the activity-dependent mechanism with other forces in axonal development. Unfortunately, the original Fraser and Perkel (1990) model did not feature neural activation dynamics and instead used a stand-in to control the attractive force on axonal projections targeting a layer of cortex. The endpoints of the axonal projections are called *axonal arbors*. We would propose updating the model to have explicit neural activation dynamics. Further we would propose using a fully differentiable Hebbian like rule for modeling how co-activating axonal arbors can be attracted to one another and to co-activating post-synaptic dendritic arbors. Allowing axonal arbors to co-locate as a result of co-activation will implicitly build post-synaptic neural windows that are linearly predictive and will thus implement a type of predictive window learning all on its own. But if

axonal arbors can co-locate as a result of co-activation with post-synaptic neurons, then even more complexly related axonal arbors can be pulled together into windows. Together, these mechanisms would make an updated Fraser and Perkel style model a powerful predictive window learner.

2.6.5 Putting TAU and Window Learning Together

When predictive window learning and TAU (z-binning and z-gluing) are combined, they allow successive cortical layers to both untangle and unfold the temporal relation manifold of the environment into representative convex regions of the environment state space (see Fig. 2.11). Working together these mechanisms should be able to untangle the temporal relation manifold because each successive layer will have complex cells that respond to fewer and fewer discrete regions of the state space as more layers are added, converging to complex cells that only respond to one region each of the environment state space after a finite number of layers (see Fig. 2.11). After this state has been reached, categories should be linearly separable in the region space if the temporal statistics of the environment support strong categorical distinctions. In practice meaningful categories may be linearly separable well before the temporal relation manifold has been fully untangled into an ideal region space.

Together, these mechanisms constitute a novel computational theory of how visual cortex may separate object categories in a layer-wise unsupervised fashion. We will call this full theory *windowed-TAU* or W-TAU.

2.6.6 The Potential Benefits of Windowed-TAU

Implementing Windowed-TAU could advance cognitive neuroscience and machine learning in several important ways, but in this paper, we will focus on two.

W-TAU models may advance unsupervised learning and be far more data efficient than contemporary models in general. Current state of the art supervised learning models require

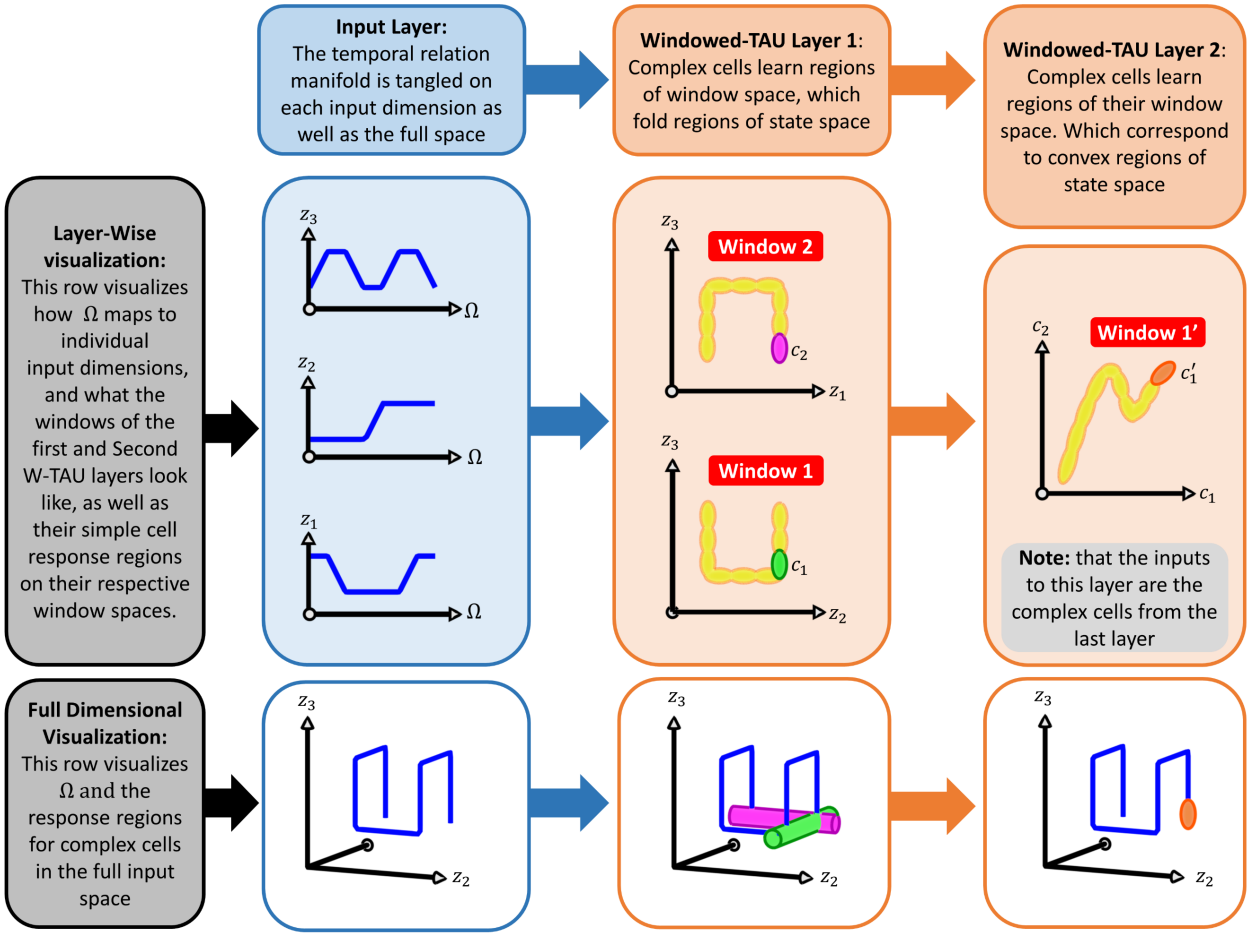

Figure 2.11. Putting TAU and Window Learning Together
This figure gives a simplified overview of how the complex cells of a hierarchy of Window-TAU layers respond to more and more convex regions of the state space as more layers are added. The temporal relation manifold is only modestly tangled in the full sensory space z, but it is folded on the available windows. The first TAU layer learns complex cells on window 1 and window 2, but each of the first layers' complex cells c_i respond to multiple regions of Ω folded together. The second TAU learns complex cells on the window $1'$, but its complex cells c'_j can respond to singular regions of Ω. Note that this example is for visualization only, and aspects have been simplified. For example, the input units are not sparsely activating, and the windows are chosen in a completely arbitrary fashion, to make it easier to follow.

vast amounts of cleanly labeled data to learn effective representations, while unsupervised machine learning models are simply not effective to the same degree even with large amounts of data. To understand how W-TAU might address this it is useful to look at how contemporary models are built.

Most state-of-the-art models today use an abstract measure of behavior and deep error propagation down through their layers in order to learn. For example, most networks use deep gradient descent with a simple classification objective that measures the difference between a model's guess and the ground truth. By definition, these techniques will seek a path of least resistance to raise their performance and do not care about the nature of the features that they learn. We theorize that this behavior may ultimately cause deep neural networks to learn large collections of low-level image statistics rather than high-level information about object categories. This would happen because low-level image statistics are a kind of plentiful low hanging fruit in natural image datasets that are easy to learn. Research suggests that this learning of facile features does indeed happen and may be endemic to today's architectures (Ilyas et al., 2019; Jo and Bengio, 2017; Szegedy et al., 2014b). We theorize that this 'low-hanging-fruit' form of learning requires large amounts of labeled stimuli to recover enough low-level image statistics to reach a reliable level of performance. If this theory is correct, then data crunch may be a persistent problem for conventional deep-learning models with no simple fix.

W-TAU models, on the other hand do not use deep error propagation. Every, neuron and connection in W-TAU is competing to learn something useful and unique from all its companions, with 'useful' being fully locally defined. Further, W-TAU is not trying to optimize an abstract behavioral metric, rather each successive layer of neurons is independently trying to improve its mapping of the underlying state space. Because of this, W-TAU is unlikely to take advantage of 'low-hanging fruit' or facile image statistics when choosing features. In fact, it may be pre-disposed to progressively ignore low-level regularities as more layers are

added, because low-level regularities will not help it learn a good mapping of the state-space using purely bottom-up local mechanisms.

Interestingly, high-level neurons in deep neural networks such as deep convolutional neural networks tend to learn semantic information at the space level (Parde et al., 2020). From a purely computational perspective, the objectives and top-layer architectures for these networks make no distinction between different rotations of feature sets. Thus, it would be highly unlikely for a network like this to learn a feature space where the meaningful directions in the space happen to be along the space's main axes (i.e. meaningful unit representation). This may even help these networks perform better per-unit, but it also means they may not be very appropriate for modeling high-level learning in the ventral stream where neurons are generally thought to be somewhat specialized. In contrast, the units of W-TAU models learn regions of the temporal relation manifold. Under, the right circumstances these regions will be semantically meaningful, making W-TAU a potentially better candidate for modeling high-level visual learning on a unit level.

Given W-TAU's unique qualities, it may hold significant advantages over current conventional models. However, W-TAU models are untested, and it is quite possible that they may not work at all. If they do work, implementations of W-TAU could face other challenges.

2.6.7 Challenges for Windowed-TAU

While z-binning, z-gluing, and predictive window learning can be implemented by many existing algorithms with only mild modification and combination, there will likely be some difficulties associated with building and training algorithms that fully implement windowed-TAU.

Firstly, today's hardware and software libraries are optimized for convolutional operations, not dynamically changing sparsely connected neuronal windows. While clever math can be used to make fully connected layers behave like dynamic window layers, this approach

is orders of magnitude less efficient and does not scale well on today's hardware. Importantly, implementations that approach the efficiency of convolution are possible, as we have implemented them repeatedly, but they rely on operations which have not been thoroughly optimized to work with today's parallel computing architectures.

Second, most of the models mentioned that implement z-binning, z-gluing, and predictive window learning, tend to be finicky and hyper-parameter sensitive with no clear way to optimize these hyper-parameters past first layers where early visual cortex can be used as a guide. In theory, this might best be addressed by updating these models to auto-tune their hyper-parameters online using simple heuristics, but this will take experimentation.

Third, the choice of datasets that would truly leverage windowed-TAU learning is poor. Most datasets intended for object category classification do not contain natural temporally related observations for all their subjects. On the other hand, most sequence learning datasets do not contain a rich array of object categories as their focus. In the short term, this could be addressed in several ways. Datasets which do not include temporal observation could have faux temporal observations created by using progressive image transformations and arranging related categories to be presented serially with a high likelihood. Alternatively, a windowed-TAU algorithm could focus on learning to differentiate specific identities within an object category, such as learning to recognize faces from a dataset of short video clips. These dataset problems that will likely be solved as datasets evolve naturally.

These challenges suggest that the first tests of windowed-TAU will probably be on a small scale and that it may be some time before the wider benefits of windowed-TAU algorithms can be examined fully, if it can be implemented successfully at all.

2.7 Simple Implementations of Windowed-TAU

2.7.1 Basic TAU Example

Here, we will provide a brief sketch of a temporal-auto untangling model that lacks window learning. Fig. 2.12 shows a two-layer model where each layer is a Kohonen self-organizing map. The first layer or simple cell layer spreads radial basis units out into the sensory space to detect small regions of the temporal relation manifold. The second layer or complex cell layer looks at a weighted temporal average or trace of the first layer's activations, which allows it to learn groups of simple cell units that are related.

While this model can learn to represent the topology of a simple temporal relation manifold, it will struggle with the complexity of a real-world situation because it cannot take advantage of the self-similarity of more complex temporal relation manifolds via window learning. Further, without window learning there is little reason to stack this model hierarchically as information does not need to be integrated across windows. To compensate for the lack of windows or window learning this model will need to be prohibitively wide and see an excessive number of examples to be effective in real world environments.

While this basic TAU model is unlikely to be useful in more demanding situations, it does outline the loose layer structure that more sophisticated W-TAU modules will use and how the TAU process can work in simplified low and high-dimensional environments.

2.8 Discussion

In the *re-defining the problem* section, we extended the object manifold framework by replacing unobservable labels and artificial category boundaries with an observable Markov chain walk based similarity metric. This similarity metric formally captures many heuristic ideas about trace theory and allows concepts from object manifolds to be directly integrated with unsupervised temporal learning theories. In this section, we also used this new framework

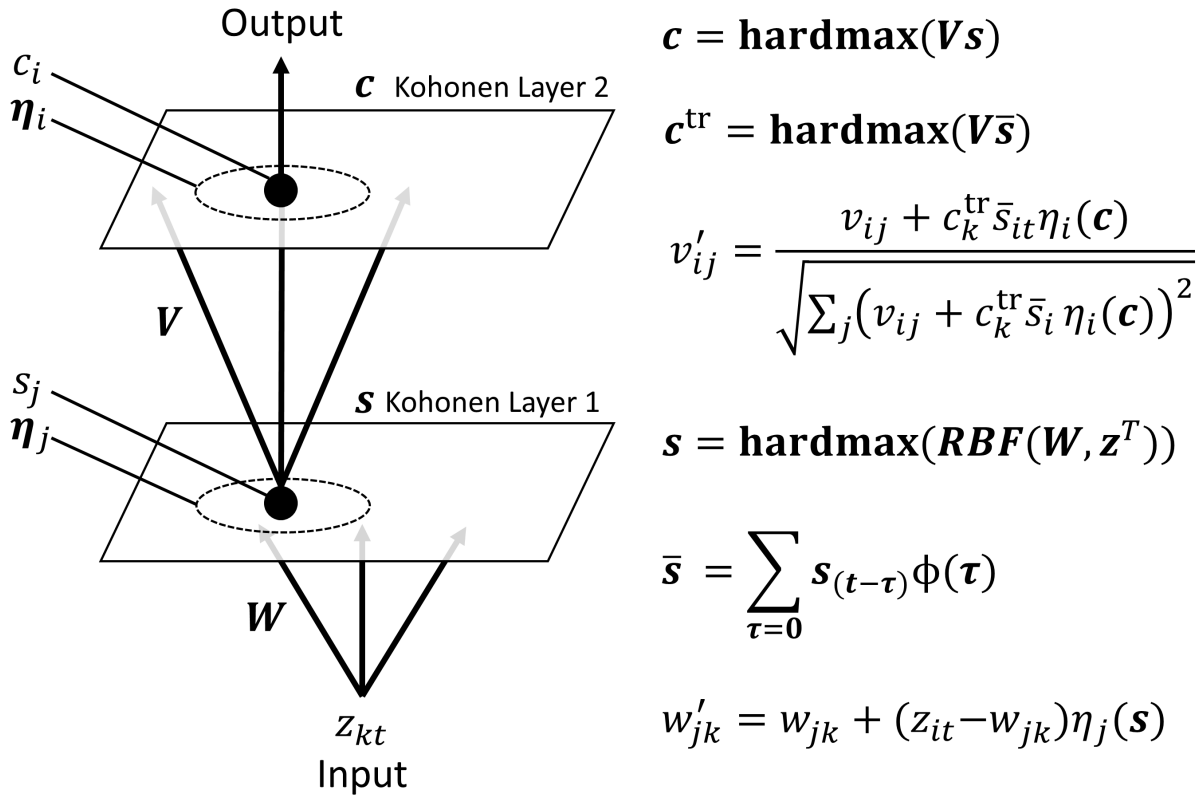

Figure 2.12. Shallow TAU model

Shows a basic TAU model, where **z** is the input sensory space, **s** is output of the simple cell layer, **c** is the output of the complex cell layer, and η refers to the lateral influence of neurons on one another as a function of the current activation of the layer. $\bar{\mathbf{s}}$ calculates a trace for the activations of **s** weighted by the ϕ function. Finally, note that the weight update function for the complex cell layer is dependent on the best match for the current trace $\mathbf{c}^{tr}$, while the actual output of the network is only dependent on the current activation of the simple cells **s**.

to understand why windows should be built from sparse predictive units, as seen in real sensory systems, rather than sparse independent units or random selections of units. Predictive windows "fold" the temporal relation manifold far less than non-predictive windows and leverage the self-similarity of the temporal manifold to reduce training and representation. Further, the global structure that the windows follow, namely the predictive lattice, directly guides how information is integrated across layers.

In the *synthesizing a solution* section, we used the temporal relation manifold framework to conjecture potential roles for axonal learning, simple cell learning, and complex cell learning, in learning a cascaded untangling function. These roles are predictive window learning, z-binning, and z-gluing. Taken together these roles constitute a novel computational theory of object recognition. This theory which we call W-TAU can be implemented by combining existing models with modest adaptation. The potential advantages of W-TAU models suggest they may be a useful avenue for future model development. However, they are untested and may have their own practical drawbacks.

In summary, we extended the DiCarlo and Cox (2007) object manifold framework of object recognition to robustly integrate theories of unsupervised object category learning, which was lacking their previous framework. This extension introduced several novel fundamental concepts: temporal relation manifolds, predictive windows, and predictive lattices. Using this new framework, we developed a new theory of cortical learning called windowed-TAU, that integrates aspects of simple cell learning, complex cell learning, and axonal development theories, within the temporal relation manifold and predictive sensory lattice framework. We hope that these concepts and theories can provide a new way of looking at the problem of object recognition and investigating the solutions visual cortex may use.

CHAPTER 3

A MODEL OF AXONAL PLASTICITY

3.1 Preface

Chapter 3 is a version[1] of the published article Ryland (2021), which gives a brief review of axonal plasticity phenomena and introduces a model to better integrate these phenomena into typical artificial neural network architectures. The integration of axonal plasticity with other cortical learning dynamics is a major contribution of the ICL model. Although Chapter 2 briefly discusses axonal plasticity, it requires a more in-depth treatment as it is a very major addition to the standard neural network framework. As an additional note, the axonal plasticity model introduced in this article directly inspired the axonal plasticity model that will be used in the ICL model, but it is substantially different in implementation. However this article should give a good overview of the types of concerns axonal models of connectivity need to address. The specific axonal model used in ICL will be discussed in detail in chapter 4.

This chapter is also directly relevant to how the ICL model addresses the three issues raised in Chapter 1: unsupervised learning, the unit specialization debate, and the development of cortical maps.

The ICL model address these issues by adapting its connectivity using an axonal development model similar to the one featured in this article. In theory, this means that the ICL model should build appropriate window structures for feature development at every level of representation and in a fully unsupervised manner.

Relevance to Cortical Map Formation. Many axonal development models including the ones developed in this manuscript are cortical map models. This paper demonstrates

[1]Chapter 3 is adapted from Ryland (2021) first published in *Neural Processing Letters, 53*, pages 1119–1146 (2021) by Springer Nature. Reproduced with permission from Springer Nature.

that axonal learning supports the development of many low-level topologically organized cortical feature maps like those seen in V1 simultaneously. The use of this kind of axonal plasticity model in the ICL architecture will help it learn topologically organized high-level feature maps like those seen in the high-level ventral visual cortical areas.

Relevance to Unit Specialization. Although this topic is not addressed in the article, the use of axonal plasticity dynamics forces the ICL model to use narrow windows at all levels of representation. As a consequence of these axonal learning dynamics, a single unit can only connect to a small number of units in the next layer that are physically adjacent. As such, even though the connections are learnable, they are highly constrained in a manner that likely promotes the development of more specialized feature representations and inhibits the development of less specialized feature representations.

Relevance to Unsupervised Learning. From a neural mechanism perspective, DCNNs lack what may be a critical type of plasticity. Real cortical layers are connected by axonal projections, which form by a complicated process guided by multiple interacting forces (Benson et al., 2001). These axonal projection maps are highly plastic, guided by both genetic predispositions and sensory experience (Innocenti and Price, 2005; Price et al., 2006), as will be discussed in this article. On the other hand, DCNNs use a fixed connectivity (or window) structure that partially models the retinotopically organized connectivity learned by primary visual cortex V1 (Fukushima, 1980). However, this simple connectivity structure is likely a poor model of higher-level cortical connectivity, as higher areas are nowhere near as retinotopically organized (Grill-Spector and Malach, 2004). The use of these inappropriate connectivity structures may actively impede strong unsupervised learning as connectivity can be viewed as a critical guide to effective feature development.

3.2 Abstract

Axonal growth and pruning are the brain's primary method of controlling the structured sparsity of its neural circuits. Without long-distance axon branches connecting distal neurons, no direct communication is possible. Artificial neural networks have almost entirely ignored axonal growth and pruning, instead relying on implicit assumptions that prioritize dendritic/synaptic learning above all other concerns. This project proposes a new model called the axon game, which allows biologically-inspired axonal plasticity dynamics to be incorporated into most artificial neural network models in a computationally efficient manner. First, we demonstrate that the axon game replicates multiple previously defined pre-synaptic cortical maps. Second, we demonstrate that the axon game integrated with a synaptic learning model similar to the Laterally Interconnected Synergetically Self-Organizing Map (LISSOM), can simulate the interaction of axonal plasticity and synaptic plasticity within one model creating both pre-synaptic and post-synaptic cortical maps. Finally, I show that pre-synaptic and post-synaptic maps can be decoupled from one another dependent on the relative sizes of dendritic and axonal arbors. This coupling phenomenon indicates a novel theoretical prediction about how axonal and synaptic dynamics interact.

3.3 Introduction

Humans integrate myriad sensory signals into high-level understandings of their environment. To aid integration from one layer of cortex to another, neurons in a given layer receive input from only small groups of neurons in a preceding cortical layer (Hubel and Wiesel, 1962, 1965). In this paper, we define a group of neurons that forms the input to a particular neuron as its *window*. The main constraint on a neuron's window are the axons that project to that neuron. Axons fundamentally limit which neurons can synaptically communicate with one another. In order to form synapses, the short-range dendrites of the receiving

neuron must make contact with the long-range axonal branches of the input neurons. In this sense, axonal development is of critical importance because it specifies which neurons a given neuron is capable of forming synapses with during gestation, early development, and maturation. The structure of axon projections can be represented abstractly as a mask on an artificial neural network's sparsity structure.

Axonal development and plasticity contrasts with synaptic development and plasticity. Once an axon has made contact with the dendrites of a neuron, synapses can form, change in number, and change in efficiency. As such, axons are often thought to have a roughly binary influence on whether one neuron can talk to another (Quinlan, 1998), whereas synapse plasticity affects how one neuron influences another in a roughly continuous fashion. Synaptic/dendritic learning is abstractly represented in most artificial neural networks as weight changes that are determined by gradient descent, Hebbian dynamics, or other learning rules.

Given these basic aspects of neuroscience, it seems like the biological processes that subserve axonal growth/pruning and synaptic plasticity are largely distinct. In this project, we propose that the computational goals of axonal growth/pruning and synaptic plasticity may also be distinct. The existing literature has largely considered this to be a single process, and the vast majority of architecture development has been focused on abstractly modeling the synaptic qualities of neural development. This paper explores axonal-development modeling as a companion to synaptic/dendritic development.

In *Section 3.4*, we briefly review some of the basic findings of the field of axonal development for the purpose of relating some of the known principles of axons to artificial neural network architectures.

In *Section 3.5*, we will review how current artificial neural network architectures approach axonal development, and how models in computational biology have already approached axonal development with greater emphasis.

In *Section 3.6*, we introduce a direct model of axonal development called the axon game. The axon game is a model that allows for the incorporation of abstract versions of many

of the behaviors of axonal development, which can then be incorporated into many modern neural network architectures.

In *Section 3.6*, we further outline the axon game with more specific detail.

In *Section 3.7*, we demonstrate that the axon game allows for the development of many different cortical maps using a single, large-scale cortical area simulation of axonal arbor development. These maps include visual field maps, spatial frequency maps, and ocular dominance maps.

Finally, in Section 3.8, we use the axon game in combination with a modified version of a common neural network model of cortical map development to demonstrate the feasibility of jointly simulating axonal map and synaptic map dynamics efficiently within the same model. We show the development of pre-synaptic ocular dominance maps post-synaptic ocular dominance maps and post-synaptic orientation maps within the same simulation. We also discuss a novel prediction that arose from my simulations of axonal and synaptic dynamics, namely that axonal arbor maps and corresponding synaptic cortical maps can be highly coupled or decoupled with one another depending on several factors under this model.

3.4 Axonal Development and Plasticity Review

One of the primary types of evidence used to understand the structure of neuronal windows is *axon tracing* (McLaughlin and O'Leary, 2005). Axon tracing is a type of staining methodology that allows the axon branches of a particular neuron to be traced to the neurons they project to. This method has been used extensively to understand how axons from one area of cortex (or sub-cortex) grow towards another area in response to either initial chemical signaling, early noise patterns generated during gestation, experiential cues during critical periods, or experiential cues into adulthood.

Another important type of evidence that informs neuroscientists about cortical window structures are *cortical maps*. The representations of neurons in adult cortex are topologically

organized on the surface of the cortical sheet in most mammals. To use V1 as an example, neurons that represent local visual features that are adjacent to each other in the visual field will tend to be adjacent to each other on V1's cortical surface. This further suggests that the windows for neighboring neurons on the cortical sheet are similar too.

Evidence from developmental axon tracing and cortical map studies together have painted a compelling picture of axonal development over the life spans and many species. Here we focus on axons projecting to the cortex from subcortical areas, as well as axons projecting from one cortical area to another. Particular emphasis is placed on axonal projections in the visual system, as it is more studied than other sensory domains. Further, we will focus on summarizing the aspects of axonal development that are preserved across both human development and the development of many mammalian species (e.g., mice, rats, cats, and monkeys). The major phenomena discussed will include (1) the initial directed growth of axons guided by chemical gradients, (2) the subsequent exuberant arborization of axon branches, (3) growth/pruning guided by spontaneous activity, (4) the critical period, and finally (5) adult axonal plasticity. For a far deeper review of these topics, see Price et al. (2006); McLaughlin and O'Leary (2005).

3.4.1 Initial Projection and Chemical Gradients

The first stage of axonal development is largely governed by *chemoaffinity* (Sperry, 1963), a process in which axons are guided by a form of molecular identification referred to as a tag. These tags guide the axon toward cortical areas marked by attractive molecular markers, and away from areas with repulsive markers. Interestingly, these tags and markers stick around after gestation and are thought to allow the healing process to guide new axons to project to the correct areas. In this process axons occasionally project to the wrong cortical areas but are mostly pruned during early gestation.

Later research into chemoaffinity has shown that the systems of tags and markers goes beyond simply telling projecting axons which gross areas of neural tissue to project to, and

can further specify location within that area of neural tissue using gradients of attracting chemicals and counter gradients of repellents, as studied in sub-cortical vision. This is true both for subcortical (Gierer and Lewis, 1983; Gierer, 1987) as well as cortical projections (Bishop et al., 2000, 2002, 2003; Fukuchi-Shimogori and Grove, 2001; Hamasaki et al., 2004). This process allows the axons projecting to V1 to create an initial retinotopic map completely independent of neural activity. Similar processes likely guide axonal development in all primary sensory areas (possibly excepting olfactory cortex). It is likely that this chemical signaling provides initial retinotopy for higher layers of visual cortex, too.

3.4.2 Exuberant Arborization

In most species studied, once the main branches of the projecting axons reach the correct destination tissue, they start to grow exploratory branches laterally within the tissue. Evidence suggests branch growth is almost balanced out by branch retractions, leading to a gradual adjustment of the arbor location (Gogolla et al., 2007; Meyer and Smith, 2006; Portera-Cailliau et al., 2005; Ruthazer et al., 2003). This exploratory lateral branching further enhances the retinotopic mapping governed by chemoaffinity both in sub-cortical vision (Simon and O'Leary, 1992; Simon and Leary, 1992; Yates et al., 2004) and early cortical vision (Bishop et al., 2000, 2002, 2003; Hamasaki et al., 2004). This process also creates an excess of broadly tuned axonal connections in the target tissue.

3.4.3 The Critical Period

Recent work suggests that many of the high-level properties of cortical maps are already present in primary visual cortex before experience driven growth and pruning take place (Crair et al., 1998; Crowley and Katz, 2000). For example, in most studied species, ocular dominance columns (stripes of cortex that respond preferentially to input from a specific eye) are already present in V1 before infants of that species first open their eyes. This is

possibly driven by spontaneous patterns of retinal activity. However, during a period after gestation, these maps and the axonal structures that support them can be greatly altered. For instance, preventing the brain from receiving input from one eye during this period for V1 will cause the ocular dominance columns to be altered or destroyed (Chapman et al., 1986). This period of development, marked by extreme sensitivity to exterior stimuli, is called a critical period. The timing of critical periods appear to be staggered for different cortical areas in the visual hierarchy, with high-level visual areas ending their critical period later than earlier visual areas. For review, see Innocenti and Price (2005).

As the critical period gradually ends, the excessive axonal branches created by the exuberant arborization phase are extensively pruned in response to experience-driven activity. At this point, axonal branches begin to approximate what they will look like for most of the mature phase of development.

3.4.4 Adult Axonal Plasticity

Axonal plasticity in the adult brain is significantly slower than in young brains (Qiao et al., 2016). Axonal growth and pruning is not eliminated after the developmental phases of a cortical area, but happens far less frequently under normal circumstances. In particular, the rate of growth and elimination of axonal branches in adults decreases and takes place on a longer time scale (Marik et al., 2010; De Paola et al., 2006). These reductions in plasticity are mirrored by cortical map stability at this point in an organism's life span. At this point, manipulations such as removing the input from one eye will no longer affect cortical map structures as greatly as they would during the critical period.

3.5 Related Model Review

To our knowledge, no existing abstract neural network models implement biologically motivated axonal dynamics driven by genetics or environmental experience. More than two

decades ago, (Quinlan, 1998) raised many of the same concerns addressed here with the models at the time. Quinlan (1998) examined neural networks that changed their structure over the course of learning. In particular, Quinlan (1998) emphasized models that grew and pruned connections. In modern terms, these networks had specific rules for determining when a connection should become sparse or not. Typically, these networks either started with random sparsity or full connectivity, then tentatively grew or removed connections. Simple decision rules pruned or grew connections based on how important they deemed a connection to the weight learning process. This way of modeling implicitly describes systems in which synaptic plasticity completely controls axonal plasticity. This contradicts the bounty of neuroscience evidence regarding axonal development.

Today, abstract neural networks often use combinations of weight decay and sparsity inducing objective functions to drive small weights in a network to zero. Additionally, it has been shown that using dropout and Gaussian noise on unit activation tends to push unnecessary weights to zero (Srivastava et al., 2014). Dropout and additive noise methods can cause the features learned in fully-connected networks to become local in nature, producing a form of sparsity reminiscent of early visual cortex (Srivastava et al., 2014). It is important to note that all of these methods generally cause neural networks to learn and converge several times slower than networks that do not use these methods, even though these approaches usually achieve better generalization performance. These current techniques still describe systems where the dendritic qualities of plasticity completely control all aspects of axonal plasticity.

The most recent advancement in neural network architectures that incorporates knowledge of cortical axonal structure was the development of convolutional neural networks (CNNs) (Fukushima, 1980). Modern versions of CNN's have dominated the field of computer vision, starting with the architectures introduced by Krizhevsky et al. (2012). These architectures have demonstrated that hard-wired sparsity patterns, that are appropriate to the domain of the inputs, combined with weight sharing, can greatly increase the effectiveness of neural networks compared to existing fully connected or random-sparsity networks.

Although they are useful for modeling the brain in certain contexts, CNNs do not properly account for many important aspects of axonal development. Primarily, CNNs typically use a completely fixed window structure. Sparsity-inducing methods are often applied to modern CNNs, but this is typically implemented by pruning an already-severely-limited set of initially available connections. Further, the structure of these initial connections is in no way influenced by the model's experience. This fixed-window structure may be acceptable for modeling early visual cortex, as the gross features of axonal development are mostly developed prior to obtaining visual experience. However, axonal development in higher visual cortex presumably depends far more on post-natal experience (for review, see Innocenti and Price (2005)), which would have compounding effects up the hierarchy of representation. Additionally, the weight sharing used by CNNs does not relate plausibly to feature learning in human cortex (Yamins and DiCarlo, 2016a). Actual cortex is thought by most neuroscientists to learn features through local, competitive interactions that are better illustrated in a cortical map context. These issues act as limiting factors on the appropriateness of using CNNs to model human cortical representations.

Axonal learning and development has been extensively modeled in the field of computational biology (Fraser and Perkel, 1990; Gebhardt et al., 2012; Godfrey et al., 2009; Simpson and Goodhill, 2011). Within this sub-domain, simulations typically emphasize how pre-synaptic axonal arbor maps form rather than the interactions with post-synaptic learning. As time has progressed, these models have become increasingly biologically detailed. Models that do feature both pre-synaptic and post-synaptic development tend to be more focused on small-scale interactions (Godfrey et al., 2009).

The model proposed by Fraser and Perkel (1990) is the seminal multi-factor axonal development model. Further it is a relatively abstract model that introduces many of the fundamental dynamics that later models would further elaborate. In their model, Fraser and Perkel (1990) treated axonal arbors as disks which move about a pre-synaptic cortical

sheet in order to minimize a fitness function composed of a weighted sum of tissue adhesion (C), arbor competition (R), activity dependent attraction (N), dorsal-ventral adhesion (DV), and anterior posterior adhesion (AP). Within this relatively simple simulation framework, many previously irreconcilable neuroscience results fit together neatly (Benson et al., 2001; Fraser and Perkel, 1990). Later models greatly expanded the detail of these forces, but many feature similar designs (Benson et al., 2001).

3.6 The Axon Game

In order to make axonal development dynamics accessible to the broader neural network modeling community, we created a model called the axon game. The essential concept of the axon game is that there is restricted connectivity between two layers in a neural network, and that the structure of this connectivity is directly determined by the axonal dynamics within an axonal development simulation. Fundamentally, the axon game takes the classic (Fraser and Perkel, 1990) multi-factor design and adapts it into a cellular automata, where individual axon branchlets grow and prune on a grid that represents a pre-synaptic sheet of blocks that belong to the dendritic arbors of post-synaptic neurons (see Fig. 3.1).

In more detail, axonal arbor branchlets can grow new branchlets based on a neighborhood function and can prune depending on their overall fitness within their axonal arbor. This fitness is derived from local information, such as the number of competing axon branchlets in the grid square (R), whether it is within a target region (C), the correlation or co-activation of an axon with other axon branchlets in same grid square (N), and the density of chemo-affinity signals (AP & DV) on the surrounding neighborhood of grid squares. Because of the local nature of the fitness rules, the axon game can be implemented with fast serial or parallel optimization in mind.

There are several possible ways to optimize the model's objective(s) or fitness function(s), such as using a Monte-Carlo Markov Chain method, simulated annealing, or an evolutionary

algorithm. For our purposes, we used a special kind of evolutionary algorithm where each axonal arbor is treated as its own population, and every branch within an arbor is given an individual fitness relative to the other branches in the arbor. This allows rapid parallel optimization. With this design, the Axon Game is intended to be an accessible but powerful tool for incorporating well-accepted axonal development dynamics into contemporary neural network architectures. The code that generates the demonstrations for this paper can be downloaded at `https://bitbucket.org/jryland/axon-game-paper-repo`. The model is implemented in written in python and makes extensive use of the TensorFlow package. For speed, we recommend using GPU acceleration. Next, we will discuss the Axon Game in more rigorous detail.

3.7 The Axon Game Algorithm

3.7.1 Branchlet Growth

In this version of the axon game, each branchlet of every axonal arbor randomly chooses one neighboring dendritic block to grow a new branchlet into during each update step. In some situations, it can be useful to allow only a portion of the existing branchlets to grow new branchlets. The fitness of each proposed new branchlet is evaluated afterwards in the pruning phase. Proposed branchlets are automatically deleted if they are either in invalid areas or are redundant. All valid proposed branchlets are immediately added to both their respective axonal arbor branchlet lists and their respective dendritic arbor branchlet lists. An axonal arbor branchlet list for a pre-synaptic neuron is the list of all of the axonal branchlets that project from that neuron to the post-synaptic sheet. A dendritic arbor branchlet list is the list of all the axonal branchlets within a dendritic arbor block associated with a post-synaptic neuron. A new branchlet represents a potential connection between a pre-synaptic neuron and a post-synaptic neuron and receives a small randomly initialized weight $w_a b$, for use in synaptic learning.

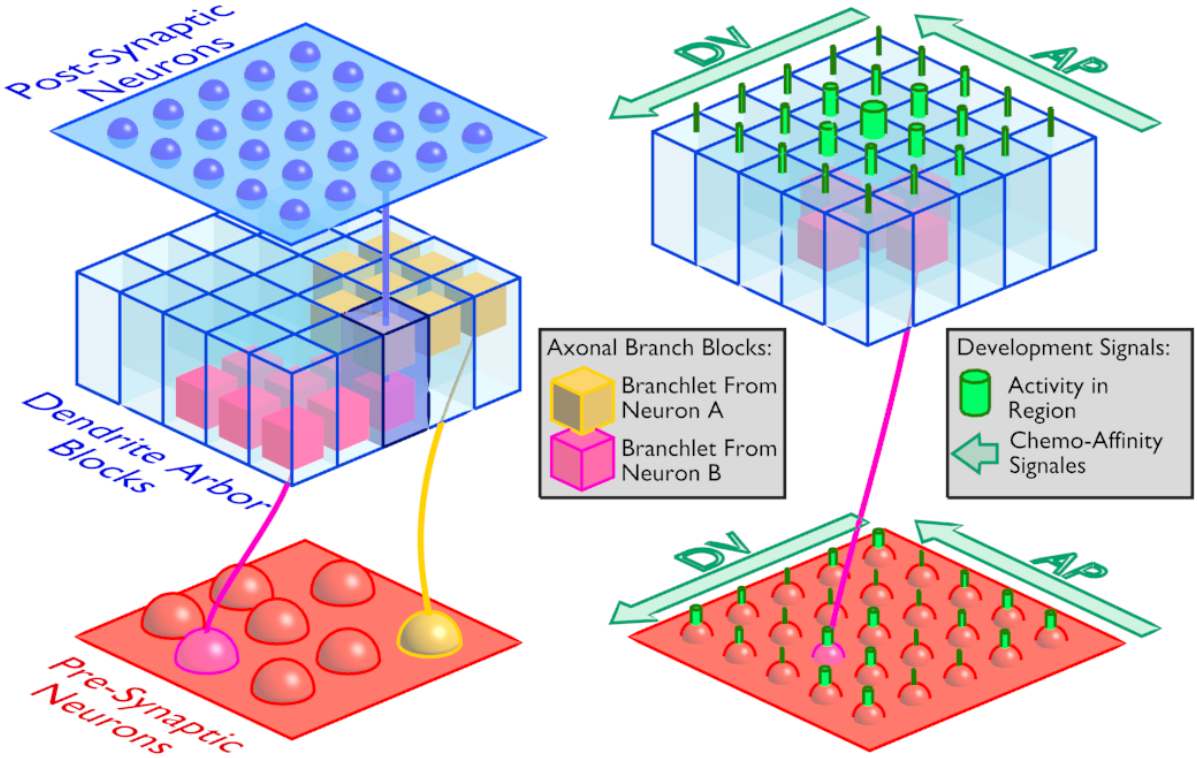

Figure 3.1. Axon Game Representation

This diagram illustrates how pre-synaptic neurons and post-synaptic neurons interface with one another using an axon game model to manage the connections between them. Each post-synaptic neuron has a dendritic arbor represented by the (dendritic arbor blocks). Each pre-synaptic neuron has an axonal arbor, which is a collection of individual axonal branchlets represented by the (Axonal Branch Blocks). The AP and DV axes indicate chemo-affinity signals associated with both the presynaptic sheet and the postsynaptic sheet. An axon branchlet inherits its AP/DV target from their originating pre-synaptic neuron. The summed branchlet activity in a region of dendritic arbor blocks is used to derive the activity-dependent plasticity (N).

Table 3.1. Axon Game Symbols

Symbol	Definition
a	arbor index
b	branchlet index
d	dendrite block index
t	time index
k	number of time steps of exuberant arborization
$ab+$	new branchlet proposed by branchlet ab
$\mathbf{x}_{ab}$	cortical position
$\mathbf{Y}^{pre}$	chemo-affinity pre-synaptic target
$\mathbf{Y}^{post}$	chemo-affinity post-synaptic target
c^{post}	border target
C	border penalty
R	competition penalty
N	co-activation reward
$APDV$	anterior posterior & dorsal-ventral chemo affinity penalty
S	seniority reward
w_{ab}	connection weight
$\alpha_?$	strength of reward or penalty
EX_t	exuberance
P_{grow}	proportion of branchlets that can propose
B	target branch number
η_{iu}	normalized pre-synaptic activity for neuron i observation u
σ_y^{diff}	diffusion of chemo-affinity signal
σ_n^{diff}	diffusion of regional neural activity

Branchlet growth is segregated into two phases. The first phase uses a 2D discrete Gaussian, with a standard deviation of EX_t, as a position-proposal function for new branchlets. A wide proposal function mimics the exuberant arborization seen in most mammal species early in early cortical development. In the second phase, the candidate proposal function is a simple Moore neighborhood around each branchlet, which promotes slow changes like those seen in mature cortical projections. Importantly as t progress from 0 to k, EX_t linearly interpolates from EX_0 to EX_1. Several other values are allowed to change in the same manner, such as P^0_{grow} and P^1_{grow}. Finally, the seniority reward can be raised gradually during the adult phase. For this, α_S is linearly interpolated between α^0_S and α^1_S as t progresses from k to t^{end}.

$$x_{ab} = \begin{cases} t < k : \tilde{g}\left(x_a b, EX_t\right) \\ t >= k : \widetilde{Moore}(x_a b) \end{cases} \tag{3.1}$$

3.7.2 Branchlet Pruning

In this version of the axon game, each projecting axonal arbor is updated in parallel using a simple axonal branchlet fitness function. Each branchlet is evaluated with the fitness function $f_a b$, and then all of the branchlets are ordered within an arbor according to their fitness values. The top B branchlets within an arbor will be kept, and all other branchlets will be pruned. The fitness function is defined in Eq. 3.2 using the notation in Table 3.1.

$$f_{ab} = \alpha_N N_{ab} + \alpha_S S_{ab} - \alpha_R R_{ab} - \alpha_C C_{ab} - \alpha_{APDV} APDV_{ab} \tag{3.2}$$

3.7.3 Algorithm Steps

1. Initialize with 1 randomly placed branchlet for each arbor
2. Loop:
   (a) Each branchlet proposes a new branchlet with probability P^t_{grow}
   (b) Valid candidates are added to respective arbor branch lists
   (c) Branchlets within each arbor, a, are ordered according to fitness measure f_{ab}
   (d) Keep top B branchlets within each arbor, prune all others
   (e) Increment t

3.7.4 Penalties & Rewards in Detail

Here we define the terms used in Eq. 3.2. To find qualities of branches in a dendritic block at a particular cortical location we will use the notation $Q_b(\mathbf{x})$, where Q_b is some quality of

a branch at location $\mathbf{x}$. If the notation is $Q(\mathbf{x})$, then this is a quality of the post-synaptic sheet or the dendritic block at location $\mathbf{x}$.

$APDV_{ab}$ is the retinotopic organizing force of the axon game model, as seen in Eq. 3.2. It combines an anterior/posterior positional signal (AP) and a dorsal/ventral signal (DV) across the simulated cortical sheet (see Fig. 3.1). Further, it combines a local continuity signal and target position signal. The superscript $diff$ for $\mathbf{y}^{post-diff}$ and $\mathbf{y}^{pre-diff}$ indicates that these values have been diffused by convolving them with a Gaussian kernel over the dendritic blocks with standard deviation σ_y^{diff}. The $APDV_{ab}$ penalty is divided into two sub-penalties; 1) a distance from the local average of the targets of the axonal branches themselves, $APDV_{ab}^{local}$, and 2) a branch's distance from its target defined by the receiving tissue's gradients, $APDV_{ab}^{global}$.

$$APDV_{ab}^{local} = \left\| \mathbf{y}_a^{pre} - \overline{\mathbf{y}^{pre-diff}}(\mathbf{x}_{ab}) \right\| \tag{3.3}$$

$$APDV_{ab}^{global} = \left\| \mathbf{y}_a^{pre} - \mathbf{y}_a^{post-diff}(\mathbf{x}_{ab}) \right\| \tag{3.4}$$

$$APDV_{ab} = \alpha_{global} APDV_{ab}^{global} + \alpha_{local} APDV_{ab}^{local} \tag{3.5}$$

From Eq. 3.2, N_{ab} is the activity dependent force in the axon game. $\bar{\eta}(\mathbf{x}_{ab})$ is a vector over time of average axonal branchlet activity at cortical block position x_{ab} that has been unit normalized across the cortical blocks. $\overline{\eta^{diff}}(\mathbf{x}_{ab})$ is the result of convolving $\bar{\eta}(\mathbf{x}_{ab})$ with a Gaussian kernel over the dendritic blocks with standard deviation σ_n^{diff}. This can be thought of as a diffuse version of the neural signal. The activity dependent force may be implemented as a co-activation (Eq. 3.6) or correlation measure (Eq. 3.7), depending on the input assumptions. Typically the co-activation method is more appropriate when only positive activations are allowed, whereas the correlation method would be more appropriate in the general case.

$$N_{ab} = \left|\bar{\eta}(\mathbf{x}_a b) \cdot \overline{\eta^{diff}}(\mathbf{x}_{ab})\right| \tag{3.6}$$

$$N_{ab} = \left| \text{Corr}\left(\bar{\eta}(\mathbf{x}_a b), \overline{\eta^{diff}}(\mathbf{x}_{ab})\right) \right| \tag{3.7}$$

From Eq. 3.2, R_{ab} is the competition force in the axon game that encourages the branchlets to spread out across the arbor blocks.

$$R_{ab} = \#\text{branchlets at } \mathbf{x}_{ab} \tag{3.8}$$

From Eq. 3.2, C_{ab} is the border force of the axon game. This border force encourages the axon game to limit branchlets to specific regions of the dendritic block grid where c^{post} is equal to one.

$$C_{ab} = c^{post}(\mathbf{x}_a b) \tag{3.9}$$

From Eq. 3.2, S_{ab} is a seniority force in the axon game. The default definition of S_{ab} starts at 0, when a branch is created, and is incremented for each time-step it is alive. S_{max} is the maximum number of increments allowed for the default S_{ab}.

$$S_{ab}(t+1) = S_{ab}(t) + 1 \tag{3.10}$$

Alternatively, S_{ab} can be defined as a running average of the N_{ab} reward, with an update speed of β_s that initializes at zero when a branchlet is created.

$$S_{ab}(t+1) = S_{ab}(t)(1-\beta_s) + N_{ab}(\beta_s) \tag{3.11}$$

3.7.5 Important Variations

In some instances, we found it useful to modify the basic update rules or the principal representations used in the Axon Game model. Some useful modifications are listed below.

Plexing: When representing larger cortical map projections with larger receptive windows, computation can be reduced by running the Axon Game with a reduced dendritic arbor block sheet. The dendritic arbor branchlet lists for each arbor block can be duplicated to form a mapping that is a multiple of the dimensionality of the original sheet. This representation is referred to as *plexing*, or a *plexed* Axon Game. Importantly, the duplicate branchlets need their own independent weights to be maintained during the simulation if axon learning and synaptic learning need to happen in parallel.

Dendritic Windows: When simulating larger cortical maps that require large input windows for the post-synaptic neurons, it can be advantageous to allow each post-synaptic neuron to connect to the branchlets of a window of dendritic blocks around the neuron's cortical location. This simultaneously allows for the manipulation of the scale of pre-synaptic maps and post-synaptic maps independently. This also allows for a large decrease in the overall amount of computations by using a reduced number of branchlets per pre-synaptic neuron to produce the same window sizes. Unfortunately, dendritic windows also create duplicate connections between pre- and post-synaptic neurons, which makes representing the weights much more difficult. As such, our implementation generates a random set of weights for each unique connection when dendritic window larger than one is used. From a biological perspective, dendritic windows would be analogous to the size of dendritic arbors for cortical neurons.

Co-Activation vs. Covariance: Depending on the input activation types, the activity-dependent reward can be based on co-activity or a covariance like measure. A co-activity reward is more appropriate when the pre-synaptic activations are restricted to be positive.

Squared correlation or the absolute value of covariance may be more appropriate when the pre-synaptic activations are real valued.

Seeding: Researchers have suggested that some amount of rough topography may be present in a cortical area due to how the fibers of the projecting area will tend to travel in parallel, this tendency is called *fasciculation* (for review see Benson et al. (2001)). For simulations of smaller amounts of cortex, it may make sense to seed the initial topology with additive noise. When seeding a map, we introduce the parameter σ_{seed}, which adds Gaussian noise to a seeded starting position at the minimums of each axon arbor's $APDV$ target. Seeding could also be interpreted as the result of a period of arbor development that is largely activity independent.

Temporal Smoothing: For certain situations, it may be useful to blur the pre-synaptic activations across time. This would allow windows to optimize co-activation or correlation over a smoothed temporal window.

3.8 Simulated Axonal Maps

In this section, we will qualitatively illustrate the axonal projection maps formed by the axon game and how they arise using a series of demonstrative simulations with specific parameter settings. Some of these results can be viewed as replications of previous simulation results from Fraser and Perkel (1990). Further, the development of retinotopic projections have been modeled robustly in the axonal modeling literature. To the best of our knowledge however, the simultaneous combination of maps simulated here are novel.

For this simulation, we calibrated the parameter settings to cause the Axon Game to mimic a relatively large portion of a V1 map. The main parameters of interest are the initial exuberance size EX_0 relative to the size of the cortex in units, and the branch depth of the cortex. We set the EX_0 to 1, and the branch depth was 8. The simulated map resolution was 170 by 250 simulated dendritic blocks. The algorithm was run for 300 steps, 200 of

which were in the exuberant growth phase and 100 of which were in the adult phase. For the exact model settings see section 3.11.

The goal of this simulation was to demonstrate that visual field maps, ocular dominance maps, and spatial frequency maps can arise under the same parameters within the axon game. To achieve this, we ran multiple simulations under different parameter settings to find values that produced all of the relevant pre-synaptic maps simultaneously and reliably. The results of the particular parameter settings shown here are not a characterization of a specific species' visual pathway. However, the parameters of the axon game have direct axonal and neural analogues that researchers could adjust to fit known information about a particular species in order to conduct more specific simulation experiments, rather than simple demonstrations.

3.8.1 Methods

3.8.1.1 Stimulus Generation

We generated synthetic stimuli that were 1600 by 1600 pixels in size. Each stimulus contained 32 Gaussians, randomly placed such that each spatial frequency sampled by a foveal representation would be statistically invariant from the other spatial frequencies. This was achieved by generating each Gaussian at a specific scale, making the allowable region for the random center to be proportional to the scale of the Gaussian, and manipulating the intensity. The aspect ratio of the Gaussians was 1 by 4, with the longest dimension pointing along a random direction. In total, 2000 stimuli were generated for this simulation.

3.8.1.2 LGN-like Input Representation

The Lateral Geniculate Nucleus (LGN) is one of the primary waystations on the route from the retina to visual cortex and will be treated as the input to our model. To mimic the spatial-frequency-sensitive (i.e., center-surround) inputs of LGN into V1, we created a

simple Laplacian image pyramid by using the standard differences of Gaussians and down-sampling method (Burt and Adelson, 1983). Further, to mimic the foveated nature of visual inputs, we also cropped each scale level of the pyramid to have a constant size of 50 by 50. Each pixel represents an on-off center-surround cell. Unrectified on-off cells were sufficient here, but for synaptic-learning rectified on-off and off-on center-surround cells would be more appropriate. The initial input images are 1600 by 1600, which produces 6 scale levels where each successive scale level looks at a central crop that is 1/2 the size along each dimension as the previous scale level. These values were duplicated, and the duplicates were weighted by the random uniform values $\tilde{u}_1$ and $\tilde{u}_2$ such that $\tilde{u} \in [-.5, 1]$. The randomly weighted duplicates are intended to simulate the imperfect correlation between the receptors of the left and right eye due to stereopsis and spontaneous retinal activity. Figure 3.2 depicts the different scale levels for a stimulus viewed by one eye.

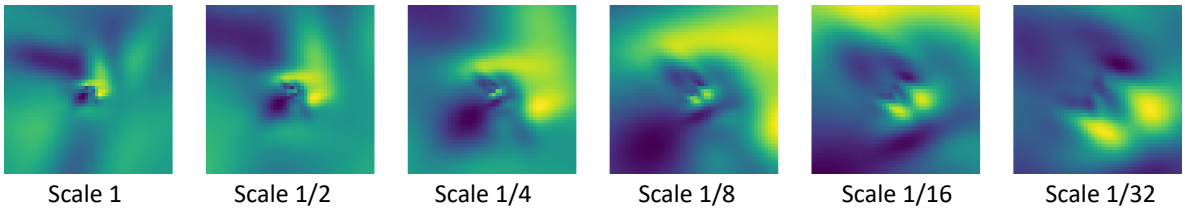

Figure 3.2. Foveated LOG Image Pyramid
This figure shows an example of one of the synthetic stimuli processed by different levels of the LOG image pyramid. From left to right the images depict the largest to smallest scale levels.

3.8.1.3 Chemo-Affinity Gradient Specification

For these demonstrations we tagged the incoming axonal projections by their radius and polar angle from the original visual representation. We tagged the cortical map using simple linear coordinates for the vertical axis and exponential coordinates for the horizontal axis. Each set of tag coordinates were normalized between the values of 0 to1. Further, only the left

visual field was included in the simulation and the boundary was set to be a simplified V1-like shape. The chemo-affinity tags for the projecting axons associated with visual eccentricity were calculated to evenly distribute the axons across the receiving cortical sheet. Because this is a large-scale simulation, we seeded the initial locations of the axonal projections to match their chemo-affinity tags as best as possible.

3.8.2 Simulation Results and Discussion

3.8.2.1 Visual Field Map Development

Examinations of different species have shown that multiple areas along the visual cortical hierarchy, including V1, contain retinotopic organization. For review, see Wandell et al. (2007). Importantly, the mapping from visual angle to cortical representation is non-Euclidean. The fovea receives far more representation than the peripheral field, and the mapping from visual eccentricity to cortical position is well approximated by an exponential function (Dougherty et al., 2003; Engel et al., 1997; Qiu et al., 2006). Figure 3.3 shows an example of real human visual-field maps in V1, while Figures 3.4 and 3.5 show the visual-field maps developed by the axon game under these settings. Importantly, chemo-affinity is the main driver in the creation of these maps, and the maps will develop with or without the presence of neural activity.

3.8.2.2 Spatial Frequency Map Development

V1 cortical cells tend to respond selectively to stimuli containing specific spatial frequencies. Further, although there is contention, experiments have suggested that spatial frequency is organized in a continuous topological fashion on the cortical sheet (Issa et al., 2000; Ribot et al., 2013). At a fine-grained level, this topology appears patchy. However. at a larger scale, this topology appears to be related to visual eccentricity (see Fig. 3.6), something that would be expected given the foveated inputs received by V1. Figure 3.6 shows an example of

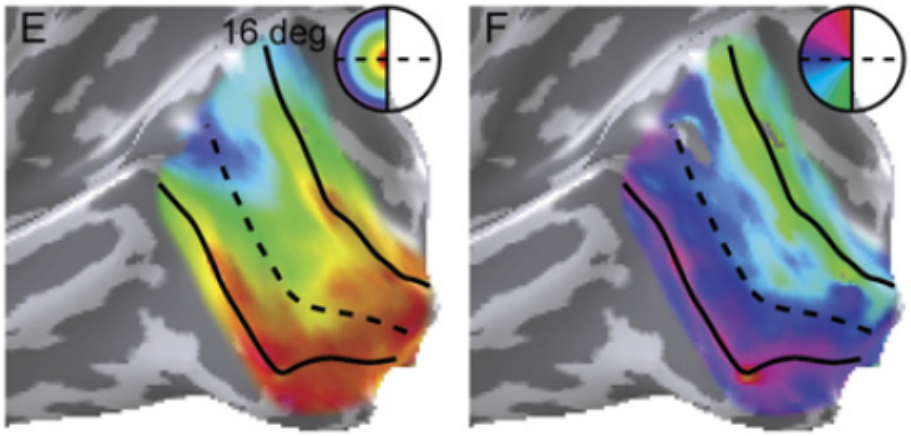

Figure 3.3. Human Visual Field Maps
This figure shows fMRI measurements of the retinotopic organization of V1. Note that the area of representation of foveal view regions is massively larger than the actual area of the visual field it represents. Reprinted with permission from (Wandell et al., 2007).

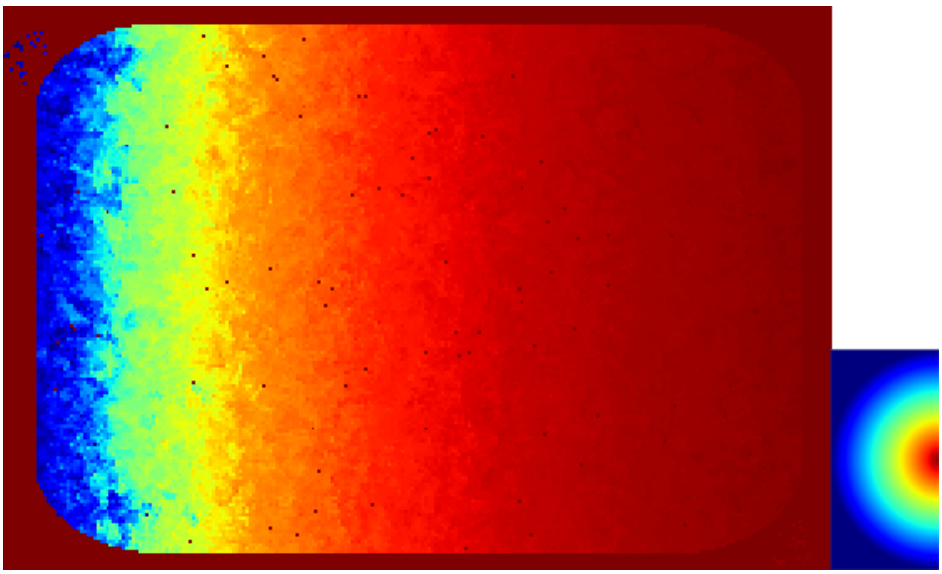

Figure 3.4. Axon Game Visual Field Eccentricity
This figure shows an example of the polar eccentricity maps that develop in the axon game under this demonstration's settings.

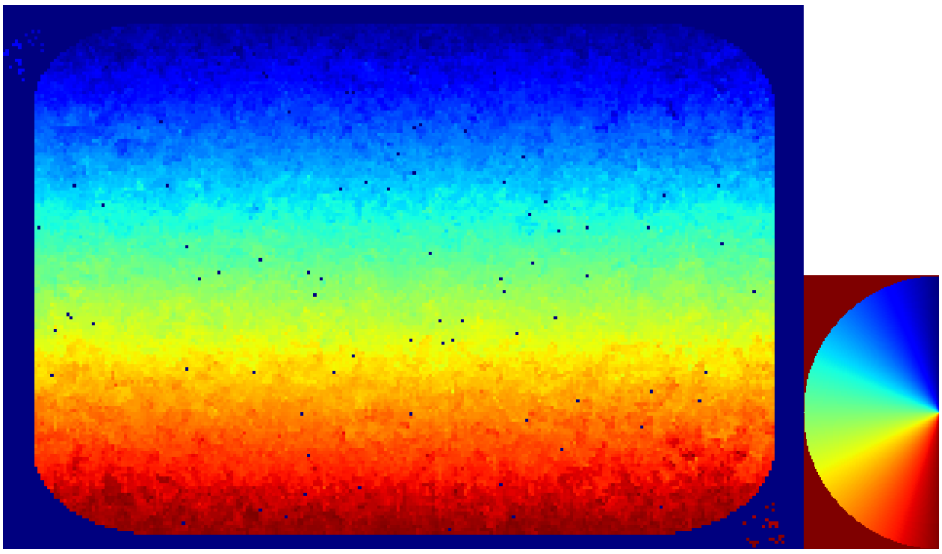

Figure 3.5. Axon Game Visual Field Angle
This figure shows an example of the polar angle maps that develop in the axon game under this demonstration's settings.

a real spatial-frequency map, while figure 3.7 shows an example of a spatial-frequency map developed in the axon game under our experimental settings.

3.8.2.3 Ocular Dominance Map Development

Early examinations of monkey brains identified cortex in V1 that showed bands of neurons, innervated preferentially by one eye or the other, spaced about .2-.4mm apart (Hubel and Wiesel, 1972; Le Vay et al., 1980; LeVay et al., 1975). These bands are referred to as ocular dominance columns and are shown to alternate in a particular kind of stripe pattern that was first pictured by LeVay et al. (1975) (see Figure 3.8).

Our example simulation produces ocular dominance columns similar to those seen in primates and other species, as seen in Figure 3.9. The size and spacing of the ocular dominance columns is highly dependent on the nature of the inputs (i.e. the size of the on-center

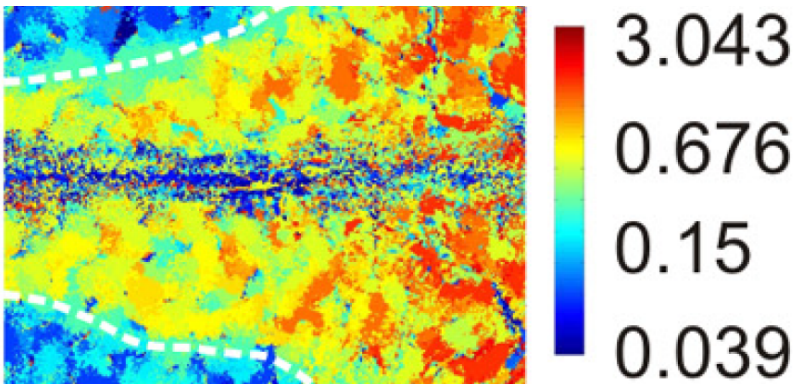

Figure 3.6. Cat Spatial Frequency Map
This figure shows a spatial frequency map for cat V1. Reprinted with permission from (Ribot et al., 2013). The red indicated high-frequency sensitive neurons that tend to be more represented near the fovea and the blue indicates low-frequency sensitive neurons that tend to be represented more towards the periphery. Note the patchy gradient from high frequency representation to low-frequency representation.

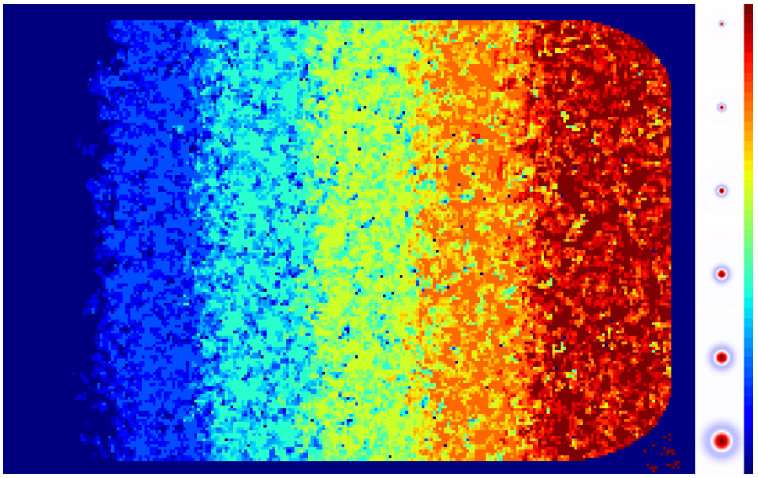

Figure 3.7. Axon Game Spatial Frequency Map
An example of the spatial frequency maps that develop in the axon game under this demonstration's settings. The red indicated high-frequency sensitive neurons that tend to be more represented near the fovea and the blue indicates low-frequency sensitive neurons that tend to be represented more towards the periphery. Note that the axon game simulation shows a similar sort of patchy gradient from high-frequency to low-frequency representation across the V1 surface.

off-surround cells, and their resolution across a scale) and the size of the axonal arbors. The qualities of the ocular dominance columns also depend on developmental parameters, such as the width of initial exuberance EX_t and the relative weight placed on the activity dependent force α_N.

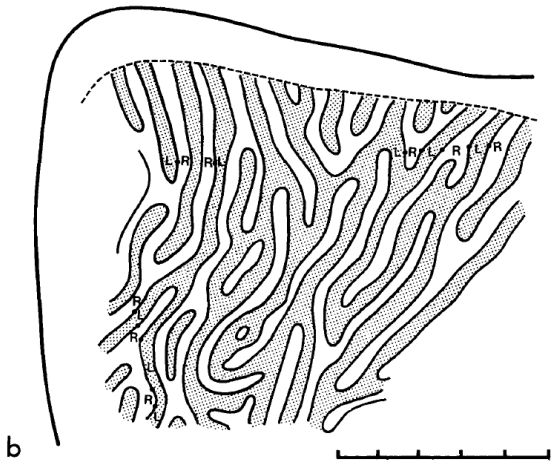

Figure 3.8. Macaque Ocular Dominance Columns
An example of real ocular dominance columns sampled from macaque visual cortex. The ticks on the scale reference line at the bottom of the figure are in intervals of 5 mm. Reprinted with permission from LeVay et al. (1975). Note the long stripe like patterns black and white that represent neurons sensitive to the left and right eye respectively.

3.8.2.4 Successes & Limitations

The version of the axon game that produced these maps simultaneously is loosely based on the earlier, less detailed model described by (Fraser and Perkel, 1990). If researchers wish to use the axon game to study more specific aspects of axonal development with more biological detail, they may wish to adapt concepts from more recent models of axonal learning.

Despite the simplicity of the model, the axon game still develops all of the classic cortical maps that, in theory, can be defined by axonal dynamics alone. To our knowledge, this is the first time the visual field, binocularity, and spatial frequency maps have been generated in a pure axonal development model under a single set of parameters and settings.

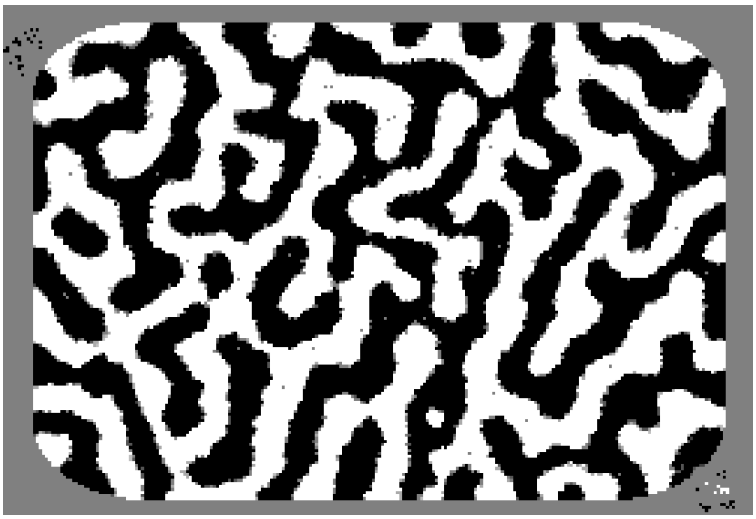

Figure 3.9. Axon Game Ocular Dominance Columns
An example of ocular dominance columns that develop in the axon game under these parameter settings. Note that the axon game simulation of V1 develops similar long stripe like patterns of eye sensitive neurons to that of real visual cortex.

3.9 Simulated Axonal & Neuronal Feature Maps

Previous dendritic models of cortical map development have produced post-synaptic cortical maps that resemble ocular dominance columns and orientation maps (Sirosh and Miikkulainen, 1994; Stevens et al., 2013; von der Malsburg, 1973). In this section, we demonstrate that a model that accounts for the interaction between axonal and synaptic plasticity can exhibit qualitatively similar cortical map development. This is by no means the first simulation model to contain both axonal and synaptic dynamics, as this is done routinely in the fields of computational neuroscience and computational biology (Godfrey et al., 2009). However, the axon game is designed to make integrated dendritic-axonal dynamics more generally accessible to the wider neural network audience, including the cognitive and cognitive-neurosciences. Correspondingly, the axon game is designed to facilitate larger-scale phenomena that require

higher data-throughputs, such as hierarchical cortical map development and the development of behaviorally functional feature hierarchies.

In order to demonstrate axonal and synaptic dynamics working in conjunction, we paired the axon game with an appropriate model of synaptic learning. Many popular models of cortical learning exist that have specific characteristics of the cortex they emphasize. The Laterally Interacting Synergetically Self-Organizing Map (LISSOM) family of models (Sirosh and Miikkulainen, 1994) emphasizes biologically inspired Hebbian learning, lateral interactions and cortical map learning. Models like LISSOM generally assume either a fixed, un-learnable axonal structure, or unrealistic fully synapse driven axonal development. Here we show how more realistic synaptic and axonal learning dynamics can be simulated at the same time, by using the Axon Game paired with a variant of the LISSOM framework. For this demonstration, we model projections from the LGN to V1, where the axon game determines how LGN projects to V1 and a novel variant of LISSOM we created called H-LISSOM simulates neural representation learning for V1.

3.9.1 H-LISSOM Overview

LISSOM is a model of cortical map development and feature-learning (Sirosh and Miikkulainen, 1994). The LISSOM model makes use of several main principles, including Hebbian weight updates and lateral connections that are both excitatory and inhibitory. The range of the allowable excitatory connections is usually much smaller than the range of allowable inhibitory connections. In LISSOM-like models, both the input weights and the lateral connections are typically learnable. However, sometimes both the lateral excitatory and inhibitory connections are held fixed. Our novel variation of LISSOM is called Homeo-LISSOM, or H-LISSOM, because it includes additional homeostatic dynamics designed to make it easier to use. The main symbols used to describe H-LISSOM's dynamics can be found in Table 3.2.

Table 3.2. Axon Game Symbols

Symbol	Definition
d	dendrite block index
η_d	activity of neuron d on cortical sheet
η_a	activity of input neuron via axon arbor a
w_{di}	weight associated with i-th branchlet connected to cortical neuron d
ω_{di}	the activity of the i-th branchlet connected to cortical neuron d
$\mathbf{x}_d$	cortical location of neuron d
σ^E	spread of excitatory lateral connections
σ^I	spread of inhibitory lateral connections
$\bar{A}$	estimated average length of cortical input window activities
$\bar{U}$	estimated activity given random weights
ϵ	values near this will not be normalized
θ_d	activation bias
reps	number of recurrent activation steps

The main activation function of H-LISSOM η_d is similar to using LISSOM with a Relu function that modifies a weighted summation of the current input to a neuron, d, plus the neuron's incoming lateral excitation E_d and lateral inhibition I_d. We found that running versions of H-LISSOM with either fixed or non-fixed lateral connectivity resulted in similar maps, so for efficiency we used fixed Gaussian lateral connectivity. It should be noted that although learned lateral connectivity resulted in sharper transitions across the simulated cortex, it greatly increased training time. LISSOM models are recurrent networks in that the activation function will be calculated multiple times, taking lateral interaction as new inputs. For H-LISSOM, we only update the weights after a fixed number of recurrent activations that we refer to as reps. The following equations specify the H-LISSOM model, using the notation in Table 3.2.

$$\hat{\omega}_{di} = \frac{\omega_{di}}{\sqrt{\sum_i (\omega_{di})^2 + \epsilon \bar{A}}} \tag{3.12}$$

$$\eta_d(t+1) = \text{ReLu}\left(\sum_i \omega_{di} \text{w}_{di} - \theta_d + E_d(t) - I_d(t)\right) \tag{3.13}$$

$$E_{d_1}(t) = \sum_{d_2} \eta_{d_1} g(\mathbf{x}_{d_1} - \mathbf{x}_{d_2}, \sigma^E) \tag{3.14}$$

$$I_{d_1}(t) = \sum_{d_2} \eta_{d_1} g(\mathbf{x}_{d_1} - \mathbf{x}_{d_2}, \sigma^I) \tag{3.15}$$

$$\text{w}_{di}(t+1) = \frac{\text{w}_{di} + \delta_\text{w} \hat{\omega}_{di}}{\sqrt{\sum_j (\text{w}_{dj} + \delta_\text{w} \hat{\omega}_{dj})}} \tag{3.16}$$

The main difference between the LISSOM model and H-LISSOM is that it includes a homeostatic activation bias, θ_d. This activation bias is similar to that of both the GCAL variation of LISSOM (Stevens et al., 2013), and a soft normalization of the length of the input window activations. In a further change to the GCAL model, the θ_d bias term is set to hit a target, $u\bar{U}$, which is proportional to the statistic, $\bar{U}$. This $\bar{U}$ statistic estimates the overall activity that cortical neurons would be expected to have if the weights of each neuron were randomly selected from the unit sphere with a bias of 0, $\hat{\text{w}}_i$. This expected value is estimated by a running average. In addition, the soft normalization target α is modified by $\bar{A}$, which is the estimated average length of the non-zero activations coming into the cortical neurons through their input windows. This can be thought of as adjusting what the network considers to be noise, given prior activations. To accomplish this, $\bar{A}_d$ only updates when neuron d currently has a window with a non-zero activity.

$$\theta_d(t+1) = \theta_d(t) - \delta_\theta(\bar{\eta_d} - u\bar{U}) \tag{3.17}$$

$$\bar{\eta}_d(t+1) = (1 - \beta_\eta)\bar{\eta}_d(t) + (\beta_\eta)\eta_d(t) \tag{3.18}$$

$$\bar{U}(t+1) = (1-\beta_u)\bar{U}(t) + \frac{\beta_u}{D}\sum_i \left(\hat{\omega}_{di}\hat{\tilde{\mathbf{w}}}_i\right) \tag{3.19}$$

$$\epsilon(t+1) = \epsilon(t) - \delta_\epsilon(\epsilon(t) - \alpha\bar{A}) \tag{3.20}$$

$$\bar{A}_d(t+1) = (1-\beta_a)\bar{A}_d(t) + \beta_A\sqrt{\sum_i (\hat{\omega}_{di})^2} \tag{3.21}$$

The additional homeostatic dynamics discussed above are meant to alleviate some of the difficulties of calibrating LISSOM-like models. LISSOM models are not typically able to handle variable intensities and often require hand-tuning to get consistent results. We address this unpredictability by soft-normalizing the inputs to each neuron, ensuring that input patterns beyond a certain magnitude will saturate with a length of 1 and that patterns below a certain magnitude will be ignored in a graded fashion. In addition, GCAL-like target activity levels are often difficult to set when the overall activation levels of the network are not known beforehand. H-LISSOM addresses this by estimating what the overall activation levels would look like without any learning, as well as by enforcing the target activity level to be a fraction of that value, ensuring that the selectivity of each cortical neuron narrows over time.

3.9.2 Methods

3.9.2.1 Stimulus Generation

For the creation of our stimuli, we used the same random Gaussian method as used in the first simulation. However, for this simulation the aspect ratio of the Gaussians was set to be 1 by 8. This led to better columnar organization in the orientation map.

3.9.2.2 LGN-Like Input Representation

The input representation used in this simulation modeled the type of center-surround cell processing seen in LGN neurons. Unlike the first simulation, however, this input representation only featured center-surround cells that were calibrated to respond maximally to a specific spatial frequency. Two random offset versions of each image were generated to represent input from each eye. The input had a resolution of 50 by 50 on-off cells and 50 by 50 off-on cells. Approximately 10,000 images were generated for this simulation.

3.9.2.3 Axon Game Parameter Settings

To encourage the growth of larger input windows, we increased the relative fidelity of the axon game by tweaking the growth parameters in the model. This axon game was on a 100x100 grid and featured much larger axonal arbor sizes. We also increased the exuberance and the activity-dependent reward weight. Further, we used simple linear AP and DV gradients for the chemo-affinity targets. For the exact settings, see section 3.12.

3.9.2.4 Interaction between Axon Game and H-LISSOM

The axon game supports continuous, simultaneous learning with H-LISSOM, but can only do so when the dendritic arbor radius is set to 1. The quality of the neuronal windows was higher when the dendritic arbor radius was larger than 1, so we chose to forgo simultaneous learning in favor of higher-quality windows. As such, we chose to have the Axon Game model learn its connections first to create a window structure, and then passed this structure to the H-LISSOM model for use in synaptic learning. Given that axonal arbor plasticity is vastly reduced in adulthood and during the critical period post-gestation, we expected that sequential development of axonal and synaptic dynamics would still capture many important aspects of the interaction between axonal maps and synaptic maps.

3.9.3 Results and Discussion

3.9.3.1 Orientation Map Development

Cells in V1 tend to be selective for specific orientations of contrast gradients. The organization of this selectivity in the cortex is mostly continuous across orientation. Researchers have discovered large, columnar regions of V1 that respond to the same orientations and dubbed them orientation dominance columns (Hubel et al., 1978). The global organization of these columns is often called an orientation map. These orientation maps tend to contain singularity points called pinwheels, where multiple orientation dominance columns come together and terminate (Blasdel, 1992; Bonhoeffer and Grinvald, 1991). These pinwheels may be a requirement for the development of an efficient orientation map (Durbin and Mitchison, 1990). See figure 3.10 for an example of a real V1 orientation map.

Orientation maps are a common milestone for models of cortical learning, both for abstract as well as more biologically motivated cortical models of V1 (Goodhill and Cimponeriu, 2000; Miikkulainen et al., 2005; Stevens et al., 2013; von der Malsburg, 1973). However, all of these models either assume a fixed connectivity, or that connections will be determined by random exploration and pruning based on synaptic learning principals. Importantly, these models have been very successful in demonstrating many phenomena of interest, but none have used a biologically plausible set of axonal learning dynamics.

Here we demonstrate that simulating axonal and synaptically driven map development is feasible with the axon game and the H-LISSOM model. Fig. 3.10 shows an example of a real orientation map, while Fig. 3.11 shows an orientation map developed by the Axon Game and H-LISSOM. Our simulation displays both orientation dominance columns and pinwheels, as seen in Fig. 3.11. To our knowledge, this is the first model to use biologically motivated axonal and synaptic principals to learn both pre-synaptic and post-synaptic cortical maps for V1.

While this test is just a demonstration of feasibility, there are many interesting applications of such a model. For instance, it would be useful to understand under what conditions the maps of binocularity and orientation become orthogonal in this model. Orthogonality happens when the contours between separate cortical maps intersect at steep angles, such as orientation and ocular dominance maps. Orthogonality promotes good feature coverage between two or more feature maps (Blasdel, 1992; Issa et al., 2008). Further, combining models of axonal development, simple cell learning, and complex cell learning, could allow researchers to build multi-layer cortical models without requiring biologically implausible window assumptions.

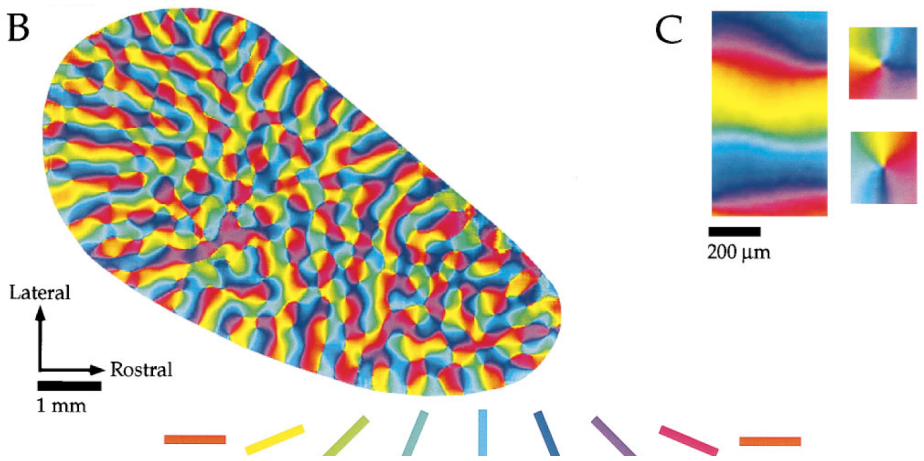

Figure 3.10. Axon Game Ocular Dominance Columns
B shows a real V1 orientation map sampled from a tree shrew brain. The small squares in C show examples of pinwheels. Reprint with permission from (Bosking et al., 1997).

3.9.3.2 Map Decoupling: A Novel Prediction

While testing the axon game and H-LISSOM combination model, we observed an interesting and possibly new theoretical phenomenon. In order to discuss this finding, some methodological clarification is required.

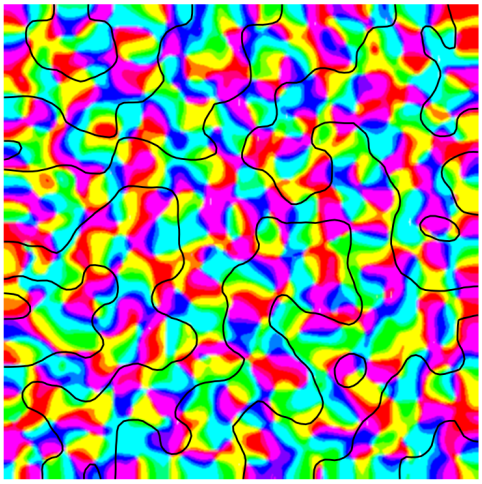

Figure 3.11. Axon Game Ocular Dominance Columns
Shows an overlay of an orientation map and an ocular dominance map spontaneously learned by simulating axonal and synaptic learning, using the axon game and H-LISSOM. This map was developed with dendritic radius of 4. The colors indicate regions that respond to the same directionality of edges, and the black lines indicate the transitions between left and right eye sensitive ocular dominance columns. Note the presence of pinwheels and strips of iso-orientation selective regions similar to those seen in actual V1 cortex.

Cortical map measurements can be broadly segregated into two categories. Some methods measure cortical maps as axonal arbor organization (*pre-synaptic maps*) and some methods measure receptive fields (*post-synaptic maps*). A reasonable assumption of axonal development research and cortical map research is that pre-synaptic measures and post-synaptic measures of cortical maps largely measure the same structures. This assumption seems valid given that ocular dominance columns, spatial frequency maps, and visual field maps tend to have similar qualitative characteristics, regardless of whether they are measured as pre-synaptic axonal arbor organization or post-synaptic receptive fields. However, our simulations suggest this may not always be the case.

During our simulations of simultaneous pre-synaptic and post-synaptic development, we discovered that some of our model's pre-synaptic and post-synaptic cortical maps can be completely uncorrelated with one another despite having similar qualitative characteristics

(see Fig. 3.13). We will refer to this lack of correlation between maps as map decoupling. Specifically, we observed that the post-synaptic ocular dominance columns could desynchronize from the pre-synaptic ocular dominance columns. This appears to happen when the width of the dendritic arbors is large enough to span multiple pre-synaptic ocular dominance columns. When post-synaptic dendritic windows span multiple ocular dominance columns, the post-synaptic neurons have the choice of which eye to represent, and thus dominate post-synaptic column development (see Fig. 3.12). Because the width of pre-synaptic ocular dominance columns is largely driven by axonal arbor size in this model, this phenomenon can also be predicted from the relative sizes of the dendritic arbors and axonal arbors.

As a side-effect of coupling, the scale of map features can become sensitive to different parameters. For example, the width of post-synaptic ocular dominance columns depended on axonal arbor size when there was a high degree of coupling, but depended mostly on the radii of lateral excitation and inhibition when there was a low degree of coupling. This may provide an indirect way of observing map decoupling, as axonal arbor size and excitation/inhibition radii may support conflicting, testable predictions in real tissue (see the progressive change in column width in Fig. 3.13).

Whether the conditions necessary for map decoupling can arise in real cortex probably depends on the species and the cortical area in question and additional factors. In V1 for macaques and higher order primates, the width of the ocular dominance stripe varies between 395µm to 670µm (Horton and Hocking, 1996). However, it is unclear whether this reflects axonal or synaptic organization. Measures of axonal arbor area suggest that the diameter of incoming axonal arbors could be as large as 800µm (Humphrey et al., 1985). On the other hand, V1 dendritic arbors seem to have diameters around 200µm to 300µm, depending on the neuronal population studied and the technique used (Elston and Rosa, 1997; Levy et al., 2014). Given these loose estimates, our model would suggest that V1 in human and higher primates features strong coupling between pre-synaptic and post-synaptic maps.

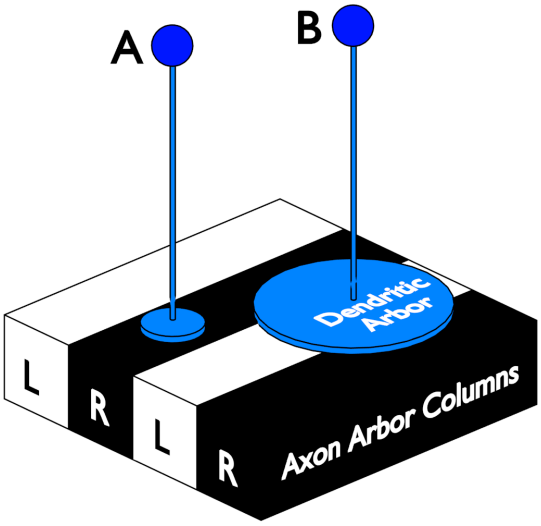

Figure 3.12. Axon Game Ocular Dominance Columns
This diagram shows how arbor size and map coupling are related. Neurons (A) and (B) have dendritic arbors that intersect the pre-synaptic axonal arbors, which group into pre-synaptic ocular dominance columns, pictured as the columns labeled (L) and (R) for left-eye representing and right eye representing. Neuron (A) has a dendritic arbor that is smaller than the width of the pre-synaptic ocular dominance columns while neuron (B) has a dendritic arbor that is much larger than the width of the pre-synaptic ocular dominance columns. Because neuron (A)'s dendrites only span one ocular dominance column it can only learn to represent a visual information from one eye. This small dendritic arbor size leads to a tight coupling between pre-synaptic and post-synaptic maps. On the other hand, neuron (B)'s dendrites spread over multiple pre-synaptic ocular dominance columns, meaning that neuron (B) can learn to represent information from either eye. This large dendritic size leads to low coupling between pre-synaptic and post-synaptic maps.

While axons projecting from the LGN to V1 appear to have relatively large arbors compared to their dendritic targets, some mathematical models have suggested that connections from V1 to higher areas may have smaller axonal arbors (Chklovskii, 2000). Unfortunately, much less is known about higher order cortical maps, and additional research is required to make further predictions about higher order map decoupling under this model.

As an additional factor, the model used in this paper featured no backwards stabilizing force from the dendritic arbors to the axonal arbors. Depending on whether this force

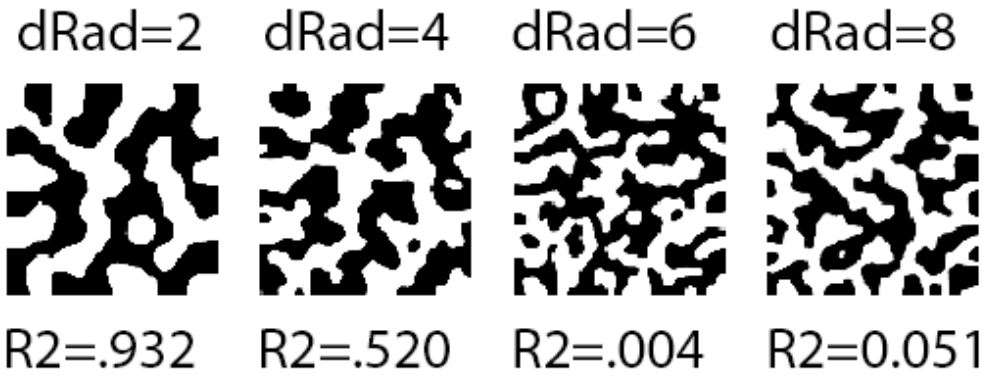

Figure 3.13. Axon Game Ocular Dominance Columns
Shows a progression of post-synaptic ocular dominance maps as the dendritic radius increases (dRad), and each map's correlation with the pre-synaptic ocular dominance map. The pre-synaptic ocular dominance map remained constant for each of these sub-simulations and had an approximate arbor radius of 6. Also, note that the widths of the ocular dominance columns change as the post-synaptic dynamics begin to dominate. The Pearson R^2 for correlations between the maps within this simulation all had exceedingly low p-values, but this is somewhat misleading as the simulation would need to be re-run many more times to get a proper estimate of the expected correlation across simulations. As such, these R^2 statistics are mainly here to illustrate a trend, further simulation would be required to estimate that trend with enough precision to find a function that maps arbor size to expected correlation precisely.

is present in real cortex and what its relative strength is, pre-synaptic post-synaptic map coupling could be affected. These effects have yet to be explored in detail.

These simulation results, based on biologically motivated hyper-parameters, suggest that researchers should consider the use of convergent measures of presynaptic and postsynaptic map organization, as their synchrony may not be guaranteed. Our simulations predict that whenever dendritic arbors are close to the same size or are larger than the axonal arbors, post-synaptic columns could be highly decoupled from their pre-synaptic counterparts. To our knowledge, it is not standard practice to use both pre-synaptic and post-synaptic sensitive methods within the same tissue sample and may be methodologically difficulty. Thus, the degree of pre-synaptic and post-synaptic map coupling may be unknown for even the most well-studied maps.

Models of synaptic development or axonal development in isolation do not predict the sensitivity of map coupling to arbor size. Synaptic models typically assume each neuron has an equivalent ability to access connections from either eye, often eschewing representations of axonal-dendritic proximity and arbor sizes. Conversely, axonal models often only simulate axonal arbor development while eschewing representations of dendritic arbors and synaptic strengths. Representation of axonal arbors, dendritic arbors and synaptic strength is necessary to see the map coupling interaction. However, it is also necessary for a model to be at the right scale to observe this interaction's effect on cortical maps. The fact that the Axon Game can be combined with models like H-LISSOM to allow this, makes it a unique contribution to cognitive neuroscience.

3.10 General Discussion

We built a novel axonal development model called the Axon Game that incorporates many of the well-known aspects of axonal dynamics in cortex, with the goal of integrating axonal dynamics into existing neural network designs and synaptic learning models for modeling large scale cortical learning phenomena. First, we demonstrated that a wide variety of cortical maps that can be defined through axonal organization can be developed using the axon game. Second, using a combination of the Axon Game and H-LISSOM, we demonstrated that a model of the interaction between axonal and synaptic dynamics on a large scale can largely replicate maps generated purely from synaptic cortical map models. Further, these simulations made the novel theoretical prediction that under certain biological conditions, pre-synaptic and post-synaptic cortical maps can be completely decoupled despite possessing similar qualitative characteristics. This map de-coupling prediction can be tested in future studies of human cortex, which can in turn be used to refine or reject future models.

The axon game also has implications for computational modeling. In preliminary work, we integrated the axon game with a simple deep learning architecture in order to learn

the sparse connectivity between layers as a pre-training step. This allows a deep model to learn flexible projection topologies that distribute connections synergistically with and in anticipation of weight learning. We found that a simple deep network augmented with the axon game could achieve state of the art performance for non-convolutional networks on the CIFAR-10 benchmark. Axon-game-augmented deep learning could be useful in a number of domains, particularly in situations that may benefit from structured sparsity but are resistant to convolutional methods.

Today, many researchers conceptualize neural circuits on both large and small scales in terms of the connections themselves (axo-dendritic contact), and the weights of those connections (efficacy and number of synapses). However, to date, abstract neural network modeling efforts have tended to focus on the latter. Models like the Axon Game make simulations of paired axonal and synaptic learning more accessible to a wider neural modeling audience. By providing tools to model how these plastic processes interact, we hope to advance the understanding of cortical development and neural systems in general.

3.11 Article Appendix A: Simulation Settings for Section 3.8

This section contains the parameter settings for the axon game used in the section titled "Simulated Axonal Maps". The version of the axon game implemented for this paper uses a simple auto-scale feature whereby some of the input parameters are adjusted to produce results that can be compared to a standard 100x100 scale simulation. The α_{APDV} is multiplied by a factor of the largest simulation dimension divided by 100 when an axis of the simulation is larger than 100. Additionally, the starting exuberance EX_0 is also scaled by the same factor.

Table 3.3. Simulation Settings Axon Game Only

Symbol	Definition
Res	170x250
Co-Act	Covariance
$Seed$	True
σ_{seed}	0
σ_y^{diff}	10
σ_η^{diff}	1
α_N	6
α_S^0	.002
α_S^1	.04
α_R	.08
α_C	50
α_{global}	.3
α_{local}	.08
β_s	.1
B	8
k	200
t_{end}	100
EX_0	1.5
EX_k	.025
P_0^{grow}	1
P_1^{grow}	1

3.12 Article Appendix B: Simulation Settings for Section 3.9

This section contains the simulation parameters used for the second set of demonstrations that combined the Axon game with H-LISSOM.

Table 3.4. Simulation Settings Axon Game Only

Symbol	Definition
Res	100x100
Co-Act	Covariance
$Seed$	True
σ_{seed}	.1
σ_y^{diff}	10
σ_η^{diff}	1
α_N	500
α_S^0	.001
α_S^1	.5
α_R	.016
α_C	5
α_{global}	.3
α_{local}	.16
β_s	.1
B	114
k	200
t_{end}	300
EX_0	4
EX_k	.05
P_0^{grow}	1
P_1^{grow}	.4

Table 3.5. H-LISSOM

Symbol	Definition
σ^E	1
σ^I	6
β^η	.004
β^U	.0004
β^A	.004
u	.5
α	.5
δ_{w}	.25
δ_θ	.001
δ_ϵ	.001
Reps	10

CHAPTER 4

INTEGRATED CORTICAL LEARNING MODELS

4.1 Introduction

For this dissertation, I developed a model which combined axonal learning, simple cell learning, and complex cell learning, which I called an *Integrated Cortical Learning Model* (ICL). A major goal in designing the ICL architecture was to create a model which avoids biologically implausible assumptions inherent in Deep Convolutional Neural Networks (DCNNs). The ICL architecture replaces fixed convolutional connections with axonal learning, global weight-sharing with simple cell map learning, and fixed pooling layers with complex cell learning. As a further goal I wanted the ICL architecture to be entirely unsupervised, since unsupervised visual learning is likely the dominant form of visual learning in mammals.

From a theoretical perspective ICL models can be thought of as an attempt to more rigorously implement functional versions of the hierarchical theory of cortex (Hubel and Wiesel, 1962). The basic idea behind the hierarchical theory of cortex is that there are two special populations of cells in V1 called simple cells and complex cells. The simple cells learn to encode stimuli as inflexible, but highly selective features, such as edges at particular locations and angles. The complex cells integrate inputs from multiple simple cells in order to respond more broadly, such as to edges at a particular angle anywhere within a large visual region. Under this theory, the pattern of simple cell layers feeding into complex cell layers may be repeated in higher-level cortical areas, such as V2 through IT, creating incrementally more tolerant and sophisticated representations of objects (Hubel and Wiesel, 1962; Riesenhuber and Poggio, 1999).

DCNNs originated from early models of the hierarchical theory of cortex(Fukushima, 1980), where convolutional feature units acted like simple cells and pooling units acted like complex cells. With the ICL architecture, I am essentially making another model of the

hierarchical theory of cortex, but using more detailed models of connectivity development, simple cell learning, and complex cell learning. Given that DCNNs, with their very schematic implementation of the hierarchical theory of cortex, have already greatly advanced the study of human vision and the brain, ICL models which incorporate more detailed forms of cortical plasticity should also advance the field.

It is important to understand the relationship between the theoretical constructs introduced in Chapter 2 and the practical simulation models introduced in this chapter. The *Temporal Relation Manifold* (TRM) framework introduced in Chapter 2 is a framework for understanding how unsupervised category learning can work, and Windowed-Temporal-Auto-Untangling (W-TAU) is a theory for how this might be accomplished using neural mechanisms. The full ICL model developed for this project attempts to implement W-TAU using a combination of axonal, simple cell, and complex cell learning. I define the general ICL family of models to include any model which combines axonal, simple cell, and complex cell learning. When the term ICL model is used in this dissertation, I will usually be referring to one of the particular ICL models built for this dissertation project. Finally, an ICL module is a sub-group of neural network layers that implement axonal, simple, and complex cell learning as a unit and can be thought of as simulating learning in a discrete visual cortical area such as V1 or V2.

Chapter Overview. In Section 4.2, I will discuss several of the specific issues with DCNN mechanisms that ICL models seeks to address. In Section 4.3, I will discuss the general form of an ICL module and how its components address the issues with DCNNs laid out. In Section 4.4, I will provide additional context about to help motivate and interpret trace learning. In Section 4.5, I will discuss the reprocessing strategies used for the two datesets used in Chapter 5, as well as a layer-wise normalization method used for the model. In Section 4.6, I will describe the specific variations of the full ICL module structure that will be investigated in later chapters. In Chapter 5, I will empirically evaluate several versions of a particular ICL model.

Code for Replication. A complete codebase for the models used in Chapter 5 can be found here `https://bitbucket.org/jryland/icl-dissertation/`. The models can all be run in google colab as of this writing, removing the need for specialized hardware or a local tensorflow environment. Thus using a free google colab environment is highly recommended.

4.2 Issues With DCNN Mechanisms that ICL Addresses

DCNNs models are valuable both to computer vision and to computational cognitive neuroscience. Although DCNNs have a quite simple architecture, they demonstrate exceptionally good performance on object recognition dataset. As previously mentioned though, DCNNs are often not intended to be neurally plausible models of cortical understanding their compenents are rarely intended to have direct biological interpretations. With the development of the ICL model, I intend to build a model which is not too much more complicated than a basic DCNN, but which captures some of the details of today's understanding of cortical learning better. To help motivate our discussion of the ICL model developed in this dissertation, this section takes a closer look at the standard components of DCNN models which do not have direct neural interpretations.

4.2.1 Issue 1: Convolutional Windows

DCNNs use a fixed form of connectivity between their layers called *convolutional windows*, where each neuron is situated on a grid and sees a slightly shifted set of inputs units in the input image compared to its neighbors. This is called a *convolutional window* structure. Convolutional windows loosely model the retinotopic nature of connectivity to V1 and other early visual areas.

While convolutional windows are probably fine for loosely modeling early visual areas like V1, as they are strongly retinotopic, convolutional windows will likely not suffice for more detailed modeling of V1 or for higher-level visual areas. Connectivity in cortex is controlled

by axonal development. Further, axonal development is reactive to learning amongst other forces (Benson et al., 2001) leading to the development of connectivity structures, such as ocular dominance columns, which are dependent on environmental exposure. But low-level structures like ocular dominance columns in V1 and their high-level analogues in late visual cortex cannot be modeled by DCNN connectivity due to its fixed nature.

In Section 4.3.2, I outline how my ICL model implements a model of axonal development that is both reactive to genetic forces and to environmental exposure as a replacement for the fixed convolutional windows of DCNNs.

4.2.2 Issue 2: Convolutional Weight-Sharing

DCNNs exploit the grid-like nature of of artificial images and of their features neurons in order to share learning between neurons. Essentially, within a *feature map* each neuron will share the same set of weights, only shifted to the location of that neuron in the map. This is called *convolutional weight-sharing*. Weight-sharing has the advantage that a feature learned on one side of the visual field will also be automatically learned all across the visual field. Convolutional weight-sharing loosely imitates an important quality of V1 in that the same visual features seem to be learned for all visual locations.

Convolutional weight-sharing is fine for abstractly modeling simple cell learning V1 and later visual areas, but it is insufficient for more detailed modeling of the ventral stream. First, the retinal image is not laid out on a regular grid, and it is sampled continuously across scale space in a foveated fashion. This makes any weight-sharing scheme based around the actual retinal image necessarily more complicated. Second, such a weight sharing scheme would require a huge number of long range lateral connections spanning each visual area. There is no evidence for and extensive network of lateral connections that perform this role.

Instead of weight-sharing, visual cortex seems to use a set of local competitive learning rules implemented via short range lateral connections and the relatively shift invariant

statistics of visual exposure to guarantee that a useful features learned in one visual location will tend to be learned for all visual locations (von der Malsburg, 1973; Stevens et al., 2013; Sirosh and Miikkulainen, 1994). The local competitive method has the added advantage that it it can deal with irregularly sampled inputs easily and that it will not over generalize and learn features in locations and configurations where they never occur.

As a consequence of local cortical learning rules, V1 develops regions of specialized neural selectivity like iso-orientation columns, and late visual areas may develop some sort of analogue for high-level features. On the other hand DCNNs, with their rigid weight-sharing method, do not develop structures like iso-orientation columns and potential analogues of these structures in high-level cortex.

In Section 4.3.3, I outline how my ICL model implements a model of simple cell learning that makes use of local competitive rules as a replacement for convolutional weight-sharing.

4.2.3 Issue 3: Fixed Pooling Methods

DCNNs further exploit the grid-like nature of artificial images and of their feature maps in order to generate tolerance to small shifts and deformations in image features. DCNNs accomplish this by *pooling* or summarizing the activity of non-overlapping squares of neurons in their feature maps. Essentially, this means that the activation of neurons that detect slightly shifted versions of a feature over a specific region of visual space will be summarized by a single output neuron called a *pooling neuron*. As such, a pooling neuron can be thought of as detecting a specific image feature anywhere within a visual region. Again, this method depends on a grid-like structure for both input neurons and for feature map neurons, whereas cortex must pool over a highly irregular input and feature structure. In theory, cortex may learn pools using a similar sort of local competitive interaction between neighboring neurons. Cortex may even use the temporal statistics of its environment in order to learn its pooling structure, as suggested by trace learning theories (Földiák, 1991). An advantage of learned

pooling approaches, such as trace learning, is that they can learn to be tolerant to a wide spectrum of feature variations beyond just shifting.

In Section 4.3.4, I outline how my ICL model implements a model of complex cell learning built on trace learning as a replacement for the fixed pools of DCNNs.

4.3 Overview of the Integrated Cortical Learning Model

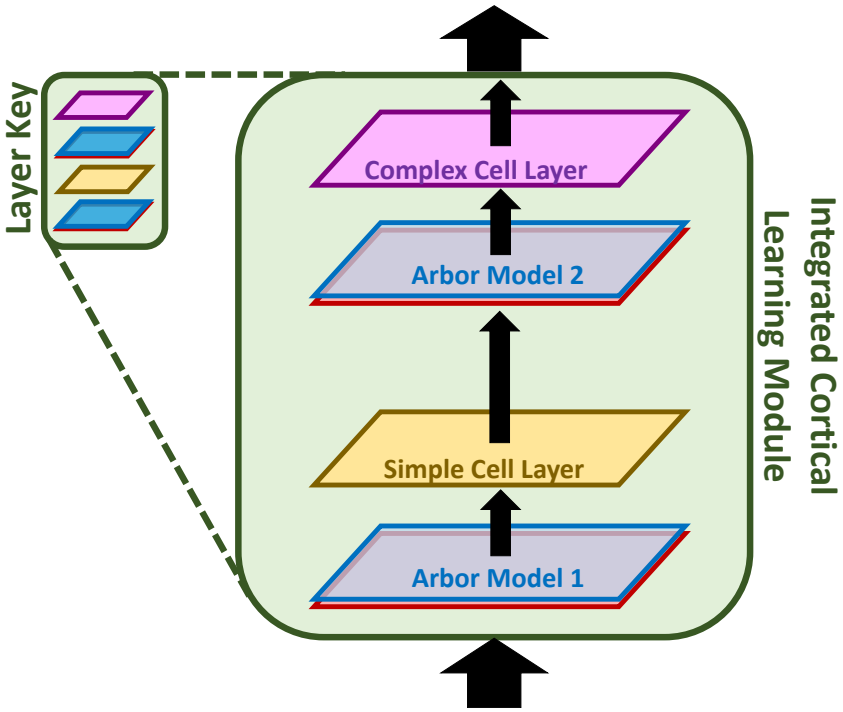

Figure 4.1. ICL Module Structure
This figure shows the layer structure of a full ICL module. The RED/BLUE layers represent the arbor model simulation of axonal connectivity between the neural layers. The YELLOW layer is the simple cell layer, and the PINK layer is the complex cell layer. A layer key has been provided in the upper left corner for reference in later diagrams.

4.3.1 Module Structure and Training

Like a DCNN, the ICL model is composed of a series of Modules stacked on top of one another. A DCNN module is composed of a convolution and a pooling layer. On the other

hand, an ICL module is composed of axonal learning layers, a simple cell learning layer, and a complex cell learning layer. Each can be thought of as addressing a specific implausible mechanism within DCNN architectures as will be discussed in the following sections.

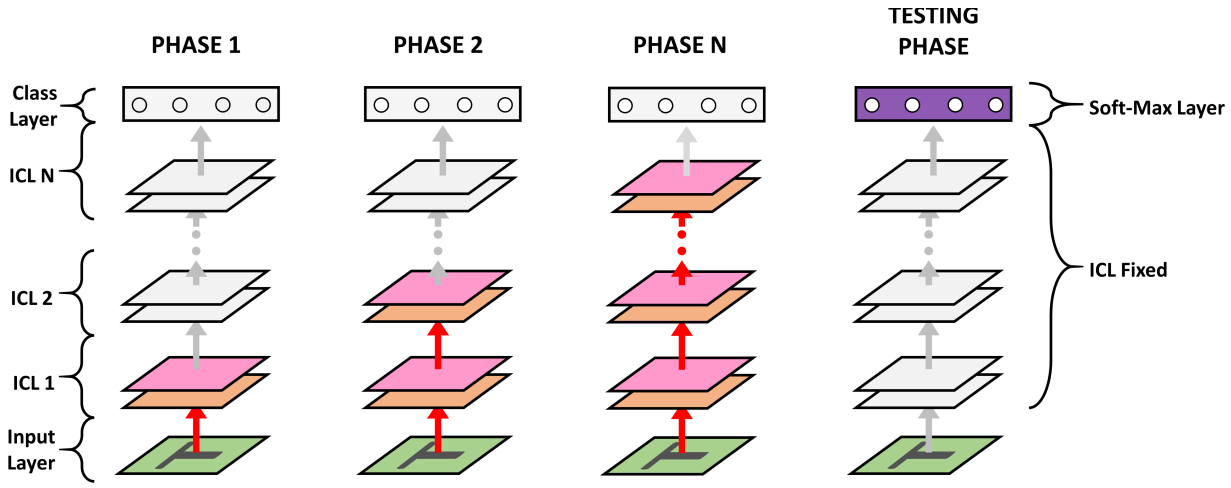

Figure 4.2. Training Phases of the ICL Model
This figure shows the training phases of the ICL model. In phase 1, the first ICL module uses its internal unsupervised dynamics to learn how to represent features in the input layer. In phase 2, the second ICL module learns to represent more complex features in the first ICL layer, while the weights of ICL module 1 are held fixed. This proceeds until the final layer N is trained, after which, the testing phase begins. During the testing phase, a soft-max layer is added and trained for classification on a training set, while the weights of all previous ICL modules are held fixed. Finally, the fully trained model is run on a validation set or put through an experimental protocol.

When training an ICL model, a series of ICL modules are stacked and trained according to their individual local learning rules similar to stack auto-encoder training (Bengio et al., 2007) (i.e. only one module is training at a time). The individual learning rules for the layers within each ICL module will be discussed in the following sections. Finally, for certain testing purposes, a final Softmax classification layer can will be added to the top of the ICL model and trained using supervised learning, while all of the ICL model's parameters are held fixed. This final classification layer is called a *readout layer*, in that its purpose is

to demonstrate how the top layer of the ICL model has encoded observed categories, while keeping the underlying representations of the ICL model fixed.

4.3.2 The Connectivity Model: Arbor Layer

In order to address the issue of using convolutional windows to model cortical connectivity (Sec. 4.2.1), I instead based the connectivity of my ICL model on a semi neuro-plausible model of axonal development, that would allow the model to radically change its structural connectivity also supporting weight learning. The model developed is called the *arbor model*, but as a part of the ICL module it is called the *arbor layer*. In general, the arbor layer should be compatible with many diverse neural network architectures, imbuing them with the ability to adjust their structural connectivity adaptively. Further, given its simplicity, the arbor layer could easily be adapted for more biologically motivated simulations for more abstract neural network usages. Below I will discuss how the arbor layer works and integrates with the ICL module.

The Arbor Layer simulates axonal connections growing from one population of neurons to another. These axonal connections terminate in axonal arbors which are represented as Gaussian influence fields (See Fig. 4.4 & 4.3.2.1). The dendritic arbors are also represented as Gaussian influence fields (See Fig. 4.4). The amount that arbors influence one another is dependent on the degree of overlap between their Gaussian influence fields. There are three forces acting on arbors which are directly inspired by neuroscience and the classic multi-factor model of axonal development (Fraser and Perkel, 1990), and include competition, activity dependent plasticity, and chemo-affinity.

4.3.2.1 Gaussian influence field

A *Gaussian influence field* is a 2D Gaussian function centered on the location of an axonal or dendritic arbor set within in a 2-dimensional interaction space, $g(\mathbf{x}_i, \sigma_j)$. Arbors influence

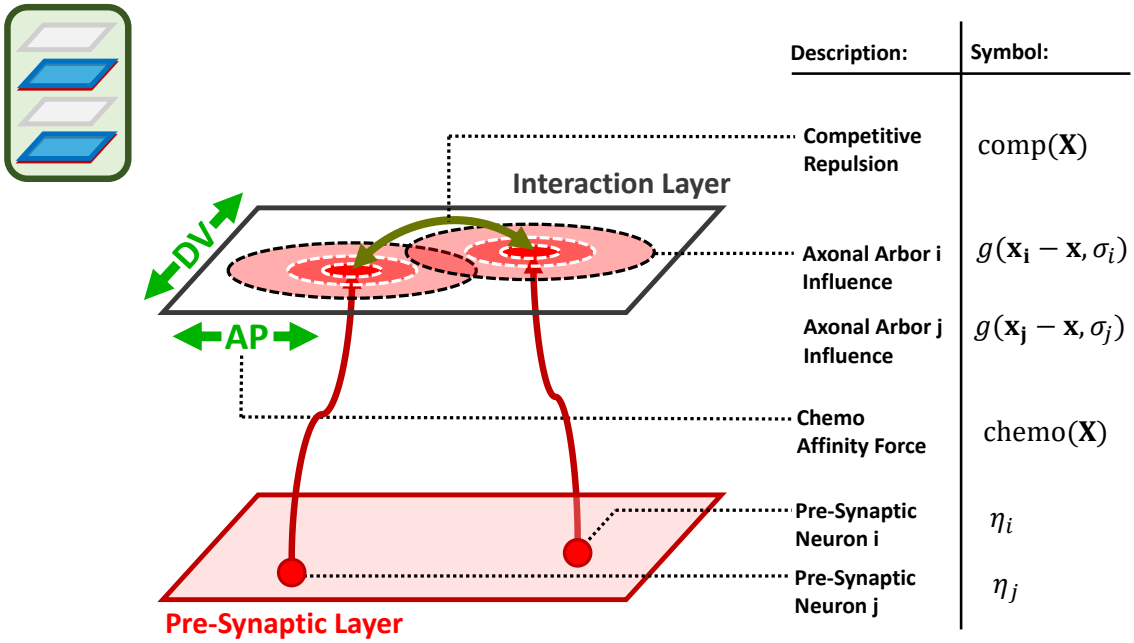

Figure 4.3. Interaction of Pre-Synaptic and Post Synaptic Arbors
This figure shows how pre and post synaptic arbors interact with the arbor model. The RED disk on the LEFT in the interaction layer depicts the Gaussian influence field of pre-synaptic neuron i. The RED disk on the RIGHT in the interaction layer depicts the Gaussian influence field of post-synaptic neuron h. The GREY-BROWN arrow indicates repulsion between pre-syanptic neuron i and post-synaptic neuron j, proportional to the overlap between their Gaissian influence fields. The GREEN arrows represent the chemoaffinity force, with DV representing a chemical gradient from dorsal to ventral and AP representing a chemical gradient from Anterior to Posterior.

each other proportional to the overlap of their Gaussian influence fields. The overlap is defined as the integral of both Gaussian influence functions multiplied by one another (see, Eq. 4.1). Conveniently, this is equivalent to the equation for convolving one Gaussian function with another, which reduces to a simple 1D Gaussian formula, with a new $\sigma_k = \sqrt{\sigma_i^2 + \sigma_j^2}$ (see, Eq. 4.2). In practice, I typically directly manipulate σ_k instead of σ_i and σ_j, but for more biologically oriented simulations it would make sense to manipulate them separately.

For the following equations, $\mathbf{x}_i$ refers to the center of an arbor i (axonal or dendritic) on a 2-dimensional cortical sheet and. σ_i refers to the standard deviation of the Gaussian influence fields of arbors i and j respectively.

$$\text{Ginf}(\mathbf{x}_i, \mathbf{x}_j) = \int g(\mathbf{x} - \mathbf{x}_i, \sigma_i) g(\mathbf{x} - \mathbf{x}_j, \sigma_j)\, d\mathbf{x} \tag{4.1}$$

$$\text{Ginf}(\mathbf{x}_i, \mathbf{x}_j) = g\left(\|\mathbf{x}_i - \mathbf{x}_j\|, \sqrt{\sigma_i^2 + \sigma_j^2}\right) \tag{4.2}$$

4.3.2.2 Arbor Forces in Detail

The arbor layer simulates three forces that move the axonal arbors: chemo-affinity, competition, and activity dependent plasticity, by minimizing a layer objective function (see, Eq. 4.3). The chemo-affinity force minimizes the distance between each arbor and its ideal position for developing a retinotopic map (see Eq. 4.4 & Fig. 4.3). Chemo-affinity is generally set to be very weak in order to match cortical observations (Fraser and Perkel, 1990). The competition term pushes axonal arbors away from one another depending on their influence function overlap (see Eq. 4.5 & Fig. 4.3). The activity dependent plasticity moves axonal arbors towards dendritic arbors depending on their influence function overlap and on how often they co-activate or on how correlated their activity over time is (see Eq. 4.6 or 4.7 & Fig. 4.4). Activity dependent plasticity is usually set to be a very strong force in order to match general qualitative findings from neuroscience (Fraser and Perkel, 1990).

The following function $\mathcal{L}(\mathbf{X})$ in Eq. 4.3 is the global objective function for an arbor layer. $\mathbf{X}$ contains the location of each arbor and is the set of parameters that are optimized in the model. The γ constants weight the forces relative to one another. Minimizing, this function via standard gradient descent or with momentum will simulate the forces discussed in the following sections.

$$\mathcal{L}(\mathbf{X}) = \text{chemo}(\mathbf{X})\gamma_{\text{chemo}} + \text{comp}(\mathbf{X})\gamma_{\text{comp}} + \text{coact}(\mathbf{X})\gamma_{\text{coact}} \tag{4.3}$$

4.3.2.2.1 Chemo-Affinity Force Chemo-affinity was one of the earliest forces hypothesized to drive axonal arbor placement in developing cortex (Sperry, 1963). The general idea is that chemical gradients along the cortical sheet allow axonal arbors to detect where they are in a cortical area and to roughly adjust themselves towards a predetermined location.

Here I implemented chemo-affinity with a simple sum squared error penalty, with $\mathbf{x}_i^0$ being the target location for axonal arbor i and $\mathbf{x}_i$ being its current location.

$$\text{chemo}(\mathbf{X}) = \sum_i \|\mathbf{x}_i - \mathbf{x}_i^0\| \tag{4.4}$$

4.3.2.2.2 Competition Force During several studies of axonal development, it was discovered that axonal arbors try to distribute themselves more or less evenly within a bounded cortical or sub-cortical region like the LGN or V1 (Fraser and Perkel, 1990). Further, re-segmentation work showed that axonal projections to halved neural areas would compress their representations accordingly. One hypothesis for why this behavior happens is that axonal arbors within an area like V1 are in competition with one another. This means they will try to avoid clumping together and will adjust towards a uniform coverage and density over an area.

I implemented a competition force in the arbor layer by penalizing each axonal arbor i by the amount of overlap it has with every other arbor j using the Gaussian influence function $\text{Ginf}(\mathbf{x}_i, \mathbf{x}_j)$ given their locations. Conveniently, the local nature of the Gaussian influence function means that this penalty prevents clumping and promotes uniform distribution without generating large gradients from distant arbor interactions.

$$\text{comp}(\mathbf{X}) = \sum_i \sum_j \text{Ginf}(\mathbf{x}_i, \mathbf{x}_j) \tag{4.5}$$

4.3.2.2.3 Activity Dependent Plasticity Many researchers were not convinced that the precise locations of axonal arbors in areas like V1 could be arrived at solely based on chemical gradients, and further fixed chemical gradients would not explain the development of clusters of arbors carrying information from the same eye that depend on spontaneous activity or experience (i.e. ocular dominance columns) (Fraser and Perkel, 1990). Katz and Shatz (1996) reviewed evidence that the brain uses both spontaneous neural activations before gestation and actual experience in order to develop ocular dominance columns. The general idea was that arbors which tended to activate together would migrate towards one another and stabilize. Earlier the Fraser and Perkel (1990) model showed that incorporating activity dependent plasticity with other forces could rectify a large amount of seemingly conflicting experimental data. Now activity dependent forces are commonly used in axonal plasticity models (Benson et al., 2001).

Here I provide two different implementations for activity dependent plasticity. The first method shown in Eq. 4.6 is generally more appropriate when the activations of pre-synaptic neuron are sparse and strictly positive. The second method shown in Eq. 4.7 is generally more appropriate to when the activations of the pre-synaptic neurons are non-spare or are both positive and negative.

The first version of the activity dependent plasticity reward term coact($\mathbf{X}$) in Eq. 4.6 implements a simple local attractive force between each pre-synaptic axonal arbor i and each post-synaptic dendritic arbor j dependent on their current co-activity, with $\eta_i^{pre}(t)$ being the current activity of pre-synaptic axonal arbor i at time t and $\eta_j^{post}(t)$ being the current activity of pre-synaptic axonal arbor j at time t. The reward for axonal arbors to move towards co-activating dendritic arbors is scaled by the overlap between their Gaussian influence functions, given by $\text{Ginf}(\mathbf{x}_i, \mathbf{y}_j)$. $\mathbf{y}_j$ refers to the center of post-synaptic neuron i's dendritic arbor. Note, that the locations of the dendritic arbors do not move, and that they are typically arranged in a regular grid pattern.

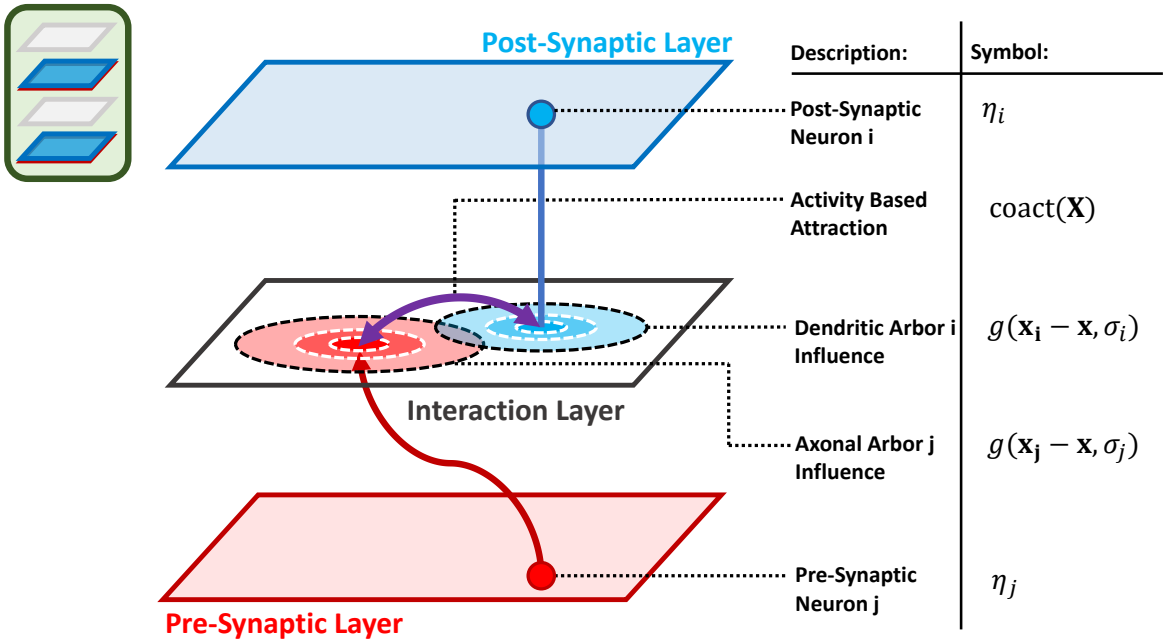

Figure 4.4. Example of different types of activation maps
This figure shows how pre and post synaptic arbors interact with the arbor model. The RED disk in the interaction layer depicts the Gaussian influence field of pre-synaptic neuron j. The BLUE disk in the interaction layer depicts the Gaussian influence field of post-synaptic neuron i. The PURPLE arrow indicates attraction between pre-syanptic neuron j and post-synaptic neuron i, proportional to the overlap between their Gaissian influence fields and their level of mutual activity.

$$\text{coact}(\mathbf{X}) = -\sum_i \sum_j \left[\text{Ginf}(\mathbf{x}_i, \mathbf{y}_j) \sum_t \eta_i^{pre}(t) \eta_j^{post}(t) \right] \tag{4.6}$$

The second version of the activity dependent plasticity reward term coact($\mathbf{X}$) in Eq. 4.7 implements a simple local attractive force each pre-synaptic axonal arbor i and each post-synaptic dendritic arbor j dependent on how correlated they are, with $\boldsymbol{\eta}_i^{pre}$ being the activation history of pre-synaptic neuron i over a set time period and $\boldsymbol{\eta}_j^{post}$ being the activation history of pre-synaptic dendritic arbor j over the same time period. Again, the reward for axonal arbors to move towards correlated dendritic arbors is scaled by the overlap of

their Gaussian influence functions given by $\text{Ginf}(\mathbf{x}_i, \mathbf{y}_j)$. Note that $\mathbf{y}_j$ refers the center of post-synaptic neuron i's dendritic arbor, and that the dendritic arbors are stationary.

$$\text{coact}(\mathbf{X}) = -\sum_i \sum_j \text{Ginf}(\mathbf{x}_i, \mathbf{y}_j)\, \text{Corr}(\boldsymbol{\eta}_i^{pre}, \boldsymbol{\eta}_j^{post}) \tag{4.7}$$

4.3.2.2.4 Arbor Layer Window Functions During an update step, a post-synaptic neuron connects to the pre-synaptic neurons whose axonal arbors that are closest to its dendritic arbor. This is implemented as a Top-K sorting operation on the distance between the dendritic arbors and the axonal arbors. For efficiency and stability, only a fraction of the axonal arbor positions and the dendritic arbor Top-K windows are updated per-step, usually between 1% and 10%. Importantly, the weights associated with each connection are non-destructively preserved through window changes to ensure stability. It should be noted that the re-ordering of weights due to the Top-K operation means that momentum like learning objectives may be unstable for weight learning (though not for arbor position learning).

To represent the changing list of weights and input units to each neuron I will use two window functions. In Eq. 4.8 $\text{Arbin}_{\eta_i^{post}}(\mathbf{z})$ gives the current inputs to post-synaptic neuron i, with $\mathbf{z}$ being the vector of all inputs for the layer. In Eq. 4.9 $\text{Arbw}_{\eta_i^{post}}(\mathbf{W})$ gives the current weights of neuron i, with $\mathbf{W}$ being the matrix of all weights for the layer. Although Arbin and Arbw are both dependent on the locations of the axonal arbors X, this relationship is not directly differentiable and is a fairly complicated dynamic process, so it is omitted as an input to the function. With some minor modifications, however X can be added as a differentiable term.

$$\text{z}_i = \text{Arbin}_{\eta_i^{post}}(\mathbf{z}) \tag{4.8}$$

$$\text{w}_i = \text{Arbw}_{\eta_i^{post}}(\mathbf{W}) \tag{4.9}$$

4.3.2.3 Arbor Layer Demonstration & Observations

As a note of comparison between the axon game in Chapter 3 and the arbor model introduced here, the arbor model is generally much faster than the axon game. This is for two reasons. First, in the arbor model the size of dendritic and axonal arbors is represented by two constants that simply adjust an influence function, while in the axon game, larger axonal arbors and dendritic arbors increase the number of objects to be managed proportional to their area. Second, the forces in the arbor model are very direct (i.e. via gradient descent), whereas the axon game uses random exploration in a genetic algorithm, which is more wasteful. Generally, this means the arbor model can learn large scale connectivity maps much faster than the axon game, as seen in Fig. 4.5.

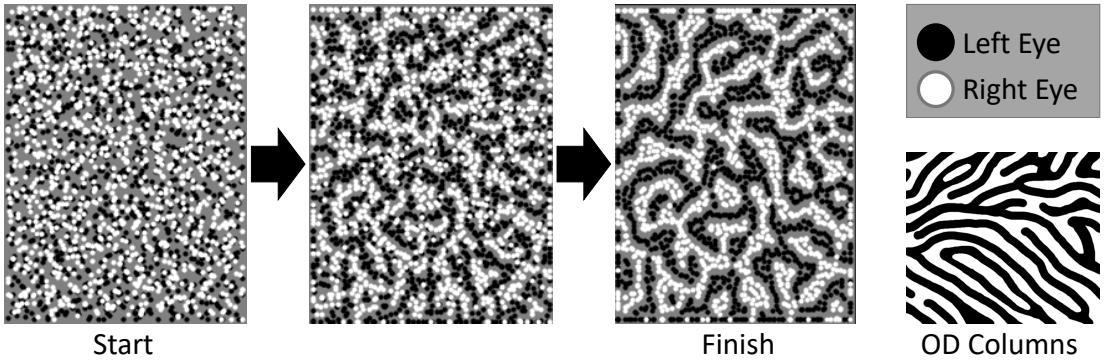

Figure 4.5. Arbor Model Learning Ocular Dominance Columns
This figure shows an example of the axon game taking an input image of 50x50 pixels and learning a 50x50 output ocular dominance map. The (WHITE) dots are axonal arbors belonging to inputs from the right eye, and the (BLACK) dots are axonal arbors that belong to the inputs from the left eye.

Fig. 4.5 shows an example of a small map being learned. This demonstration used simple winner take all activations for the post-synaptic layer where if the majority of inputs to a cell were on it turned on and vice versa. Note that this simulation quickly develops ocular dominance columns quite similar to those seen in V1.

4.3.3 The Simple Cell Model: Simple Layer

The simple layer was largely inspired by LISSOM, GCAL, Malsburg and Kohonen Map models of V1 simple cell feature map learning (Sirosh and Miikkulainen, 1994; von der Malsburg, 1973; Stevens et al., 2013; Kohonen, 1982).

First, the weight update mechanism borrows from both the Kohonen Map method and the LISSOM method. Essentially, a simple cell's weight update attempts to minimize the distance between the current weight vector and the current normalized input activation pattern with a speed proportionate to the activity of the unit (see Eq. 4.16). After that, the weight vector is normalized to the unit sphere (see Eq. 4.17). This method is essentially a conjunction of both the LISSOM and Kohonen map update methods. Using this method has the advantage that it can handle both positive and negative input values while keeping weights fixed to the unit sphere, whereas the kohonen map method does not restrict weights to the unit sphere, and the LISSOM method has stability issues with negative activations.

Second, the lateral inhibition and excitation is somewhat similar to the Malsburg and Kohonen models in that their influence functions are fixed. Each simple cell has an excitatory and inhibitory Gaussian influence on its neighbors (see Eq. 4.15). Usually, the inhibitory influence σ_{inh} is much wider than the excitatory influence σ_{exc}. For learning the lateral interaction modulated activity is used as the target. For the simple layer, the raw activation of the simple cells is passed as the output for the layer (i.e. without the lateral influences) (see Eq. 4.12). This is different from the Malsburg and Kohonen models in that the Kohonen method (Kohonen, 1982) only uses a shrinking excitatory Gaussian influence for learning, and the Malsburg model (von der Malsburg, 1973) uses rectified hyperbolas in order to build its lateral interaction weights.

Using fixed Gaussian lateral inhibition and excitation functions instead of learned lateral influence functions or rectified hyperbolas greatly increased the speed of the simulation. This is because 2D Gaussian filtering can be implemented in a two step process, where a

horizontal 1D Gaussian filter is applied and then a second vertical 1D Gaussian filter can be applied. This happens because a 2D Gaussian filter can be written the product of two 1D Gaussians (i.e. it is a separable filter). Further, I observed that Gaussian lateral influences still produced the desired V1 map qualities for lower-level simulations.

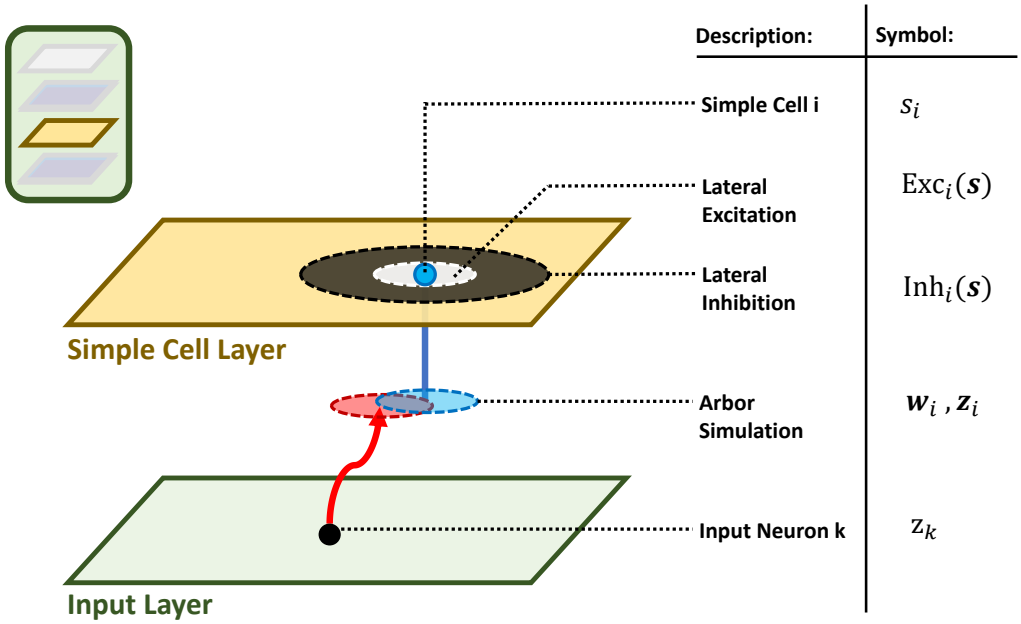

Figure 4.6. Diagram of the Simple Cell Layer
This figure shows how the simple cell layer is structured and how neighboring simple cells interact. The GREEN layer is the input layer for the model or input from another ICL module. The RED and BLUE disks denote the connection between input neuron k and simple cell i mediated by the interaction of their axonal (RED) and dendritic (BLUE) arbors respectively in an arbor simulation. The YELLOW layer is the simple cell layer. The WHITE disk indicates that simple cell i laterally excites its neighboring neurons. The BLACK disk indicates that simple cell i latterly inhibits its neighbors over a larger distance than it excites. Combined with hebbian learning and lateral interactions, the simple cell layer will learn a smoothly varying set of features.

4.3.3.1 Simple Cell Layer Equations in Detail

4.3.3.1.1 Simple Cell input Window Functions The simple cell layer takes input from the input layer or from the output of a previous ICL module. The pattern of activity coming into the layer is called $\mathbf{z}$. Each simple cell's weights and input window are managed by an arbor layer simulation, (i.e. the order of the weights and the membership of the input window will be changing). Here the input window of simple cell $\mathbf{s}_i$ is given by $\mathbf{z}_i$ in Eq. 4.10, and the current weights for simple cell $\mathbf{s}_i$ are given by in Eq. 4.11, where $\mathbf{W}$ is the matrix of all weights for the simple cell layer.

$$\mathbf{z}_i = \text{Arbin}_{s_i}(\mathbf{z}) \tag{4.10}$$

$$\mathbf{w}_i = \text{Arbw}_{s_i}(\mathbf{W}) \tag{4.11}$$

4.3.3.1.2 Simple Cells Activation and Lateral Interaction The simple cell layer uses the standard ReLu activation function defined in Eq. 4.12. ReLu which is short for Rectified Linear Unit and returns 0 when its inputs are negative and the original input value when positive, $\text{ReLu}(u) = \max(u, 0)$. Note that s_i is the actual activation that will be passed to the next layer. The lateral interactions defined next are only used for learning purposes.

$$s_i(\mathbf{w}_i, \mathbf{z}_i) = \text{ReLu}(\mathbf{w}_i^T \mathbf{z}_i) \tag{4.12}$$

For the purposes of learning, the activations of the simple cell layer are modified by lateral excitation and inhibition given by Eq. 4.13 and 4.14 respectively. $\text{Exc}_i(\mathbf{s})$ in Eq. 4.13 is excitatory influence of neighboring neurons on simple cell s_i, where $\mathbf{x}_i$ is the location of simple cell s_i, $\mathbf{x}_j$ is the location of simple cell s_j on the cortical sheet, and $g(\mathbf{x}_i - \mathbf{x}_j, \sigma_{\text{exc}})$ is a 2D Gaussian function centered on $\mathbf{x}_i$ and with a standard deviation of σ_{exc}. $\text{Inh}_i(\mathbf{s})$ in Eq.

4.14 is inhibitory influence of neighboring neurons on simple cell s_i, where $\mathbf{x}_i$ is the location of simple cell s_i, $\mathbf{x}_j$ is the location of simple cell s_j on the cortical sheet, and $g(\mathbf{x}_i - \mathbf{x}_j, \sigma_{\text{inh}})$ is a 2D Gaussian function centered on $\mathbf{x}_i$ and with a standard deviation of σ_{inh}. $\text{EI}_i(s)$ in 4.15, gives the combined influence of lateral excitation and inhibition on simple cell s_i. Note that an additional ReLu operation is performed removing any negative values.

$$\text{Exc}_i(\mathbf{s}) = \alpha_{\text{exc}} \sum_j g(\mathbf{x}_i - \mathbf{x}_j, \sigma_{\text{exc}}) s_j \tag{4.13}$$

$$\text{Inh}_i(\mathbf{s}) = \alpha_{\text{inh}} \sum_j g(\mathbf{x}_i - \mathbf{x}_j, \sigma_{\text{inh}}) s_j \tag{4.14}$$

$$\text{EI}_i(\mathbf{s}) = \text{ReLu}(\text{Exc}_i(\mathbf{s}) - \text{Inh}_i(\mathbf{s})) \tag{4.15}$$

4.3.3.1.3 Simple Cell Learning The weight update step for simple cell learning is divided into two steps. First, the weights of cell s_i move in the direction $\mathbf{w}_i^{\text{targ}}$ given by Eq 4.16, where ϵ is a term used to prevent arbitrarily large enhancement of small values. Note that the weights s_i of a simple cell move towards a unit normalized version of the current input activation on the window $\mathbf{z}_i$, and the speed of their movement is proportional to the lateral interaction modified activity of the simple cell given by $\text{EI}_i(s_i)$. Second, the weights of each simple cell s_i are hard-normalized to have a length of one in Eq. 4.17.

$$\mathbf{w}_i^{\text{targ}} = \text{EI}_i(s_i) \left(\frac{\mathbf{z}_i}{\|\mathbf{z}_i\| + \epsilon} - \mathbf{w}_i \right) \tag{4.16}$$

$$\mathbf{w}_i(t+1) = \frac{\mathbf{w}_i(t) + \sigma_{\text{lrate}} \mathbf{w}_i^{\text{targ}}(t)}{\|\mathbf{w}_i(t) + \sigma_{\text{lrate}} \mathbf{w}_i^{\text{targ}}(t)\|} \tag{4.17}$$

4.3.3.2 Simple Cell Layer Demonstration & Observations

As a note, we actually developed many versions of the simple cell layer with different activation functions and tweaked learning dynamics. For the purposes of illustration the version of the simple cell layer I chose to use for this dissertation a minimalist version that still implemented most of the expected dynamics of a simple cell model.

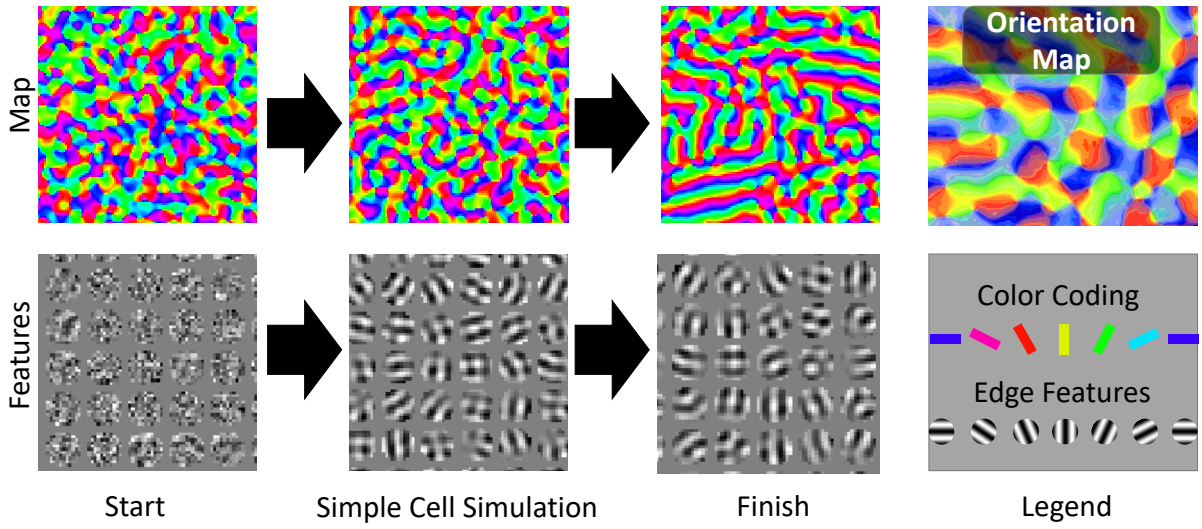

Figure 4.7. Demonstration of Simple Layer Simulation
The TOP row shows the evolution of orientation selectivity across a set of simple cells in a simple cell layer. The BOTTOM row shows the development of learned features for cells in the simple cell layer. Each color in the top row represents neurons which are sensitive to a particular orientation of edge contrasts, see the legend on the LOWER RIGHT. TOP RIGHT shows an example of a typical orientation map structure adapted from Bosking et al. (1997). Note that the simple cell layer develops similar strip and blob like regions of iso-orientation selectivity.

As a further note, I generally found that the recurrent activation used in many simple cell models (von der Malsburg, 1973; Sirosh and Miikkulainen, 1994; Stevens et al., 2013), was not strictly necessary to for good feature development or orientation map development. For reference, *Orientation Columns* are strips of neurons in V1 that respond to edges at the same orientation. This would make sense if the primary purpose of the recurrent activation

in V1 was simply to narrow down or sharpen the activity of already active neurons. It is possible that by omitting this mechanism my simple cell model may feature slightly poorer coverage of the feature space or lower stability than other typical simple cell models, in favor of mechanical simplicity and easier tuning.

Fig. 4.7 shows a simulation of a version of a simple cell layer. The particular version used for this demonstration actually used a competitive local winner take all function in order to restrict the output values to zero or one and a slightly different learning method. Regardless, all of the models learned quite similar features and produced quite similar orientation column maps.

4.3.4 The Complex Cell Model: Complex Layer

The complex cell layer uses a form of trace rule pool learning to learn the weights of its pools (Földiák, 1991). A trace is a temporal average of activations for a layer or set of units. For a more in depth explanation of Trace Learning and it's motivation see Sec. 4.4. Chapter 5 also incorporates a high-level interpretation Trace Pool Learning.

For the complex cell layer's form of trace learning, the pool weights of each complex cell are additively re-normalized. Here, additive re-normalization means that the target is added to the weights, then the weights are re-normalized to have a sum of 1 4.27. The target of learning is the trace of the activation after lateral interactions (see Eq. 4.25, Eq. 4.24 & Eq. 4.26).

Much like the simple cell layer, the raw output of the complex cell layer is passed along as the output (see Eq. 4.28). Here the initial activation used for each complex cell is a form of weighted Maxpooling operation.

This version of the complex cell update rule tended to generate edge effects. To correct for this and to encourage stability, the output of the complex cells was modified by an average normalization operation, where the average activation of each cell is normalized to 1 using a

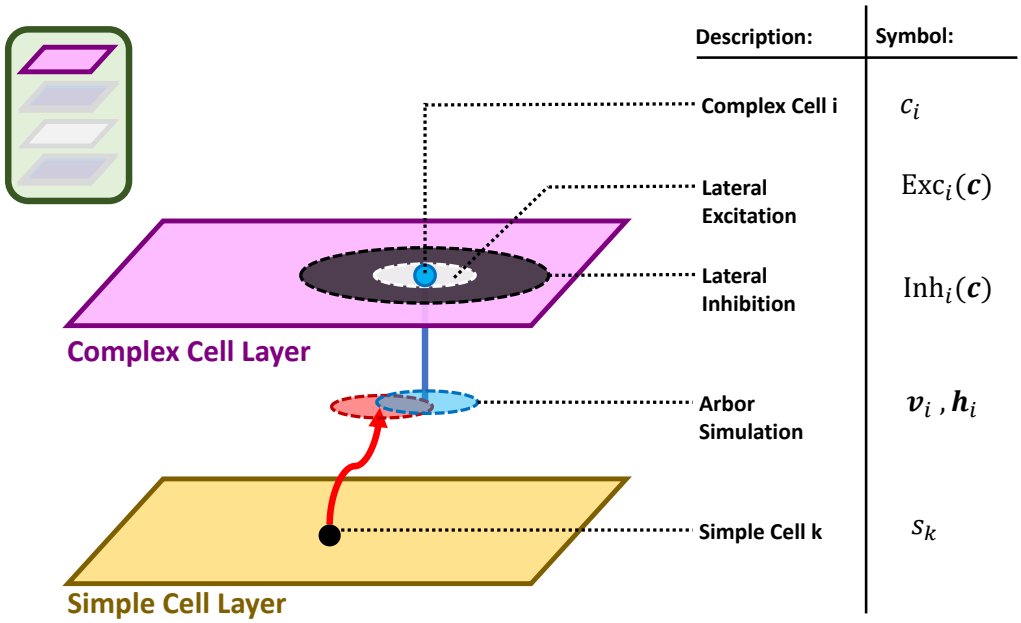

Figure 4.8. Diagram of the Complex Cell Layer

This figure shows the structure and interactions of the complex cell layer. The YELLOW layer represents a lower simple cell layer. The RED and BLUE disks denote the connection between simple cell neuron k and complex cell i mediated by the interaction of their axonal (RED) and dendritic (BLUE) arbors respectively in an arbor simulation. The PINK layer is the complex cell layer. The WHITE disk indicates that simple cell i laterally excites its neighboring neurons. The BLACK disk indicates that simple cell i laterally inhibits its neighbors over a larger distance than it excites. Combined with hebbian learning and lateral interactions, the complex cell layer will competitively learn groups of simple cells which tend to belong to the same temporal trajectories, and which hopefully represent variation due to minor image transformations.

running mean estimate of its typical activity during training (see Eq. 4.28). For comparison, this is very similar to a type of batch-normalization.

4.3.4.1 Complex Cell Equations in Detail

4.3.4.1.1 Complex Cell Window Functions Much like the simple cell layer the input window and weights of the complex cell layer are managed by an arbor layer model. $\mathbf{v}_i$ in Eq. 4.18 gives the vector of current vector of input simple cell activities to complex cell c_i, where s is the vector of all simple cell activities. $\mathbf{v}_i$ in Eq. 4.19 gives the current weight vector for complex cell c_i, where $\mathbf{V}$ is the matrix of all weights for the complex cell layer.

$$\mathbf{h}_i = \text{Arbin}_{c_i}(\mathbf{s}) \tag{4.18}$$

$$\mathbf{v}_i = \text{Arbw}_{c_i}(\mathbf{V}) \tag{4.19}$$

4.3.4.1.2 Complex Cell Lateral Interactions Similar to the simple cell layer the complex cells compete with one another using lateral interactions. $\text{Exc}_i^c(\mathbf{c})$ in Eq. 4.20 gives the net excitatory input from other complex cells to complex cell c_i, with exc^c as the width of influence. $\text{Exc}_i^c(\mathbf{c})$ in Eq. 4.21 gives the net inhibitory input from other complex cells to complex cell c_i, with inh^c as the width of influence. c_i^{lat} in Eq. 4.23 gives the combined effect lateral inhibition and excitation on complex cell c_i. $\mathbf{c}$ is the complete activation vector for all of the complex cells.

$$\text{Exc}_i^c(\mathbf{c}) = \alpha_{\text{exc}}^c \sum_j g(\mathbf{x}_i - \mathbf{x}_j, \sigma_{\text{exc}}^c) c_j \tag{4.20}$$

$$\text{Inh}_i^c(\mathbf{c}) = \alpha_{\text{inh}}^c \sum_j g(\mathbf{x}_i - \mathbf{x}_j, \sigma_{\text{inh}}^c) c_j \tag{4.21}$$

$$\mathrm{EI}_i^c(c) = \mathrm{ReLu}\left(\mathrm{Exc}_i^c(\mathbf{c}) - \mathrm{Inh}_i^c(\mathbf{c})\right) \tag{4.22}$$

$$c_i^{\mathrm{lat}}(\mathbf{v}_i, \mathbf{h}_i) = \mathrm{EI}_i^c(\mathbf{v}_i^T \mathbf{h}_i) \tag{4.23}$$

4.3.4.1.3 Complex Cell Trace Dynamics The hallmark of trace learning is that a temporal average of activations is used to create a target for learning. $c_i^{\mathrm{trace}}(t)$ in Eq. 4.24 is the temporal average of a complex cell's activation after lateral interaction c_i^{lat}, where β_c is the speed of the running average. $\mathbf{h}_i^{\mathrm{trace}}(t)$ in Eq. 4.25 is the temporal average of a complex cell's current input window $\mathbf{h}_i(t)$, where β_h is the speed of the running average.

$$c_i^{\mathrm{trace}}(t) = c_i^{\mathrm{trace}}(t-1)(1-\beta_c) + c_i^{\mathrm{lat}}(t)\beta_c \tag{4.24}$$

$$\mathbf{h}_i^{\mathrm{trace}}(t) = \mathbf{h}_i^{\mathrm{trace}}(t-1)(1-\beta_h) + \mathbf{h}_i(t)\beta_h \tag{4.25}$$

4.3.4.1.4 Complex Cell Learning Dynamics The complex cell learning dynamics use an additive re-normalization scheme with two steps. First, $\mathbf{v}_i^{\mathrm{targ}}$, the current learning target for complex cell c_i, is generated by multiplying its current trace activity c_i^{trace} by the unit normalized version of the trace of its input window activations. Second, the target $\mathbf{v}_i^{\mathrm{targ}}$ is added to the current weights after being multiplied by the learning rate σ_{lrate}, and the weight vector is normalized such that it sums to one.

$$\mathbf{v}_i^{\mathrm{targ}} = c_i^{\mathrm{trace}} \left(\frac{\mathbf{h}_i^{\mathrm{trace}}}{\|\mathbf{h}_i^{\mathrm{trace}}\|} \right) \tag{4.26}$$

$$\mathbf{v}_i(t+1) = \frac{\mathbf{v}_i(t) + \sigma_{\mathrm{lrate}} \mathbf{v}_i^{\mathrm{targ}}(t)}{\sum_j v_{ij}(t) + \sigma_{\mathrm{lrate}} v_{ij}^{\mathrm{targ}}(t)} \tag{4.27}$$

4.3.4.1.5 Complex Cell Output Finally, the output of the complex cell layer is given by $c_i^{\text{out}}(\mathbf{v}_i, \mathbf{h}_i)$, where the AvgNorm operation normalizes each unit to have a mean activation of one, and the Maxpool operation returns the maximum value of its input vector. Note $\odot$ is used to indicate element-wise multiplication.

$$c_i^{\text{out}}(\mathbf{v}_i, \mathbf{h}_i) = \text{AvgNorm}(\text{Maxpool}(\mathbf{v}_i \odot \mathbf{h}_i)) \tag{4.28}$$

4.3.4.2 Complex Cell Demonstration and Observations

Much like the simple cell layer, I tried many versions of the complex cell layer learning dynamics. Again the version selected for this project was chosen for its simple dynamics. Generally, performance and the features learned were qualitatively similar.

Fig. 4.9 shows a demonstration of a complex cell layer learning pools of simple cells. In this demonstration notice that the orientation columns learned by the complex cell layer are significantly wider than the orientation columns of the simple cell layer, suggesting that they aggregate simple cells with similar orientation tuning over wide regions of the visual field. Also, based on my observations the learned complex cell pools appear to follow similarly tuned iso-orientation columns in the the simple cell layer.

The structure of these pools is influenced by a complex interplay of input stimuli, temporal speed, and temporal dynamics. Increasing the temporal speed of the environment will increase the width of tuning across position, but will require a slower complex cell learning rate, while applying a threshold to the weights of the complex cells for activation dynamics will narrow the orientation selectivity. Finding a good rule of thumb or a way to automatically set these hyper-parameters will likely take focused study, observation, and possibly the development of a special hyper-parameter search algorithm. A future study to examine these issues is proposed in Chapter 6 Sec. 6.4.5.1.

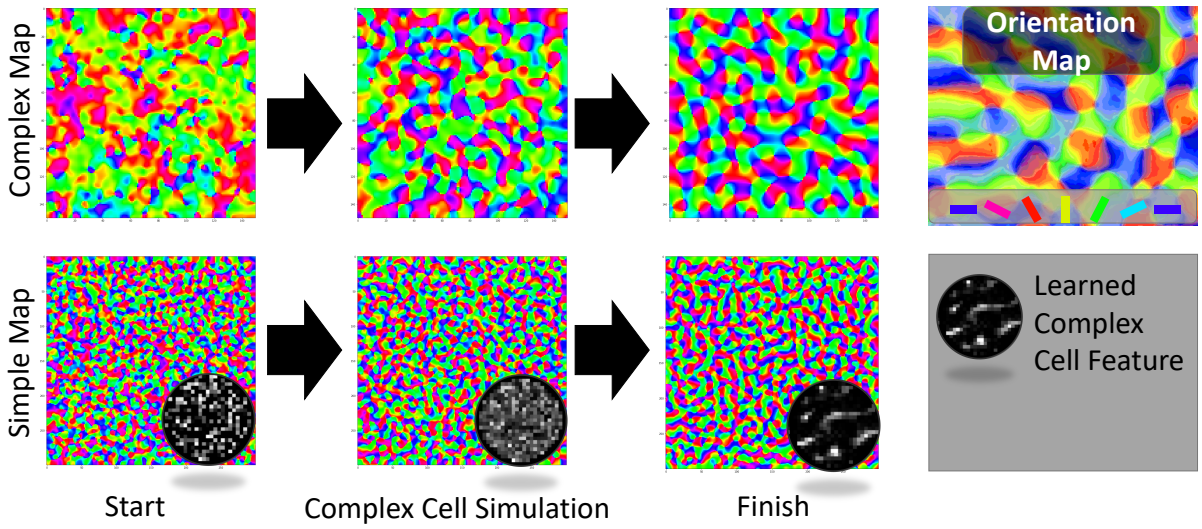

Figure 4.9. Demonstration of Complex Layer Simulation
The TOP row shows the evolution of orientation selectivity across an entire 50x50 complex cell layer, that takes input from the 100x100 simple cell layer that develops simultaneously as shown in the BOTTOM row. The BLACK and WHITE DISK shows the development of a single complex cell's pool, where the lighter areas correspond to simple cells that belong in its pool. For the orientation maps each color in the top row represents neurons which are sensitive to a particular orientation of edge contrasts, see the legend on the TOP RIGHT. TOP RIGHT shows an example of a typical orientation map structure adapted from Bosking et al. (1997). Note that the orientation columns of the complex cell layer are much wider than the orientation columns of the simple cell layer, meaning that they aggregate simple cell edge detectors over large regions of the input visual field.

4.4 Review of Trace Rule Learning for Complex Cells in Depth

Trace rule pool learning was first introduced conceptually by Földiák (1991), but many researchers have developed their own versions of it (Rolls and Stringer, 2001; Wallis, 1996; Einhäuser et al., 2002; Michler et al., 2009). Mechanically, trace rule pool learning was introduced to explain the odd behavior of complex cells in early visual cortex. Namely, many complex cells appear to react in a phase invariant manner when seeing local edge contrasts and/or Gabor filter batteries, while maintaining selectivity to orientation (see Fig. 4.10B). In contrast, simple cells react in a fairly straightforward manner, with a peak activation for

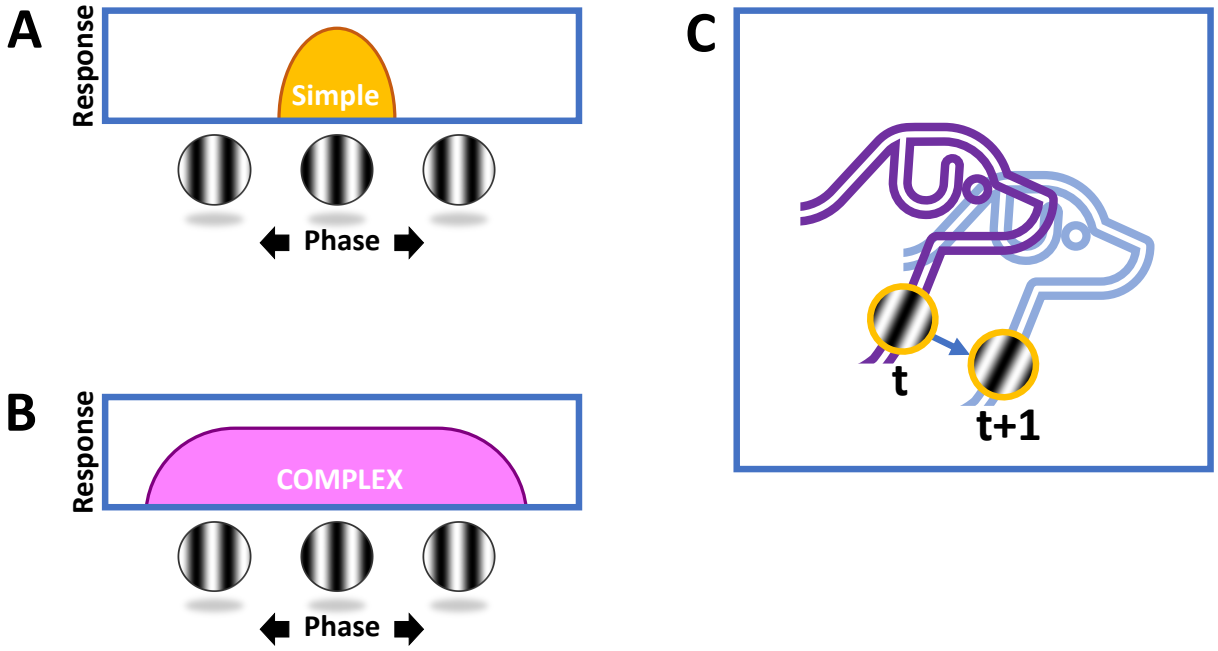

Figure 4.10. Simple and Complex Response Properties, and Feature variation due to Temporal Image Variation
(A) shows how the response of simple cells is narrow over the phase and position of input edge stimuli or contrast stimuli. (B) shows how the response of a complex cell is roughly invariant over a wide range of positions and phases for input edge or contrast stimuli. (C) shows how small image features that belong to objects will vary mildly in there presentation as the view of the object they belong to varies over short time intervals, and how this variation does not change the semantic meaning of the image feature within the image context (i.e. the edge still belongs to a dog).

a specific phase, location, and orientation for edge stimuli (see Fig. 4.10A). Simple cell like unit response properties can be learned with artificial neurons that feature simple Hebbian learning rules, and rectified linear units. Complex cells response properties are generally more difficult to learn and represent.

Computationally, complex cells are often thought to reduce the differences present in neural representation of an object over the many ways it can present on the retina, when its position, scale, rotation, posing, and other viewing factors are changed. This would allow semantically identical stimuli to seem more similar under simple image variations (Hubel and

Wiesel, 1962). For example, pooling edge across small shifts in position would be very useful for classification. The orientation of the lines and edges and their conjunctions would still be clear in relation to one another, but the relative location of each feature would be somewhat tolerant to visual translation. When hierarchically compounded, local tolerances to translation and other image variability can greatly improve a system's recognition performance, as DCNNs have demonstrated.

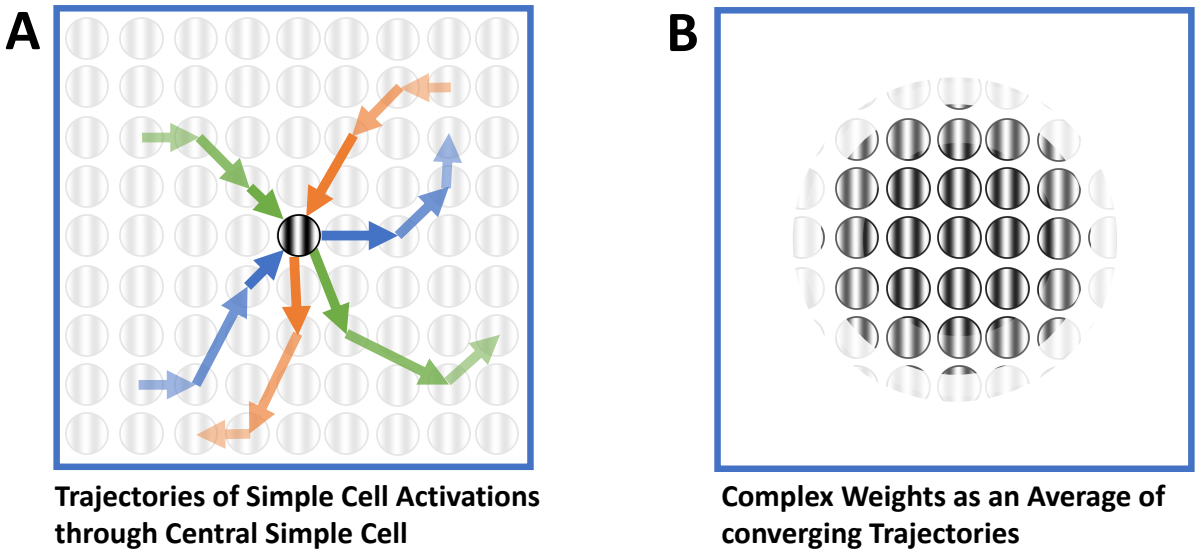

Figure 4.11. (A) Trajectories of Simple Cell Activation and (B) Complex cell weights over Input Simple Cells
(A) shows how the sparse activations of individual simple cells will follow trajectories of serially active units, as image features move/vary in the input image over time. Each contrast grating represents a simple cell. Importantly, many different trajectories (BLUE, GREEN, ORANGE ARROWS), due to temporal image variation will tend to pass through each simple cell. (B) shows how an average over all of the trajectories passing through a single simple cell can be used as the weights for a complex cell's pool. The weight of each simple cell is represented by its visibility. Importantly, this weighting of simple cells likely represents the effects of small image perturbations rather than changes affecting the semantic meaning of the image.

The hierarchical theory of cortex (Hubel and Wiesel, 1962) suggests that complex cells achieve their more tolerant tuning curves by "Pooling" the activations of multiple simple cells (see Martinez and Alonso (2003) for a review of this theory and its competitors). This can be done by a divisive normalization, maxpooling, or average pooling over the inputs to a complex cell. However, this theory does not explain how the memberships of these pools are learned (if they are even learned in the first place).

An image feature such as an edge contrast presented to the retina, will tend to move across the retina due to several factors: body motion, visual saccades, and the movement of the object to which it belongs (see Fig. 4.10C). Because of this, the visual system has access to statistical information about which visual transformations of a feature do not change its overall meaning or relationship to other features (i.e. a series of edges that belong to a dog, still probably belong to that dog after being shifted slightly due to a small movement or saccade) (see Fig. 4.10C). These statistics go beyond just shift, as the temporal environment can give information about rotation, scaling, and more complex feature transformations for more complex features. However, the mechanical nature of the eye and the properties of the real world would suggest that simple image shifts would be the most common form of temporal variance and would need to be addressed first. Trace rule pool learning methods allow complex cells to learn about these visual transformation statistics from temporal observations.

Under trace rule pool learning, a complex cell uses a running temporal average of its own activation for use with hebbian learning. The complex cell can also use running temporal averages of their inputs for the purposes of learning. If the speed of learning is set correctly, the weights learned by the complex cell will be an average of all the trajectories of simple cell activations over time that tend to activate some central group of simple cells (see Fig. 4.11). If used with an average pool or weighted maxpool operation the activation of this neuron should resemble the activation of real complex cells in the right statistical environment.

Some form of cap on the weights may also be helpful in order to encourage a flatter peak response over common feature variability, and a threshold before a weight acts as non-zero would allow for a sharper selectivity of less common feature variability.

4.5 Input Pre-Processing for ICL Models: LGN like processing

4.5.1 Difference of Gaussians and Low-Pass Spatial Filtering

Before being fed to the ICL model proper visual images need to be processed in a manner similar to how the *Lateral Geniculate Nucleus* (LGN) process stimuli. In mammal brains, visual images pass from the retina to the LGN and then to V1 and other cortical/sub-cortical areas. One of the main ways that the LGN encodes information is by using on-center off-surround cells. The receptive field of these cells resembles a donut, where the center of the donut excites the neural firing and the outer ring of the donut inhibits neural firing (see Fig. 4.12). For completeness, the LGN has both on-center off-surround cells and off-center on-surround cells with some form of rectification. However, for modeling purposes only one set may be necessary, as is the case for this version of the ICL model.

The receptive fields of on-center off-surround cells are often modeled as a difference of 2-dimensional Gaussian filters (see, Fig. 4.12). These filters are also sometimes called Mexican Hat filters after their resemblance to the sombrero de charro, a traditional hat of Mexico's horsemen (see, Fig. 4.12D). A special property of difference of Gaussian filters is that they isolate features in an image associated with a specific spatial scale (see Fig. 4.13). This is directly analogous to the way in which sine waves and wavelet filters can be used to isolate the contribution of different frequencies to a particular one dimensional signal or sound. This property stems from the fact that 2D Gaussian filters eliminate information below a specific spatial frequency. Thus a Difference of Gaussian filters act as simple band-pass filters (see Fig. 4.13).

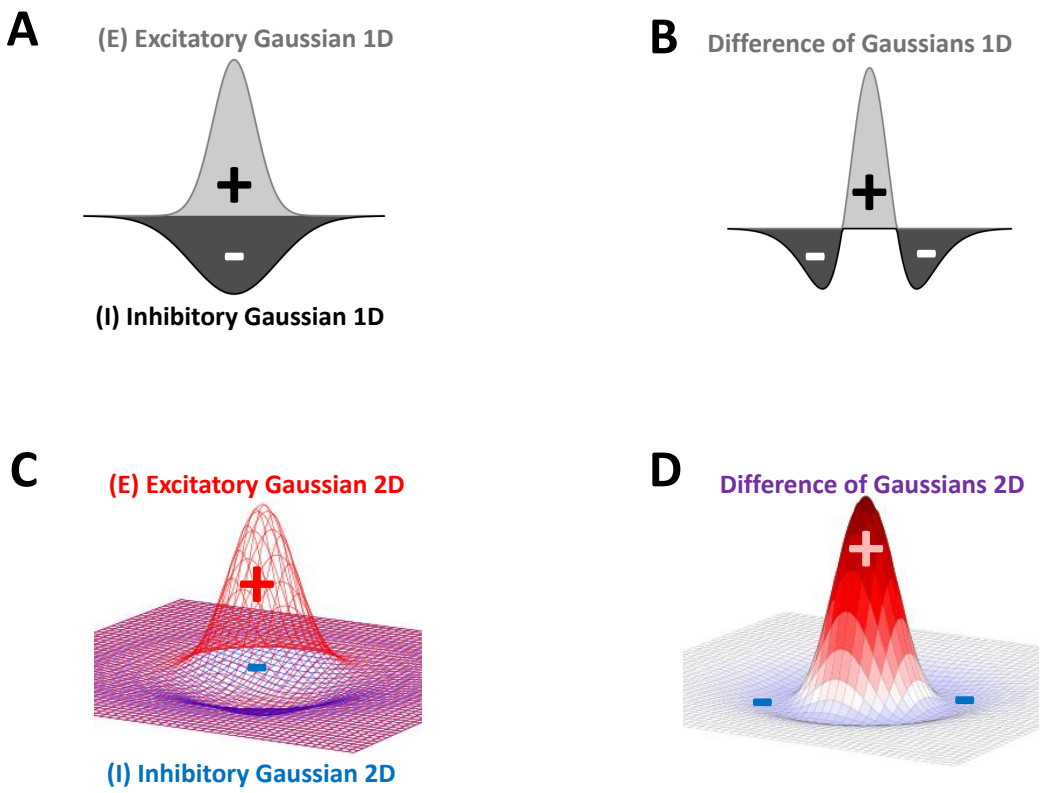

Figure 4.12. Composition of Difference of Gaussian Filters
(A) Shows the excitatory (LIGHT GRAY) and inhibitory (DARK GRAY) Gaussian envelopes that added together form a basic 1-Dimensional difference of Gaussian filter. (B) shows the combined envelope of a standard 1-Dimensional difference of Gaussian filter used for image processing. (C) shows the excitatory (RED) and inhibitory (BLUE) envelopes of a 2D Gaussian filter. (D) shows the combined envelope of a standard 2-Dimensional difference of Gaussian filter.

For the ICL models in this paper, I actually used a filter similar to a difference of Gaussian band pass filter, but with notable differences. For computer images, the width of a pixel already acts as an upper limit on high-frequency information. As such, I used a simple high-pass filter instead instead of a band pass filter. This is accomplished by subtracting an image across all channels by a version of itself that has passed through a Gaussian filter (see Eq. 4.29). As a receptive field, this looks like a single on pixel surrounded by a negative

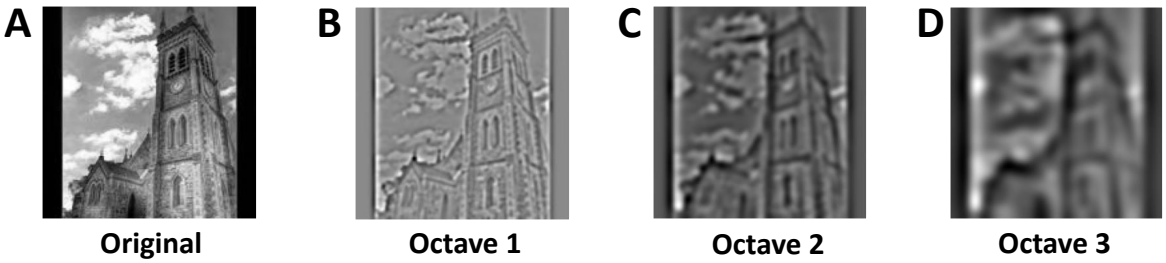

Figure 4.13. Using DoF filters to Isolate Spatial Frequencies
(A) shows an image for the ImageNette dataset. (B) shows the contribution of a band of frequencies to the original image in A. These frequencies were isolated by passing a Difference of Gaussian (DoF) filter over the original image. This filter had an excitatory width of 1 and an inhibitory width of 2. (C) shows the contribution of the next lowest octave of frequencies (1/2 the frequency range present in C). This was isolated by passing a DoF filter similar to the first one but scaled up by a factor of 2. (D) shows the next lowest octave of frequencies after the one present in C. This was accomplished using a filter similar to the first one but scaled up by a factor of 4. For reference, this is the same kind of octave seen in music, as an octave difference between notes corresponds to either a doubling or halving of a note's frequency.

Gaussian. In practice, this acts like a difference of Gaussian filter for selecting the highest available frequencies in a discrete image.

For the following equation Z is an image, G_0 is a Gaussian filter, and $*$ the discrete convolution operator. Z_{high} is the result of the high-pass operation.

$$Z_{\text{high}} = Z - G_0 * Z \tag{4.29}$$

4.5.2 Layer-wise Gain Control

The LGN and retina have other dynamics that enhance their representations of images. One such mechanism is called local gain control. In natural images, when difference of Gaussian processing is done, some regions of the image will have large activation contrast while others will have very small activation contrasts. Most cortical models are quite sensitive to the

magnitude of local contrasts between input units. This means that locations with smaller contrasts tend to be poorly represented, despite carrying important visual cues. To combat this, gain control scales local contrasts competitively, so that regions of slight contrast will be enhanced and regions of large contrast will be reduced (see Fig. 4.14).

From a computational perspective, this is quite useful. (Stevens et al., 2013) demonstrated that adding local gain control to the LGN inputs for a V1 model greatly stabilized its feature learning and tolerance to different contrast levels. Further, local gain control helped stabilize learning in our implementations of Deep ICL models when used as a normalization process between ICL modules. Here local gain control played an analagous role to "batch normalization methods" in DCNNs, by keeping the range of activation values within a semi-consistent distribution.

In the following equation, G_1 refers to a Gaussian filter of a lower spatial frequency than G_0 used in Eq. 4.20 to generate the high-pass image Z_{high}. ϵ_{gain} is a constant that prevents the contrast adjustment from boosting values that are too close to zero. Note that the division operator and the absolute value function used here are element-wise operations.

$$Z_{\text{gain}} = \frac{Z_{\text{high}}}{G_1 * |Z_{\text{high}}| + \epsilon_{\text{gain}}} \tag{4.30}$$

4.5.3 Color Pre-Processing

LGN like pre-processing for gray-scale inputs is relatively straightforward, but color processing has several unique demands. Firstly, color is processed in an opponent fashion in the LGN, with different channels inhibiting one another. Secondly, most of the color gradient information is contained within the lower-frequency bands of an image. And third, because color contrasts correspond to smaller deviations over larger portions of the image, color information tends to have a lower variance than luminance information on a per-pixel basis. Together these qualities of color contrasts in images means they need to be processed in a

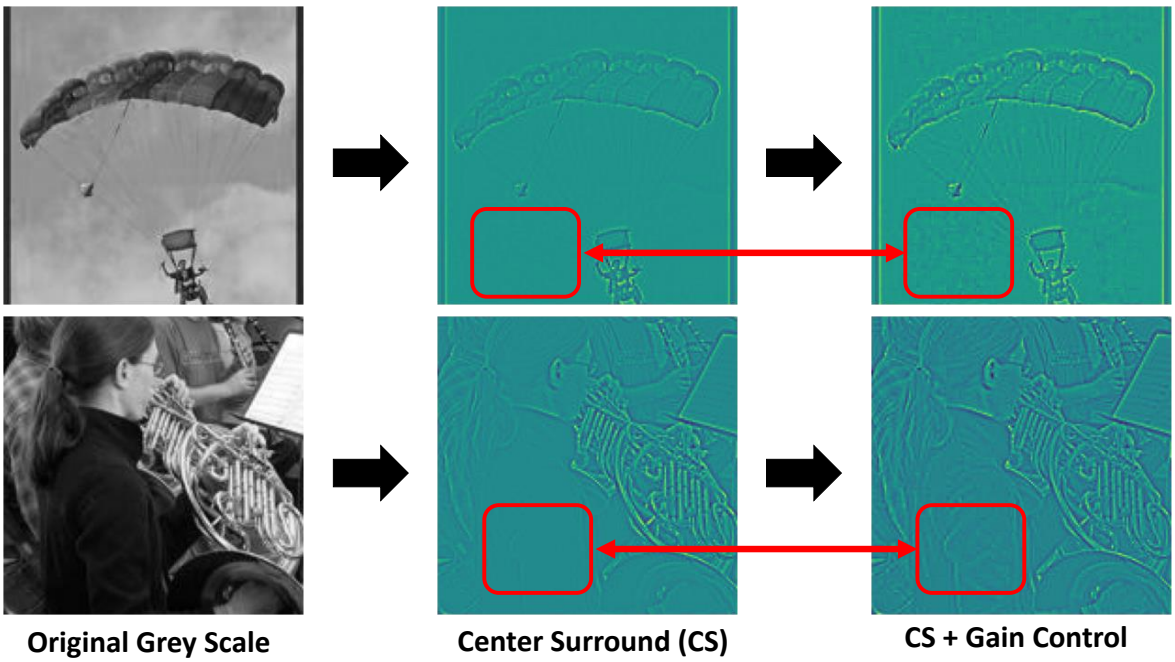

Figure 4.14. Enhancing Low Contrast Areas with Local Gain Control
This figure shows the effects of local gain control in adjusting the activations of on-center off-surround cells. The LEFT set of images show original images from the ImageNette dataset. The MIDDLE image shows the output of on-center off-surround cells. The RIGHT images are the result of the local gain control operation modulating the activity of the on-center off-surround cells. The RED boxes highlight that areas of low contrast in the on-center off-surround images are boosted in the gain control images, allowing for the perception of additional image features.

slightly different manner in order to encourage the first level of an ICL model pay attention to color contrasts.

I developed a simple method to encourage ICL models to give more representation to color contrasts in their early layers, called the color splice method. In the color splice method, the inputs are divided between normal High-Pass + Gain Control inputs tuned to pick up on high-frequency contrasts across RGB, which will tend emphasize luminance contrasts, and inputs that use a lower-frequency cutoff where the RGB channels are in opposition with one another using a similar setup to the local gain control process, which will tend to emphasize color contrasts. These two types of inputs are then spliced together in a single input image

where every fourth pixel in a bloc of four is a low-frequency color emphasizing unit, and the rest are the higher frequency luminance emphasizing units, see Fig. Finally, the activations of the lower-frequency units are amplified to promote color contrast learning. 4.15.

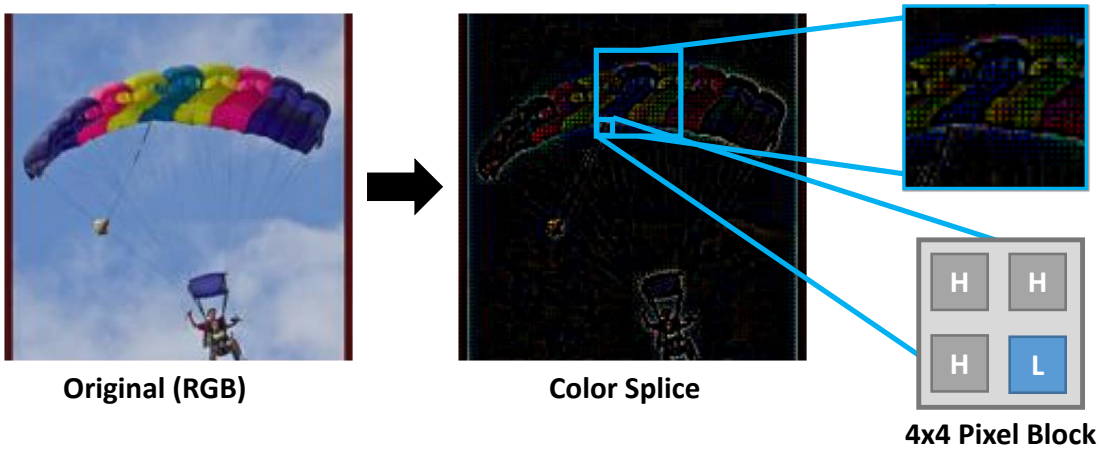

Figure 4.15. Enhancing Color Input using Color Splicing
This figure shows how the color splice operation can enhance the visibility of color contrasts in an image. On the LEFT an image from the ImageNette dataset is shown. In the MIDDLE the result of the color splice operation is shown. The UPPER RIGHT image shows a blown up portion of the color splice image, highlighting that every 4th pixel carries clear color contrast information, while the neighboring pixels appear black and white. In fact, the each pixel in the color splice image is color sensitive, however, color contrasts vary at a slower rate and with a lower amplitude than luminance contrasts. To account for this, every 4th pixel in a block of pixels is sensitive to low frequency information and is also boosted by some fixed amplification constant. This is useful for ensuring that color contrast information will be represented adequately alongside luminance contrast information in the first layer of representation for ICL-like models.

4.6 Model Variations

In this dissertation we explored three major variations on the ICL module design: simple cells only, simple cells + axonal learning, and fixed pools.

In the **Simple Cells Only** variation the the complex cell layer and its associated axonal layer were removed entirely. Further, the initialization of the arbor layer was fixed to be a proper retinotopic mapping and left constant.

In the **Simple + Axonal** variation the the complex cell layer and its associated axonal layer were removed entirely as with the last variation, but the simple layer's associated arbor layer was left unchanged from the full model.

In the **Fixed Pools** variation the only modification from the full model was changing arbor layer associated with the complex layer to be fixed at a retinotopic initialization state.

The reason I used these different versions of the ICL model was to better examine the relative contributions each component of the model was making to performance and feature development. Generally, in the Simulation Study portion of the dissertation I started with the most basic model (Simple Only), then progressed to the Simple+Axonal model, then moved to the full ICL model. The Fixed Pool variation was mostly used internally for testing and to speed up prototyping.

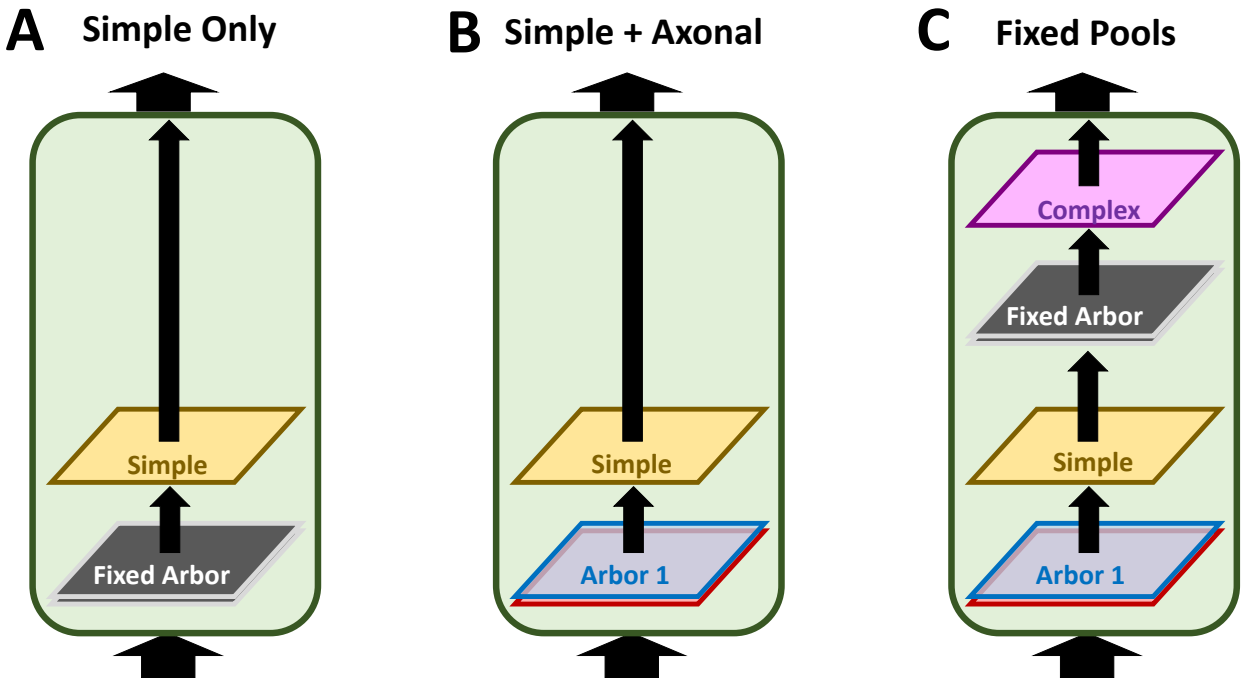

Figure 4.16. Variations of the ICL Module

This figure shows different configurations of the ICL module architecture. (A) shows the Simple Only variation which only uses the the simple cell layer and a fixed arbor simulation initialized to a simple retinotopic mapping. (B) shows the Simple + Axonal variation of the ICL model, where the simple cell layer has learned connections via the arbor layer simulation. (C) shows the Fixed Pool variation of the full ICL module, in which the arbor layer feeding into the complex cell layer is held fixed and initialized to a default retinotopic mapping.

CHAPTER 5

SIMULATION STUDIES OF THE INTEGRATED CORTICAL LEARNING MODEL

5.1 Introduction

5.1.1 Rational

From a computational modeling perspective, Deep Convolutional Neural Networks (DCNNs) have broken new ground in enabling computers to recognize objects and make inferences from visual images (Krizhevsky et al., 2012). Also, DCNNs are considered the state-of-the-art cognitive-neuroscience methodology for modeling higher-order object identification in the ventral visual cortex (Yamins and DiCarlo, 2016b). Both lines of research suggest DCNNs are potent models for understanding the computational principles of human object recognition. Historically many of the core mechanisms of DCNNs were originally directly motivated by attempts to model cognitive functions in the visual cortex (Fukushima, 1980). However, recent research suggests DCNNs possess serious shortcomings both as computational models of human vision and as potential models of the visual cortex, and, further several core mechanisms of DCNNs are highly implausible from a neuroscience perspective even at an abstract level. Such shortcomings support the consideration of alternatives to DCNN modeling approaches.

Here, the generic terminology DCNN refers to a large generic and widely used class of deep convolutional neural networks described in Yamins and DiCarlo (2016b) and Krizhevsky et al. (2012).

This chapter will examine if the more neurally conservative *Integrated Cortical Learning* (ICL) model, introduced in Chapter 4, addresses three of the important shortcomings of DCNNs as models of human visual cortex: 1) their lack natural analogs of cortical maps; 2) their lack of biologically mechanisms in making predictions about unit specialization; and 3)

their reliance on biologically implausible supervised learning and global information sharing. These issues are detailed more below.

5.1.1.1 DCNNs are Not Designed to Model Cortical Maps

The human cortex is a two-dimensional sheet of neural tissues. Further, response selectivity for neurons varies continuously across this cortical surface. On a large scale, the physical organization of this response selectivity forms contiguous regions of the cortex that code for related features and categories (Bednar and Wilson, 2016). The physical organization of feature selective neurons within a cortical area is called a *cortical map*. Because cortical maps mostly group features and attributes that are implicitly similar, they provide insight into the structure of the feature space learned by a cortical area. As such, the study of cortical maps is critical to neuroscience.

However, there is a serious problem in cortical map modeling, in that there is a lack of cortical map models that can perform real-world sensory discrimination tasks. This would seem to be a natural fit for DCNNs, which feature strong real-world discrimination performance, but DCNNs do not have a natural analog to the cortical maps developed in visual cortex.

While DCNNs use two-dimensionally organized convolutional feature maps, they segregate these features into separate hardcoded channels with a fixed topology (Fukushima, 1980; Krizhevsky et al., 2012). In contrast, the primate cortex is highly-plastic and maps all its features to the same cortical surface competitively (Bednar and Wilson, 2016). As a further disconnect, DCNNs typically use multiple fully-connected layers, which have no physical topological organization, for their top layers. These fully connected layers are even less appropriate as stand-ins for cortical maps. As architectures, contemporary DCNNs are simply not designed to model cortical maps. From a neuroscience perspective, it would be useful for a model to make explicit predictions regarding the structure of high-level visual cortical maps.

The ICL architecture proposed in Chapter 4 features cortical map mechanisms at each of its representational stages. These mechanisms include self-organizing maps to represent the simple cell layer and complex cell layers (Ch. 2), and axonal plasticity maps to control the connectivity between layers (Ch. 3). Each of these mechanisms encourages continuous representation of concepts and features across the simulated cortical sheet. If the ICL model develops high-level cortical maps, then it could be a useful tool with which to study many poorly understood phenomena, such as how and why functional areas form, and howthe organization of high-level feature units arises. Further, this would allow researchers to better understand the relationship between high-level feature spaces and their physical manifestation on the cortical sheet.

5.1.1.2 The Unit selectivity of DCNNs appears to be spurious.

Researchers often conceptualize neural representations on two levels, namely, the unit-level and the space-level. A *unit-level* analysis examines the response properties of individual neurons or the receptive fields of individual neurons. A *Space-level* analysis examines the pattern of responses generated by a collection of neurons, where the vector of their combined activation levels exists in a geometric space. Correspondingly, a unit-level analysis examines the unit-level representation of a system, and a space-level analysis examines the space-level representation of a system.

Computational neuroscientists often compare representations in the high-level ventral cortex with high-level layers of DCNNs at both the space and unit-levels. However, several recent analyses of high-level DCNN units suggest their unit-level encoding may not be that meaningful (Szegedy et al., 2014a; Parde et al., 2021). Furthermore, these studies question the conclusions drawn from the unit-level analysis of the brain. In neuroscience, a neuron's *selectivity* and *preferred stimulus* are conventionally used to probe its representation within a neural system. Unit selectivity is widely thought of as an indirect measure of the specialization or role a neuron plays in information processing. However, these recent computational

results can be thought of as leading to two competing hypotheses regarding whether the apparent selectivity of neurons and presence of preferred stimuli in high-level visual areas truly indicate functional unit specialization.

The first hypothesis is that high-level ventral stream neurons are selective and specialized. Studies of the organization of the visual cortex in primates show that the receptive fields of high-level ventral stream neurons are highly selective for semantic categories and attributes, even if they are not so-called "grandmother cells" (Grill-Spector and Malach, 2004; Grill-Spector and Weiner, 2014). For instance, the high-level ventral stream has multiple clusters of neurons that tend to be roughly selective for faces, bodies, places, different objects, visual wordforms, and attributes of these categories (Grill-Spector and Weiner, 2014). These results suggest that neurons in high-level ventral cortex serve specialized roles in representing stimuli.

The second hypothesis is that high-level ventral stream neurons appear selective but do not serve a functionally specialized role in stimulus representation. Szegedy et al. (2014a) showed that high-level DCNN neurons appear specialized using traditional methods, such as examining maximally activating stimuli and selectivity, but that their selectivity and preferred stimuli seem indistinguishable from those of a randomly chosen linear combination of the responses of high-level DCNN neurons. Essentially, this means that current measures like selectivity may and presence of preferred stimuli may not be a good indicators of specialization. Such findings and others have led several authors (Parde et al., 2021; Szegedy et al., 2014a) to suggest DCNNs may encode most of their semantic information at the space-level and that unit selectivity is possibly spurious. Further, they suggest that some of the apparent selectivity in high-level visual cortex could be spurious as well.

A more neurally plausible alternative to DCNNs could give differential predictions regarding unit selectivity and specialization. The DCNN architecture's preference for developing functionally unspecialized units may emerge for architectural reasons that have little to do

with neuroscience. Chiefly, DCNNs lack any of the biological constraints which might push actual neural representations to learn more functionally specialized unit representations. Without these constraints, DCNNs will naturally learn unspecialized space-level rather than specialized unit-level representations as these representations are more computationally efficient for encoding categories (Foldiak, 2003). Further, it is unclear that including such constraints would confer any computational benefits to the DCNN architecture. However, the lack of these constraints casts doubts on whether the unit-level representations of DCNNs are useful for making predictions about the specialization properties of high-level neurons in actual ventral visual cortex.

The ICL architecture proposed in Chapter 4 contains biologically motivated learning constraints that may cause it to learn specialized unit representations in its highest layers, such as trace learning (Ch. 2), self-organizing map learning (Ch. 2), and axonal plasticity modeling (Ch. 3). Trace learning makes it the goal of individual neurons to learn specialized representations of parts of the temporal environment. Self-organizing map learning forces neurons to compete and avoid representing concepts too redundantly. Axonal plasticity forces each neuron to use limited inputs that are already highly related, which should encourage further specialization.

Current evidence and biological theories of primate visual representation in ventral cortex (Grill-Spector and Malach, 2004; Grill-Spector and Weiner, 2014) support a more specialized view of neuronal representation in the brain. If the ICL architecture learns a specialized unit representation, then it would give support for Hypothesis 1, that the apparent selectivity of neurons in the brain reflects a real functional specialization of those neurons, and that the additional biological learning constraints used by ICL may be the determiner for this phenomenon.

For clarification, when talking about high-level layers in DCNNs, I are typically referring to layers before the classification layer with a specific focus on the penultimate or second to

the last layer. For reference, the classification layer typically includes both a modifiable linear transformation followed by a softmax transformation (Goodfellow et al., 2016). Because the output of the final linear transformation and the outputs of the softmax transformation are directly constrained to approximate match a simple categorical output (i.e. one-hot coding), these representations are not viewed as meaningful to compare to neural representations. The inputs to the final modifiable linear transformation are the chief representation of interest because they are not explicitly constrained to follow a simple categorical representation and are instead influenced chiefly by the computational demands of the task and the learning architecture.

5.1.1.3 DCNNs Rely on Supervised Learning and Global Coordination

Most computationally successful DCNN models use a strong form of supervised learning that requires massive numbers of images with high-level semantical labels to train effectively (Yamins and DiCarlo, 2016b). In contrast, humans learn extensively with large amounts of unstructured and unsupervised experience, without the need for huge amounts of externally applied labels. From a computational neuroscience standpoint, more biologically plausible unsupervised learning algorithms need to be developed that achieve high performances (Yamins and DiCarlo, 2016b).

Researchers have actually proposed many unsupervised DCNN learning algorithms, but these models largely fail to fit measures of high-level representation in the ventral stream. More specifically, unsupervised models do not predict the similarity matrix between items computed using the activation space of high-level ventral cortex, and further they usually suffer from weaker overall performance (Khaligh-Razavi and Kriegeskorte, 2014; Yamins and DiCarlo, 2016b). In general, supervised learning models have been far more successful at fitting high-level representations seen in the ventral stream, despite the likelihood that this amount of supervision is cortically implausible (Khaligh-Razavi and Kriegeskorte, 2014; Yamins and DiCarlo, 2016b).

The learning issues with DCNNs go beyond reliance on supervised learning. In both supervised and unsupervised learning, most DCNNs generate an error signal in their final layer and propagate it back through all of the layers of the network. The results of this error propagation are used to modify the connection weights within each layer. This learning method is called *backpropagation*. Although researchers continue to try to pose neurally plausible implementations of this scheme (Scellier and Bengio, 2017), it is hard to ignore that backpropagation requires the close coordination of many neural layers that are not directly linked. As a further problem, the core mechanism of DCNNs involves sharing the weights of one neuron in a layer's feature map with all the neurons in that map in a method called *convolutional weight-sharing*. As such, neurons within a convolutional layer communicate their weights directly over the full layer laterally in a way that is not supported by neuroscience.

Importantly, the notion that backpropagation and weight sharing are not neurally conservative is uncontroversial among even prominent researchers using these techniques (Yamins and DiCarlo, 2016b). Generally, these methods are tolerated as abstract stand-ins for more local but neurally plausible processes.

In a further expansion of this point, DCNNs also use either fixed convolutional connectivity schemes or fully connected connectivity schemes in their later layers. Here connectivity refers to the arrangement of connections between one layer of neurons and another layer of neurons, where neurons that are not connected cannot talk to one another or form synaptic weights. For future reference the set of input neurons that connect to a neuron of interest are that neuron's window. Windows are another way to refer to the connectivity between neurons. As discussed in Chapter 3, neither fixed convolutional windows nor full connectivity is not an accurate portrayal of how the cortex develops its connections and may ultimately limit feature learning. Further, convolutional connectivity and full connectivity likely do not describe connections between higher-level cortical well at all; This problem may compound as DCNNs are tasked with learning higher and higher-order features.

In contrast to DCNNs, the core mechanisms of the ICL model are derived from common and accepted theories in neuroscience, making it neurally conservative. These mechanisms include but are not limited too Hebbian learning, lateral inhibition, trace rules, self-organizing maps, and multi-factor axonal development mechanics. Further, the ICL model only generates learning signals between neighboring layers and neurons, which means it does not coordinate behavior over long distances like DCNNs do. Further still, the inclusion of axonal learning dynamics constitutes a strong move in the direction of neural plausibility, as these mechanisms are almost entirely omitted from the current modeling literature for the ventral visual pathway.

A pivotal challenge of this proposal is to examine if these more neurally conservative mechanisms in the ICL model are sufficient to generate performance comparable to the mechanisms used in standard DCNNs that feature global coordination, supervised backpropagation, and fixed connectivity.

Evaluating the Challenge and Looking Forward. Very few learning algorithms have ever demonstrated a strong ability to learn deep multi-layer neural representations while also achieving high-performance on challenging datasets. Supervised backpropagation, while not particularly cortically plausible, is by far the most successful algorithm for this to date and for a number of reasons. Creating a new class of algorithms based on cortical learning mechanisms that compete with or surpass supervised backprop would be a major achievement, and should be viewed as an aspiration of computational cognitive neuroscience that will require a large amount of research and development in the field. As I will discuss later, the ICL model I developed is a first step, but not the last step in achieving this goal. ICL should be viewed as one attempt to move towards the goal of more cortically plausible deep learning methods, rather than the final word on whether it is feasible or not. As such, I hope that ICL's progress towards this goal can illustrate better ways forward for future cortically inspired deep learning methods.

5.1.2 Overview of Simulation Studies

This project explored how the ICL model deals with the problems of unsupervised learning, unit-level encodings, and cortical maps. In Simulation Study 1, I examined whether the ICL model learns continuous topographic maps of category-selective units similar to high-level cortical map representations. In Simulation Study 2, I examined whether the ICL model learns specialized unit representations or unspecialized unit representations using a method that examines the relationship between the unit-level and space-level encodings within a representation. In Simulation Study 3, I tested if the ICL model could perform unsupervised visual category learning, using common object recognition datasets.

5.2 Simulation Study 1: Cortical Map Development

5.2.1 Introduction

While cortical maps have been a hallmark of neuroscience research (Bednar and Wilson, 2016; Grill-Spector and Weiner, 2014), today's high-performing models largely omit them. For example, the top levels of today's DCNNs are fully connected, meaning that their units have no representation of their physical relation to one another. Because of this, the physical topology of unit selectivity is simply not a relevant point of comparison. However, given the ubiquity of cortical maps in cognitive neuroscience, it would be useful to have a model that featured high-performance like DCNNs and also provided an analog to cortical maps. If the ICL model groups neurons that represent similar categories of objects, like what is found in high-level visual cortex (Grill-Spector and Weiner, 2014), it would mean that ICL may be useful for modeling high-level cortical maps.

The goal of Experiment 1 was to test whether the top-level feature maps of the ICL model exhibit cortical mapping behavior, specifically whether the ICL model's top-level group neurons that encode features which select for similar categories together. To accomplish this

goal, I measured the area of contiguous regions of units that were preferentially selective for each category within layers of the model. Then I tested if this organization could have arisen out of chance. In theory, the ICL model should generate contiguous regions larger than would be expected by chance, if it was displaying high-level cortical map development.

5.2.1.1 Predictions

Given that the ICL model is explicitly a deep multilayer self-organizing map, I predicted that it would develop a high-level category contiguous cortical map structure (i.e. grouping features that tend to select for similar categories together). If this structure was found I planned to do several followup explorations.

For future studies, it would be useful to know how factors like the radius of lateral excitation affect the structure and size of category contiguous regions. I assume that larger radii of lateral excitation will lead to more contiguous maps with fewer discrete regions. This is generally the result observed in low-level cortical map models.

5.2.2 Simulation Methods

5.2.2.1 Object Recognition Dataset: MNIST-F

For this Simulation Study, the ICL model was trained on a temporally augmented version of Fashion-MNIST which contains 28 by 28 gray scale images of 10 categories of clothing items (Xiao et al., 2017). Fashion-MNIST is designed to be a replacement for the original MNIST for the purposes of prototyping and developing new neural networks, because MNIST was generally too easy for modern machine learning techniques to be validated on. Fashion-MNIST should suffice to see if the model is learning interesting cortical map topology.

5.2.2.2 Model Configuration for MNIST-F

For the versions of the ICL model run on MNIST-F, I kept the native resolution of the dataset, which was 28 x 28 pixels gray scale. As pre-processing, I performed a high-pass Gaussian filter operation (similar to on-center off-surround processing), then a gain control operation in order to scale these LGN like inputs. See Chapter 4 Sec. 4.5.1 and Sec. 4.5.2 for more information about LGN like processing for ICL like models.

For the MNIST dataset, I used 4 stacked ICL modules with either a maps resolution of 100 x 100 for each module or a narrowing series of resolutions (100x100, 80x80, 60x60, 40x40) for each module. These setups were used for the ICL-Lat-Optimal and ICL-Map-Optimal model configurations respectively. For this dataset I also explored the Simple-Only, Simple + Axonal, and Full ICL module (referred to as the simple + complex configuration in this chapter) configurations, see Chapter 4 Sec. 4.6 for more information on the different module configurations.

5.2.2.3 Temporal Augmentation

The ICL must be exposed to statistical environments where the images evolve since, unlike DCNNs, the ICL model uses the temporal dynamics of the statistical environment as an important hint for developing its pooling structure. The datasets used in these experiments do not include natural temporal observations. To address this, I introduced artificial temporal relations by applying progressive image transformations such as rotation, scaling, and translation to the individual images to create sequences of images. The beginning state of the input image was pulled randomly from a uniform distribution across rotation, scale, and position. The speed of these transformations was included as a set of special hyper-parameters that controlled the training environment. Further, the probability of switching to an object within the same category or to a new category was included as an additional hyper-parameter.

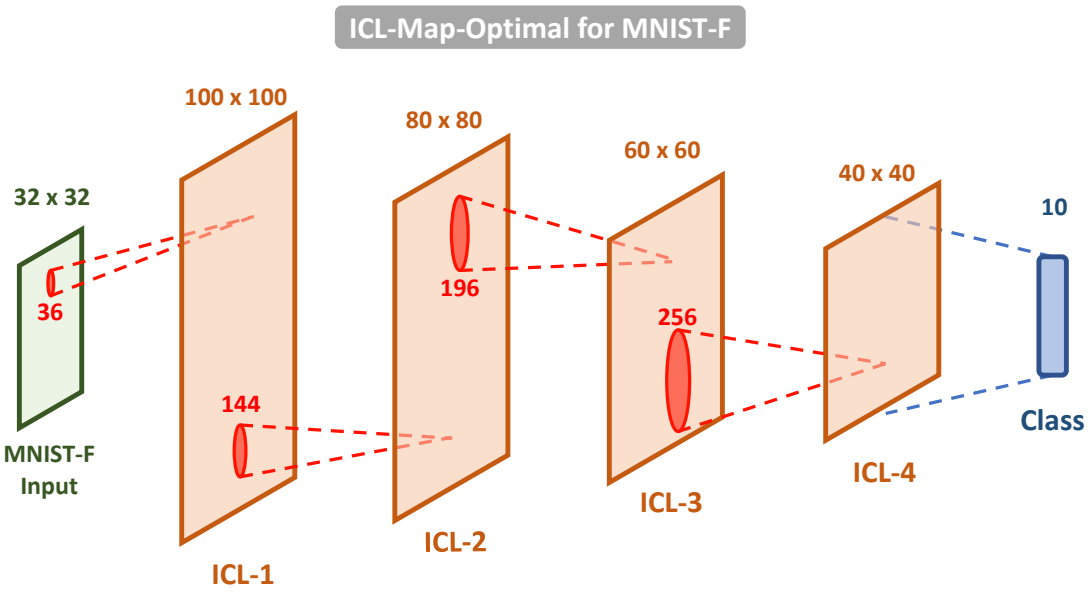

Figure 5.1. Network Diagram: ICL-Map-Optimal: MNIST-F
This diagram shows the layer setup for the ICL-Map-Optimal model trained on MNIST-F. Note that the windows (RED) connecting neurons from one layer to the next are not static given that they will be adjusted by the arbor layer's learning dynamics when axonal learning is enabled, and are only roughly disk-like in practice. The red number next to each window indicates the total number of connections in each window. For the ICL-Map-Optimal model the layer sizes progressively shrink, while the window sizes progressively grow. This increased the scale structures in the high-level cortical maps considerably, but appeared to lower performance considerably.

It is important to note that there was fine-tuning of the temporal augmentation method for each dataset in addition to fine-tuning the structure of the ICL model for each dataset. Finally, I would like to emphasize that the addition of temporal dynamics to a collection of static images was designed to create a more realistic model of the statistical learning environment. In the real world, the statistical environment does not consist of a collection of isolated static images which are then sampled with replacement. Rather, dynamic temporal transformations are an integral part of the statistical learning environment of the real world.

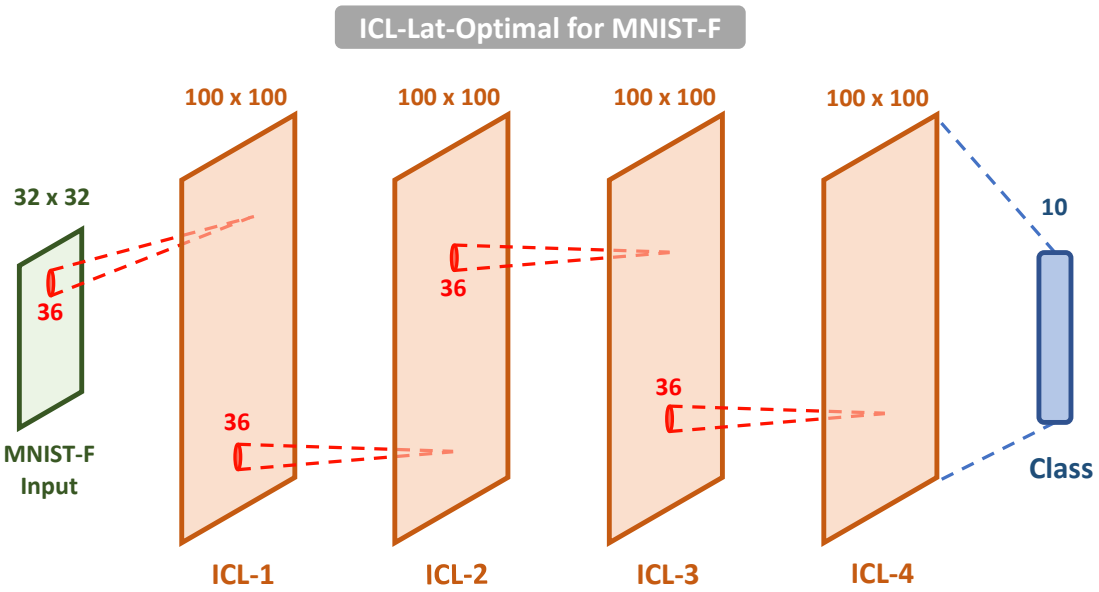

Figure 5.2. Network Diagram: ICL-Lat-Optimal: MNIST-F
This diagram shows the layer setup for the ICL-Lat-Optimal model trained on MNIST-F. For the ICL-Lat-Optimal model the layer sizes and window sizes remain constant. This layer-diagram generally increased performance significantly over the ICL-Map-Optimal model.

5.2.2.4 ICL Training Procedure

The training methods for the ICL model were substantially different from those of DCNNs and or standard deep learning models. See Fig. 4.2 for a graphic overview of the training process.

5.2.2.4.1 ICL: Layer-wise Learning Rules The ICL model was composed of a stacked series of modules, where each module could have 4 component layers: arbor layer 1, simple layer, arbor layer 2, and a complex layer. Each of these layers learned in an unsupervised fashion using only local information, such as the activity of its inputs and the activity of the layers it directly connects to.

The Arbor layers used local gradient descent to move the axonal arbors of pre-synaptic neurons towards the dendritic arbors of correlated or co-activating neurons, while also min-

imizing overlap between axonal arbors and lightly encouraging a rough retinotopy. The objective function for this layer only took activations from the pre-synaptic and post-synaptic cells as input, and no backpropagation learning signals were sent to external layers.

Each Simple layer used a form of laterally competitive hebbian learning that encouraged it to learn a map of neural feature detector units that have good coverage over the distribution of input patterns coming into the layer. The learning method was not implemented using gradient descent, and only uses the activity of the layer's inputs and the layers current activity.

Each Complex layer used a form of Trace Learning to learn sets of simple cells that tended to be temporally related (i.e. tend to co-activate near one another in time) (See Sec. 4.4 for a quick overview). The complex layer's learning rule only relies on a running temporal average of the input activations to the complex cell layer and a running temporal average of the current activation of the complex cell layer.

5.2.2.4.2 ICL: Module-wise Learning Rules Each module within an ICL model was trained one at a time in order according to its internal unsupervised learning rules, similar to stacked-auto encoding (Bengio et al., 2007), see Fig. 4.2.

5.2.2.4.3 Readout Layer When classification performance needed to be examined a final softmax classification layer was placed on top of the ICL model (or on top of the layer of interest within the ICL model). This classification layer was trained separately to categorize labeled object images, using supervised learning via the Adam optimizer method with dropout on its inputs to prevent over-fitting. While training the weights of this classification layer, the weights of the ICL model were held fixed. This final classification layer is called a *readout layer*, as its purpose is not to change the underlying representations present in the model via deep backprop, but to make the model's categorical representation more accessible

for analysis. Although the readout layers were not used in Simulation Study 1 they were used heavily in Simulation Studies 2 and 3.

5.2.2.5 Testing Procedure

To test if the ICL model groups neurons that represent the same category together on the cortical surface, I developed a novel procedure. Under this procedure, I tagged each high-level unit with the categories they are selective for. After the tagging process, I found each contiguous region of units that are selective for a category and recorded their area. These areas were aggregated to find a mean region size statistic for all of the map's contiguous category-selective regions.

As a second measurement, I calculated the mean region size statistic repeatedly for randomly permuted maps of the same ICL units. This gave me an estimate of the expected mean region size statistic's distribution for category-selective regions that would occur due to chance with the same set of units.

5.2.2.6 Defining Unit Selectivity

I used the ratio of the mean activation of a neuron for a category versus the mean activation of that neuron to all other categories to determine if a neuron was selective for a category. This is a common measure of selectivity in both fMRI studies and multi-unit recording studies. When weighted by the standard deviation of activity for non-categories of interest, a p-test for significance can also be done. In one of our followups, I also used this weighting in order to rank the relative selectivity of neurons for visualization purposes.

5.2.2.7 Defining Contiguous Regions

For my purposes two units were contiguous if they shared selectivity for a category and were part of a von Neumann neighborhood, which is simply up, down, left, and right of a unit on

a gridded cortical sheet. To find contiguous regions, I used a variant of the common flood-fill algorithm (Torbert, 2016) to group together linked units that select for the same category.

5.2.3 Data Analysis

The purpose of this statistical analysis was to estimate the probability that the average size for category selective regions could occur by chance when the units were actually arranged in a purely random fashion. I performed a non-parametric hypothesis test where the mean area statistic for the normal ICL model's contiguous regions was compared to the mean area statistic's distribution for random maps. Here the area distribution for random maps was calculated by resampling randomly permuted versions of the units in the normal ICL map and recalculating the mean area statistic. This allowed me to compare the average area of contiguous regions in the normal ICL map to the expected distribution of areas from random maps in a Monte-Carlo permutation test. I expected the formation of non-spurious contiguous regions to be all or none, thus I anticipated a very large effect size.

I also be visualized the activation characteristics of the high-level cortical maps using several methods. Firstly, I looked the ratio of the mean activation to each category to other categories in order to label each neuron with the category (indicated by color) that it was most traditionally selective for in a *selectivity map* (SEL map) (see Sec. 5.2.2.6 X & Fig. 5.3). Second, I looked at each neuron's *Maximally Exciting Stimulus* and their combined map structure in a *Maximally Exciting Stimulus map* (MES map) (see Fig. 5.4B). Third, I re-labeled the MES maps by the category of each neuron's MES using the same color coding used with the selectivity maps (see Fig. 5.4C). Importantly, these visuals will only be used for qualitative comparison in this study.

For an intuitive reference, a Selectivity Map (SEL map) can be thought of as visualizing the category of items that activate each neuron most strongly on average, whereas the Maximally Exciting Stimulus Map displays the single item that caused each neuron to reach

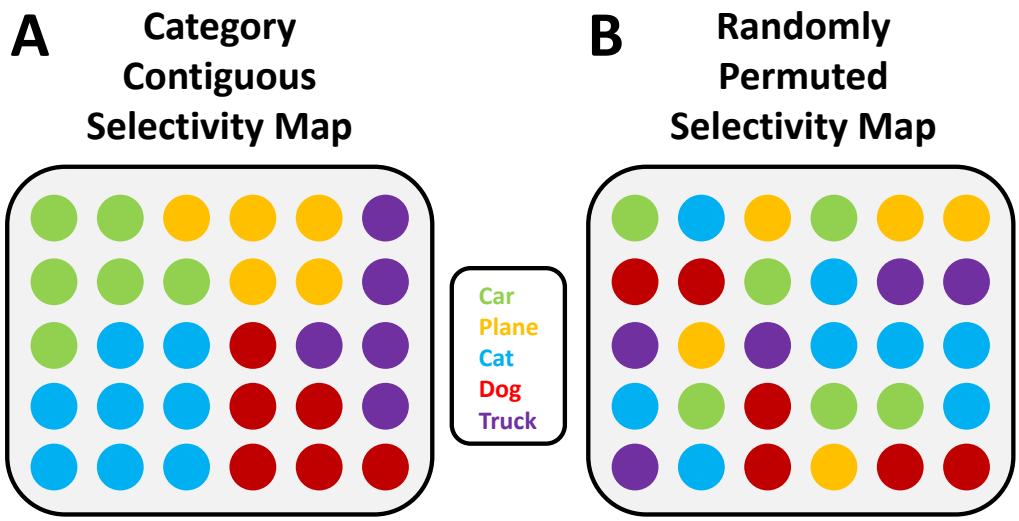

Figure 5.3. Illustration of contiguous and random maps.
These pictures are somewhat simplified, as each unit can actually be selective for multiple categories and thus a unit can be a part of multiple contiguous category selective regions. Also note that the randomly organized maps still develop contiguous regions, but these tend to be small when there are numerous categories and are purely generated by chance.

its peak activity. For better comparison to the Selectivity Maps, I often recolored the MES maps by the category of each neuron's maximally exciting stimulus.

5.2.4 Results

In the main part of Simulation Study 1 I wanted to examine if the mean category selective region size or MRS is larger than would be expected due to chance. To test this hypothesis I performed a non-parametric bootstrap test for the MRS statistic on an ICL-Map-Optimal model trained on the Fashion-MNIST image dataset. I found that the MRS statistic was significantly above chance, $p < .001$ (Fig. XB). Further, the size of the effect was very large indicating that a less powerful test could be used in the future. As a sanity check (given that this is a new method) I also tested what the MRS statistic would be for an untrained version of the ICL-Map-Optimal network. For the untrained network the MRS statistic was

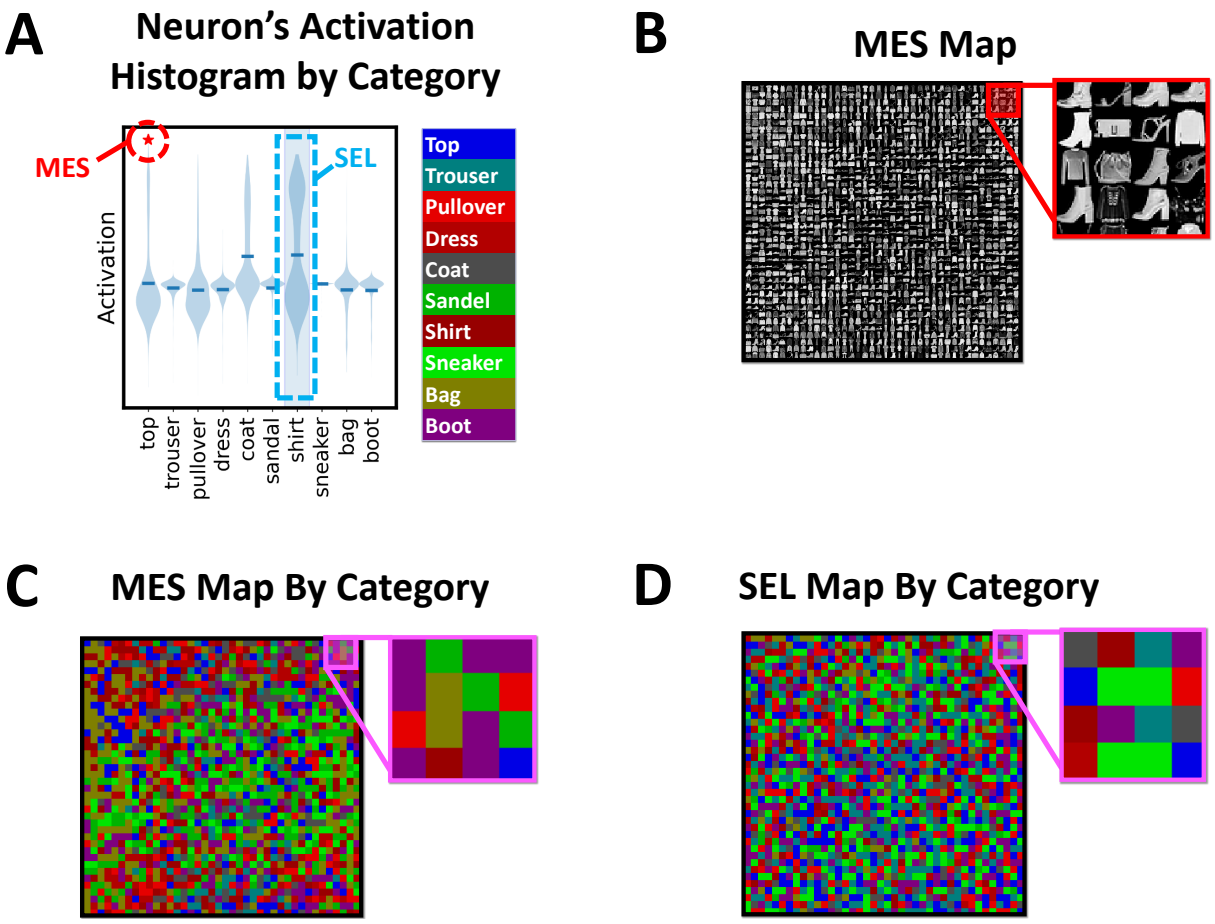

Figure 5.4. Examples of Map Visualization
(A) Show an example neuron's activation histogram by category. The (RED) circle star shows the activity and category of the *Maximally Exciting Stimulus* (MES) (i.e. the stimulus item that caused a neuron to reach its peak activation). The (BLUE) dashed box indicates the category for which the neuron is most selective as given by. Further, (A) also shows the color coding for each category in the map visualizations. (B) shows a picture of each neuron's MES located where the neuron is positioned on the cortical surface. (C) shows the category of each neuron's MES color coded by its category. (D) shows the category for which each neuron on the map is most selective. For reference, the MES and SEL maps by category are two different ways of characterizing the activation characteristics of a set of neurons, where the SEL map visualizes the categories that neurons activate most strongly to on average, and the MES map by category visualizes the category of the item which causes each neuron to reach its peak activation over the entire dataset.

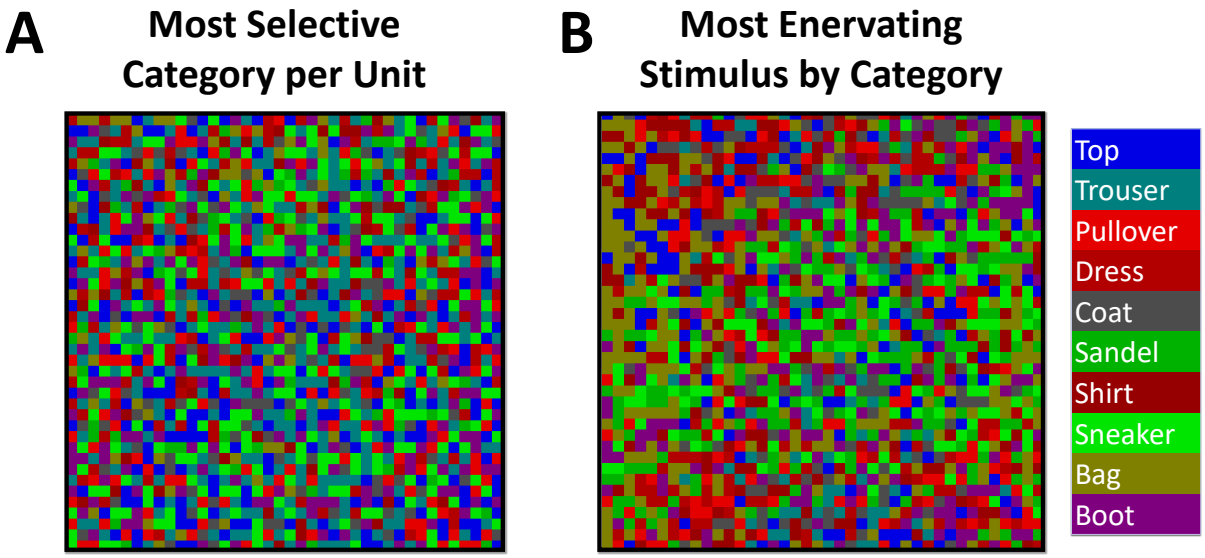

Figure 5.5. Visualizing Category Representation with Standard Methods
(A) shows a representation of how the neurons in a map respond to different categories of objects by color coding each unit according to the category it is most selective for (i.e. responds most to on average). Often this kind of map will be abbreviated as a SEL-map. (B) shows a representation of how neurons in a map respond to different categories by color coding according to the category of the stimulus item which caused each neuron to reach its peak activity. Often I will refer to this map as a MES-map. Note that highly similar kinds of object categories are usually given similar colors for visualization purposes.

not above chance, $p = N.S.$, and was in fact slightly below the chance distribution (Fig. 5.6).

I also visualized the activation characteristics of these maps relative to the 10 categories in the dataset. I noted several qualitative comparisons. First, the agreement between the MAS and SEL maps for both the untrained and trained ICL-Map-Optimal networks was generally quite low, at 19.6% and 19.0% respectively. Second, the **untrained** networks displayed speckle or noise like patterns of selectivity with little to no contiguity of category selective regions, while the **trained** network displayed large regions of contiguous category selective neurons, some with around 30 neurons (see Fig. 5.7).

While the large effect size for the MRS statistic and the qualitative map characteristics in the top layer of the ICL-Map-Optimal network could indicate the development of hierarchical

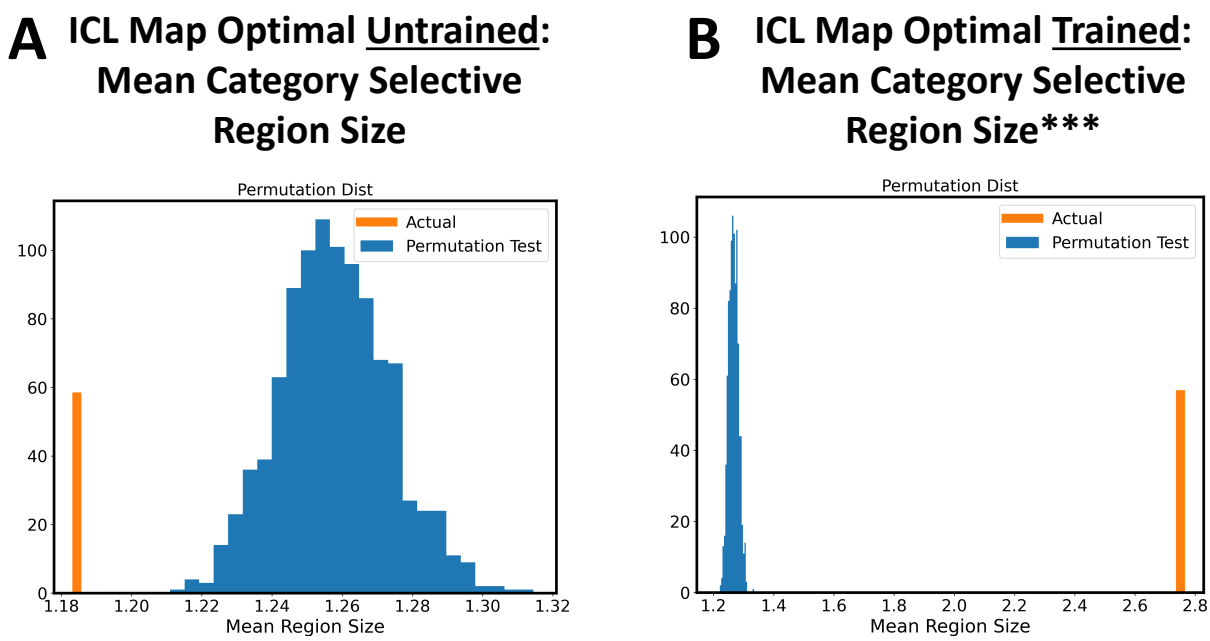

Figure 5.6. MRS Boostratp Test: Untrained vs. Trained ICL
(A) the distribution of the randomly permutation map MRS statistic (BLUE) for the untrained ICL model and its actual value (ORANGE). Note that the MRS statistic is below the permutation distribution. (B) shows the distribution of the permuted MRS statistic (BLUE) for the trained ICL model and its actual value (ORANGE). Note that the MRS statistic is significantly above the permutation distribution.

categorical features, I wanted to examine if this apparent map structure could be caused by propagation of low-level feature representations and biases in the dataset through multiple layers. To test this I performed the same non-parametric bootstrap test on another version of the ICL model where the lateral-interactions and size of each layer in the model were kept constant called ICL-Lat-Optimal. If the model is learning sophisticated categorical features I would expect that the MRS statistic would increase from layer 1 to layer 4 of the model. In theory, the MRS statistic would grow because each layer would introduce incrementally more specialized features selective for objects that would tend to co-activate more than local simple features, causing the cortical map forces to aggregate them together in larger groups. I found that the MRS statistic tended to go down from layer S1 to layer S4 of this model,

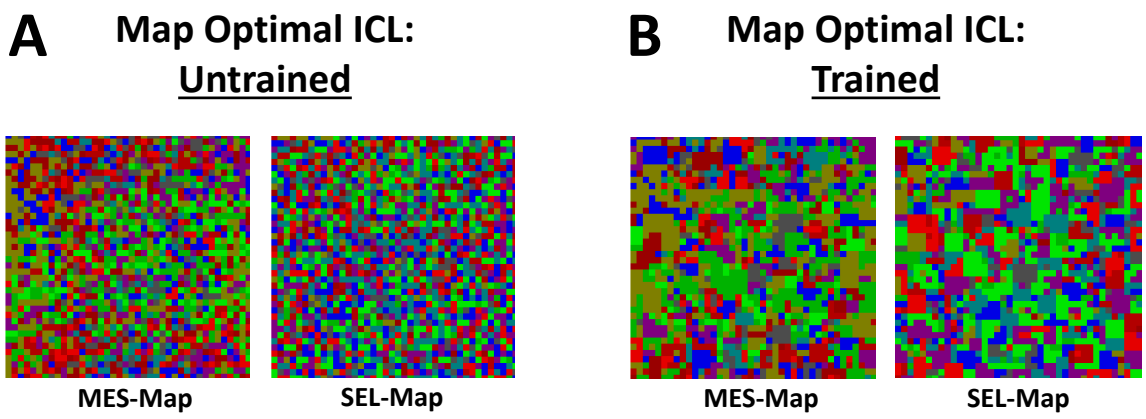

Figure 5.7. MAS and SEL-maps: Untrained vs. Trained Map-Optimal-ICL
(A) shows the side by side MAS and SEL-maps for the untrained Map-Optimal-ICL model. Note that both maps show speckle patterns and very little regular structure. (B) shows the side by side MES and SEL-maps for the trained Map-Optimal-ICL model. Note that contiguous regions of category selectivity and MES organization have formed due to learning.

with the MRS = 2.1 and MRS = 1.96, for layers S1 and S4 respectively (see Fig. 5.8), suggesting that more sophisticated categorical features were not developing. Because of the nature of this comparison, a proper bootstrap test is less feasible.

Visualizing the selectivity and MES map for layers S1 and S4 of the ICL-Lat-Optimal network revealed one main qualitative characteristic. The overall distribution of category selective units is different between layers S1 and S4, with the selectivity and MES maps of layer S1 appearing to display edge effects, while layer S4 grouped categories by selectivity and MES in more or less evenly distributed clumps across the map (see Fig. 5.9). This likely indicates that dataset biases drive the locations of category selective units in layer S1, while this is less the case for layer S4.

Further following up on these results, I wanted to see if the addition of complex cell layers would change the region size characteristics. Using a version of the ICL-Lat-Optimal network with complex cell layers interleaved between the simple cell layers I again computed the MRS statistics for the layer C1 and layer C4. With the addition of complex cells the

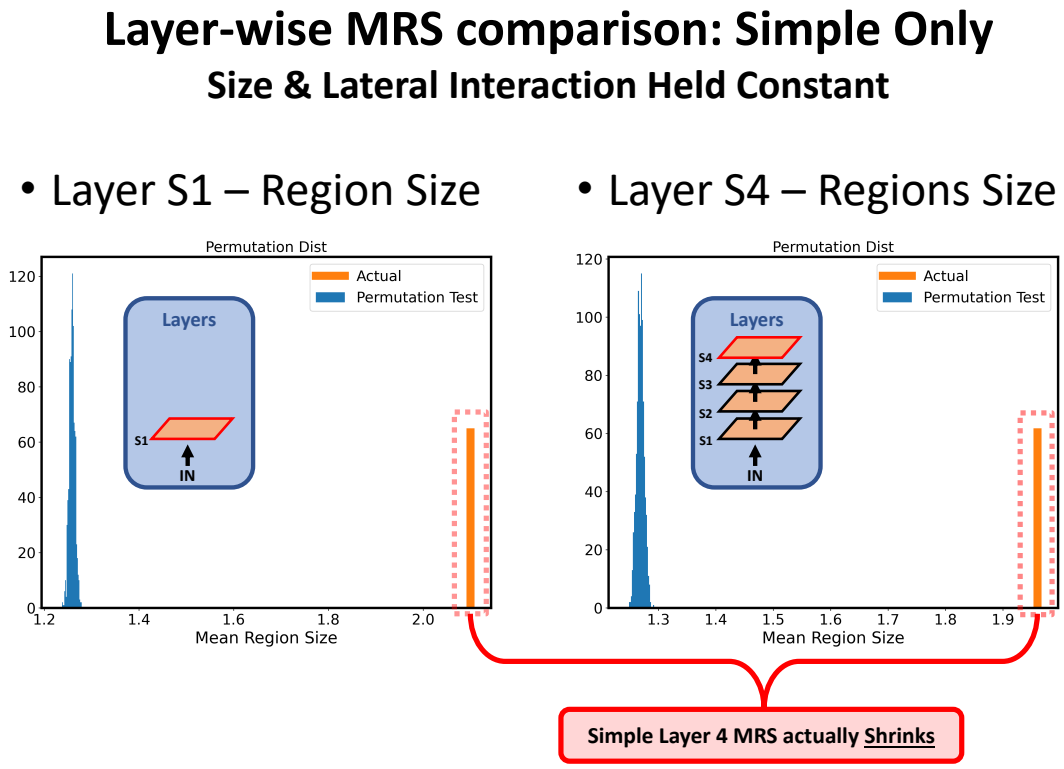

Figure 5.8. Comparison of MRS: Layer S1 vs Layer S4: Simple-Only
This figure shows a comparison of the MRS statistic and permutation distributions between the first simple cell layer and the 4th simple cell layer in a version of the model where lateral interaction are uniform across layers. Note that actual MRS statistic (YELLOW) tends to shrink from layer S1 to layer S4. This suggests that the nature of category selectivity from layer S1 to S4 may not have changed much.

areas appeared to grow from layer C1 to C4, with MRS = 2.1 and MRS 2.4 respectively (Fig. 5.10). This suggests that the complex cells could be causing the network to develop more sophisticated category selective features across its layers. Further, layer C1 and layer C4 of the simple only and simple + complex networks display markedly different map structures, with the MES-map for the simple + complex network displaying large regular patches of units that activate maximally to the same category (Fig. 5.11). In addition, the MES-map for the top layers of the simple cell only network vs. the complex+simple network look vastly

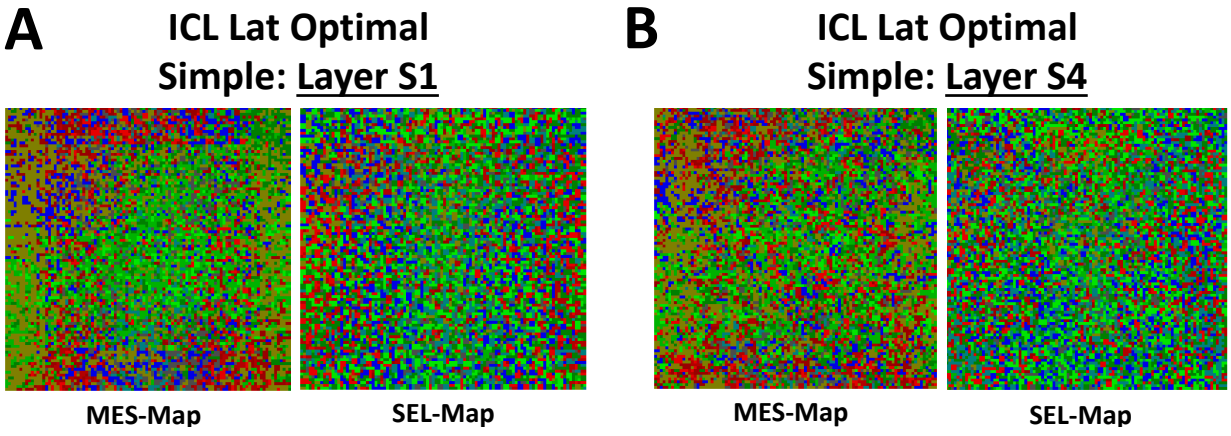

Figure 5.9. MES and SEL Maps: Layer S1 Vs. layer S4 : Simple Only
(A) shows the MES and SEL maps for layer S1 of the LAT-Optimal-Simple Only ICL model (B) shows the MES and SEL maps for layer S4 of the LAT-Optimal-Simple Only ICL model. Note that both the MES and SEL maps change from layer S1 to S4. S1 appears to have more regional selectivity, but this may be due to edge effects in the dataset. Additionally, layer S4 may simply be reducing the appearance of these edge effects via its effectively larger receptive fields.

different, with the simple network displaying small MES patches, and the complex network displaying very MES patches (Fig. 5.12).

5.2.5 Discussion

Simulation Study 1 shows that more biologically plausible learning rules can in fact lead to the development of cortical map structures that are seemingly contiguous for category selectivity. However, it is highly possible that the development of category continuous map structures may be an artifact of the contiguous feature learning constraints of the ICL model and of the low-level stimulus regularities present in the test dataset.

I did several follow-up examinations to get a better sense if this the maps were due to lower level interactions. When I examined map development across multiple layers of representation, it appeared that the size of the categorically selective regions did not grow substantially from layer S1 to layer S4, in the models without complex cells, when controlling for lateral interaction and map size (see Fig. 5.8). This lends credence to the idea that

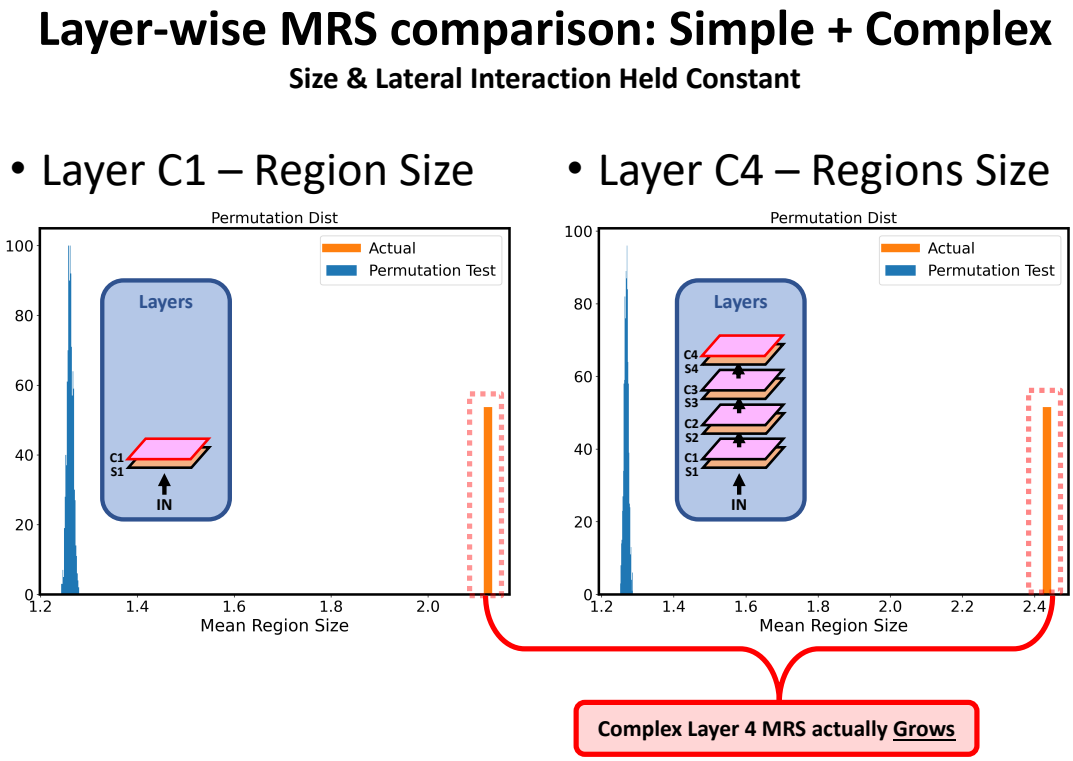

Figure 5.10. Comparison of MRS: Layer S1 vs Layer S4: Simple+Complex
This figure shows a comparison of the MRS statistic between layer C1 and layer C4 of the LAT-Optimal-Simple+Complex version of the ICL model. Note that the MRS statistic tends to grow from layer C1 to layer C4. This suggests that contiguous representation of categories may enhance as more layers of this form of ICL are added.

appearance of categorically organized maps was due to low-level artifacts and biases in the dataset. When complex cells were added however the size of categorically selective regions typically grew (see Fig. 5.10). Further, the addition of complex cells greatly changed the character of the maps (see. Fig. 5.12). However it remains to be seen if the complex cells will change the more computational characteristics of the of the network with respect to categories.

Looking at the top-selective neurons I found "classically" selective neurons (see Fig. 5.13), but when I varied the input locations of stimuli the appearance of "classically" se-

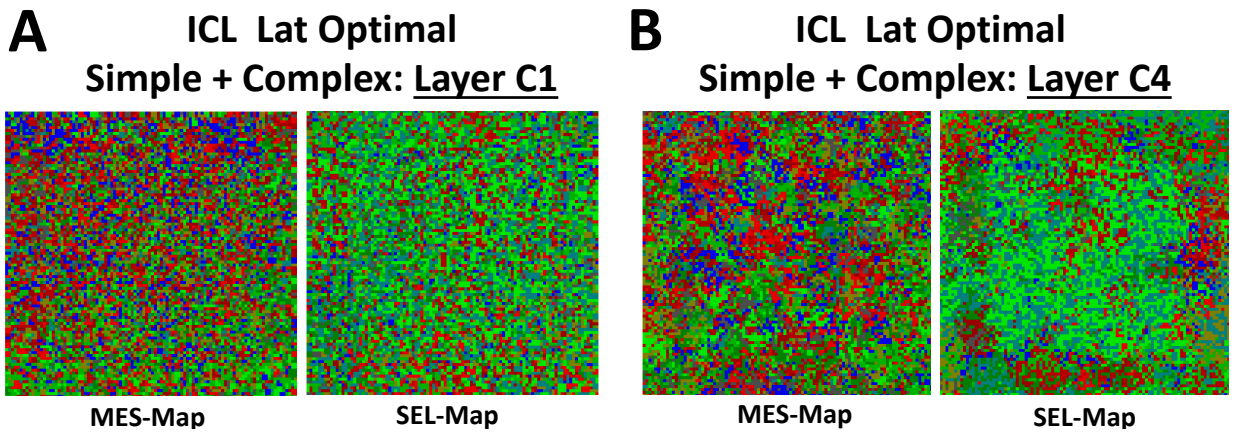

Figure 5.11. MES and SEL Maps: Layer C1 Vs. layer C4 : Simple + Complex
This figure shows the MES and SEL maps for layer C1 (A) and layer C4 (B) of the LAT-Optimal-Simple+Complex ICL model. Note that the MES map for layer C4 takes on a patchy structure where neurons activating most strongly to stimuli within a particular category group quite strongly. The SEL-map for layer C4 appears to show some form of edge effect.

lective neurons reduced as I increased the positional variability. At half stimulus width variability, selectivity was greatly reduced and at full image width selective neurons all but disappeared. This suggests that even if these neurons are encoding high-level features, they are not particularly invariant which again may suggest that these high-level feature maps are not organized by category so much as low-level stimulus features. The results of Simulation Study 2 which get to the heart of ideas about selectivity vs. functional specialization, and may further cast doubt on the idea that these maps are strongly category based.

Importantly, Simulation Study 1 shows that more neurally inspired architectures like the ICL model can be used as a basis for starting to investigate and build better models of hierarchical maps along the ventral visual stream. ICL is likely a first-of-its-kind model in its ability to develop deep topographic cortical maps, which should hopefully encourage more development in this area.

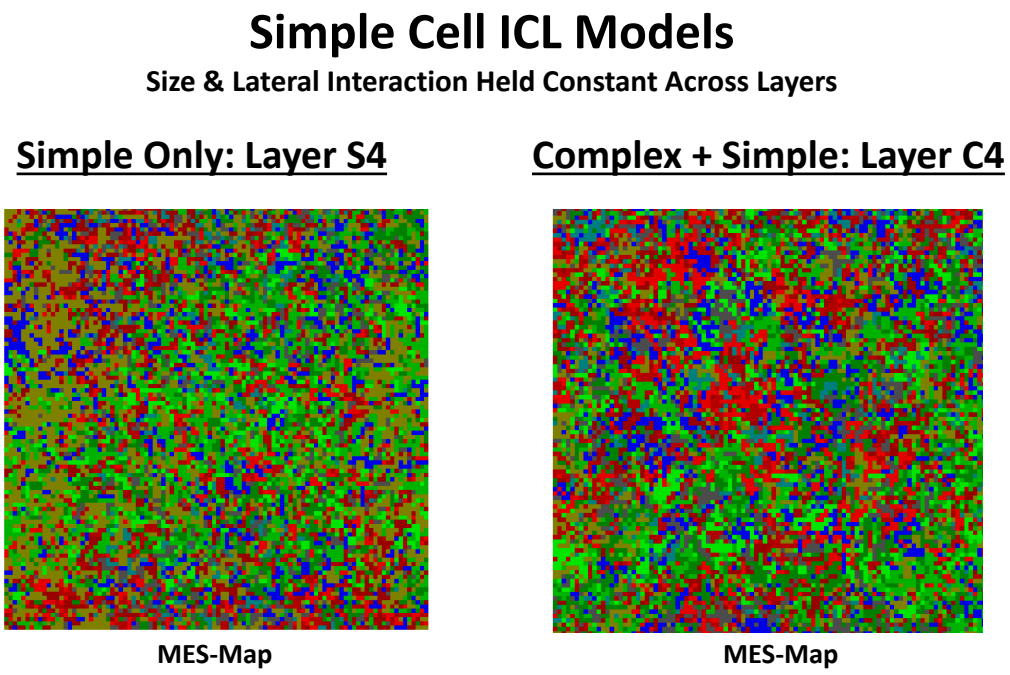

Figure 5.12. Comparison of MES-maps: Layer S4 Vs. layer C4
This figure shows a direct MES-map comparison between layer S4 of the simple only model (LEFT) and layer C4 of the simple+complex model (RIGHT). There appears to be larger more regular grouping of neurons which have their MES in the same category. This might suggest that the addition of complex cells in the ICL model changes the high-level representation of categories to be more continuous on the cortical surface.

5.3 Simulation Study 2: Unit Specialization

5.3.1 Introduction

Researchers have long been interested in whether specific neurons in the brain serve specialized functional roles in the high-level ventral cortex. For example, is a neuron that reacts strongly to "houses" utilized by the nervous system to represent that a "house" has been identified or are the neuron's weaker responses to other categories equally important from an information processing perspective? If so, this would call into question the entire concept of unit specialization in the field of neuroscience.

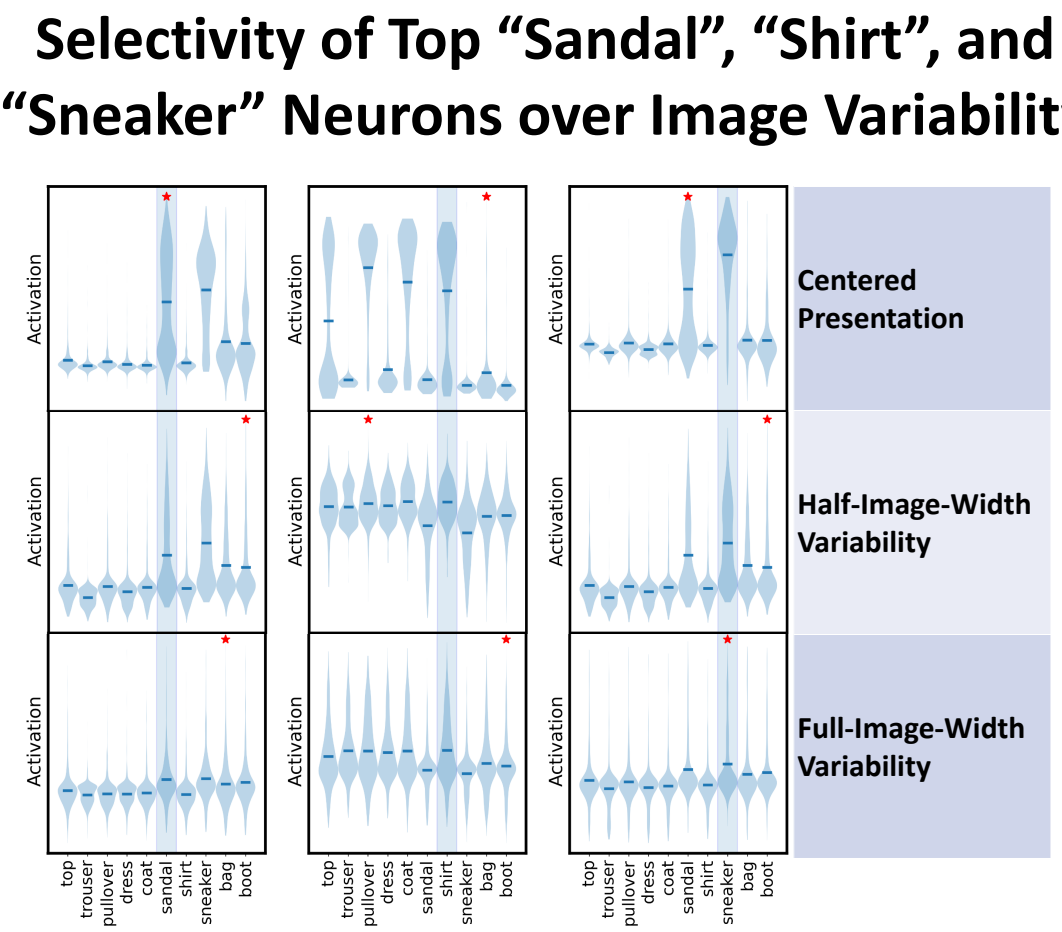

Figure 5.13. Apparent Selectivity as a Function of Stimulus Variability
This figure shows how the the apparent selectivity of neurons changes as a function of image position variability in the testing dataset. When images are centered (TOP ROW), the network appears to have highly selective neurons. When the images are uniformly distributed across different shifts in position that span half the image width (MIDDLE ROW), the apparent selectivity of neurons appears to drop significantly. At full image width shift variability (BOTTOM ROW), the strong selectivity seen earlier appears to largely disappear and also appears to be driven more by outlier stimuli. The disappearance of strong selectivity, as image presentation variability is increased, suggests that these neurons may not be learning strongly categorical representations and that at least part of their apparent selectivity is due to lower-level image regularities.

Neuroscientists often try to understand a neuron's functional specialization by measuring its so-called *selectivity*. *Selectivity* is usually just a measure of whether a neuron tends to activate more strongly for some classes of stimuli or situations than others. While selective neurons may be specialized, selectivity is a highly imperfect measure of specialization and does not imply specialization without further information.

Going as far back as 1981, connectionists were showing that simple neural networks could represent, recall, and distinguish between stimuli without the need for functionally specialized units (Anderson and Mozer, 1981). Further, they demonstrated that conventional neuroscience measures of selectivity would almost always show that these networks have highly selective neurons, even though the functional role of those neurons was provably unspecialized. Recently, this line of reasoning has been revived with modern experiments using much more sophisticated neural network models like DCNNs (Parde et al., 2021; Szegedy et al., 2014a). Essentially this new work has demonstrated that functionally specialized units in the upper layers of these networks are not required to generate state-of-the-art performance for real-world image recognition benchmarks, and that similarly deceptive highly selective neurons are still found in the advanced networks.

If computational experiments were the only line of evidence for specialization in high-level cortical areas, then the hypothesis that high-level cortical units are relatively unspecialized might be more common in neuroscience. However, many lines of research suggest this is an overly simplistic view of how neurons represent stimuli in the brain. Generally, research has shown that the brain physically groups neurons that are selective for related stimuli together. For example, these groupings include but are not limited to retinotopy, ocular dominance columns, orientation columns, FFA, FBA, VFWA, and PPA (Bednar and Wilson, 2016; Foldiak, 2003; Grill-Spector and Malach, 2004; Grill-Spector and Weiner, 2014). Further, when these physical groupings of related neurons are injured, humans tend to exhibit specific behavioral deficits related to the selectivity of the damaged neurons. For example, damag-

ing the peripheral portion of V1 impairs peripheral visual perception, and damaging FFA specifically impairs face perception (Farah, 2004; Schiltz et al., 2005).

This evidence strongly suggests that, at least at a large-scale, areas of the ventral path serve functionally specialized roles. However, at a smaller scale, the evidence that individual units are serving specialized roles within functional domains like FFA, VWFA, or object selective fusiform cortex, is more lacking. For instance, it is still reasonable to ask if individual neurons within the *Visual Word Form Area* (VWFA) are specialized to represent specific semantically meaningful features of words and letters or if every neuron is participating holistically in the representation of word forms despite their apparent selectivity. Further, even at a large scale the actual degree of functional specialization is still very open for debate.

The smaller scale question of specialization within specific functional domains is a fertile ground for simulation work using today's state-of-the-art DCNN models. Szegedy et al. (2014a) showed that DCNNs trained for numeral recognition and general object recognition had interpretable unit selectivities, but that these selectivities were just as interpretable as those of random linear combinations of the same units. Further, (Parde et al., 2021) demonstrated that deleting high-level units in networks trained for face recognition caused performance degradation more in-line with units that were unspecialized for particular identities while still making identities highly separable (it should be noted that their results for other stimulus axes such as gender and head rotation were less clear cut, but the expectation for these categories is also unclear). In other words, while each individual unit tended to encode all the identities, their activation was so predictive that only a small number were needed in order to effectively separate individual identities. I will call these performance curves over unit deletion *unit deletion curves* (UD).

However, these results should be cautiously interpreted, as DCNNs and other state-of-the-art models today generally lack any of the biological mechanisms that would be expected to generate unit specialization in the first place. This leads to the question of Simulation

Study 2: does a model that features many of these omitted biological mechanisms learn functionally specialized units, contrary to DCNN models, or will it reproduce the work with DCNNs?

Simulation Study 2 follows a relatively simple logic. First, the main operating assumption of this field is that the high-level ventral cortex's goal is in part to make relevant categories separable so they can be readily available to other neural systems (DiCarlo and Cox, 2007). Second, applying a random rotation to a unit representation fully preserves separability between categories, while removing any functional specialization for the units within a representation. Third, unit deletion affects representations with specialized units and unspecialized units in very different ways.

Given these stipulations, the unit-deletion curve of the normal model can be compared to the distribution of unit-deletion curves for randomly rotated versions of the same model. These randomly rotated representations are guaranteed to be unspecialized while preserving the original model's ability to separate categories. This comparison constitutes a simple hypothesis test where the distribution of unit-deletion curves for randomly rotated versions of the model acts as the null distribution for the level of specialization expected.

5.3.1.1 Definitions

Before discussing Simulation Study 2 in detail, I will clarify the terms selectivity and specialization. *Selectivity* can have several meanings in neuroscience, but the general assumption is that a neuron is more selective for a type of stimulus if it responds more strongly to that stimulus type relative to other stimuli. Given this general assumption, there is an abundance of measures of selectivity, which mostly give similar results. *Specialization*, on the other hand, connotes that a neuron serves a specific functional role within a representation. As discussed earlier, selectivity is often assumed to capture specialization, but this need not be the case from a computational perspective.

To better understand how selectivity and specialization are related I will introduce the concept of activation-space. When a stimulus is presented to a set of neurons, the neurons will activate to different degrees. Each neuron's level of activation can be regarded as an axis of geometric space or activation space. Each stimulus would be represented as a point in activation space, or equivalently as a vector whose elements are the unit activation levels. In activation-space, selectivity only implies that certain groups of stimuli tend to cluster about a particular axis, but it does not imply how this occurs. Fig. 5.14 shows that neurons can be selective in very different ways that may or may not imply specialization. Thus, it is important to make a careful conceptual distinction between unit selectivity and unit specialization.

Selectivity is a measurable quantity, while specialization is more concerned with how the brain uses a neuron's response properties. It is quite difficult to say with certainty what role any specific neuron plays in information processing in the brain. Generally, neuroscientists instead try to characterize how a neuron could be used by the brain to process information. To answer this weaker question, researchers often use the concept of separability. For our purposes, separability measures how easily a single neuron's activity or a set of neuron's activation space can separate different groups in a geometric sense. Fig. 5.15 shows how selectivity does not necessarily imply specialization as measured by separability.

In practice, separability requires more sophisticated measures than selectivity, often requiring the training or fitting of a model which I will call a *decision function*. Essentially this model draws boundaries on a high-level activation space with some limitations depending on the type of decision function used. For example, a linear decision function can only separate groups using high-dimensional planes, and a nearest-neighbor decision function can only separate an activation space into a set of convex tiles called a Voronoi system. Different kinds of decision functions can be used to characterize how difficult it is to separate categories within an activation space. Importantly, this measure confounds two sources of

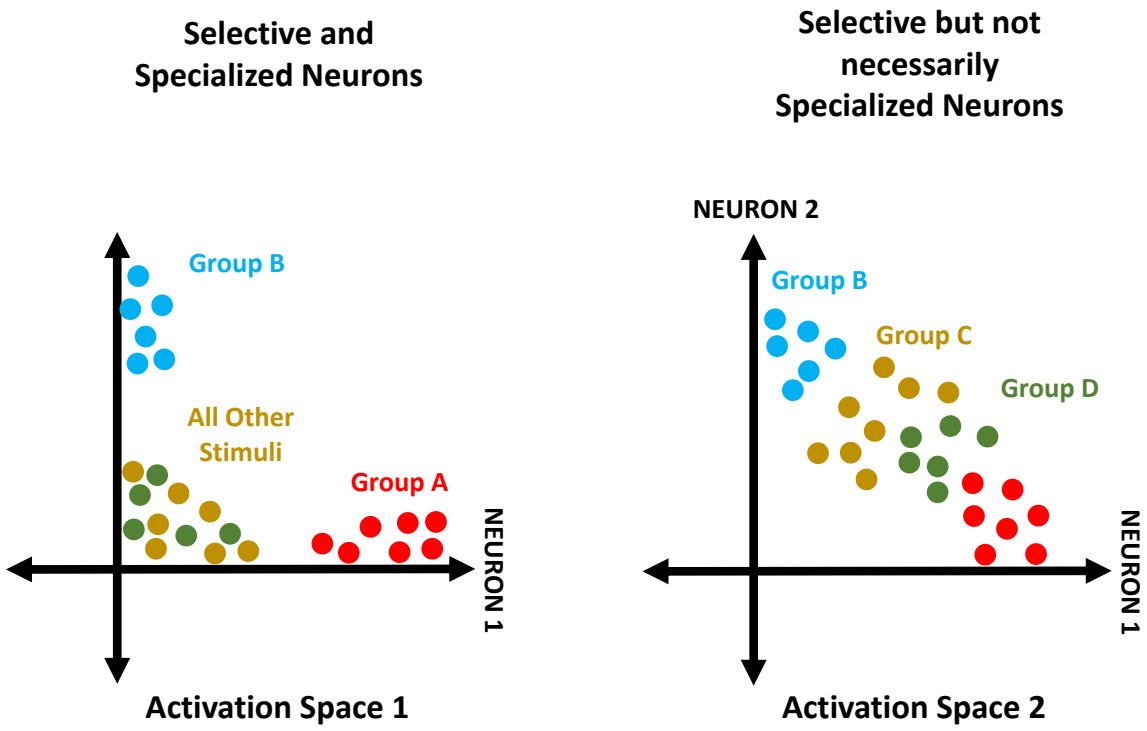

Figure 5.14. Activation Space: Specialized Vs. Unspecialized Neurons
This figure shows how neurons can be selective for the same groups of stimuli while having quite different response properties in activation space. In the first activation space, group A can only be identified using the activity of neuron 1 and group B can only be identified using the activation of neurons 2, which implies that these neurons serve specialized functions. In the second activation space, the responses of both neuron 1 and 2 can be used four different stimulus categories, meaning these neurons do not have to serve a specialized function.

separation difficulty. Some groups may be difficult because they require a more complex decision boundary to be separated, and some groups may be difficult to separate because their distributions overlap substantially in activation space.

For this paper, I will use a soft-max readout layer as my decision function to estimate how well the neural activation spaces in the ICL model separate groups of stimuli. Importantly this kind of readout layer is mostly insensitive to the magnitude of input patterns and can be thought of as a form of nearest neighbor function for direction rather than location. I chose this decision function because its similarity measure is closely related to many commonly

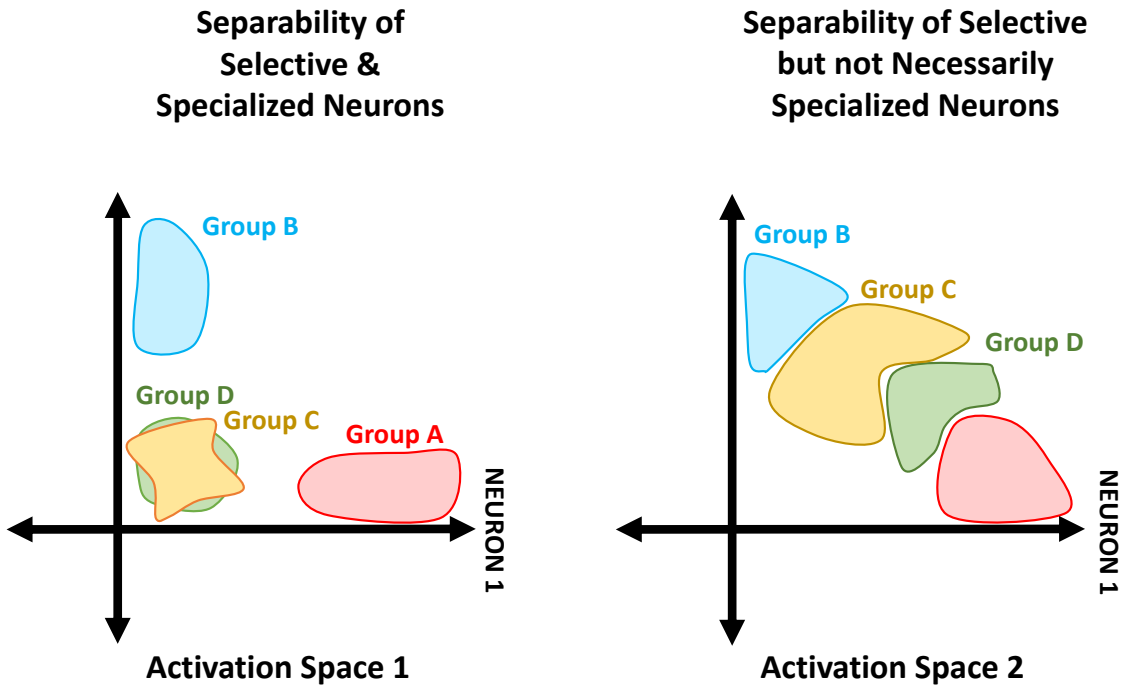

Figure 5.15. Activation Space: Separability
This figure shows how selectivity does not necessarily imply specialization, while separability can be regarded as a more reliable indicator of the level of specialization neurons might have. In the first activation space, the activation of neuron 1 can separate group A from the other groups and the activation of neuron 2 can completely separate group B from the other groups. However, even in the combined activation space groups C and D cannot be separated from one another, suggesting that neuron 1 and neuron 2 are specialized for group A and group B respectively because those are the only groups these neurons can separate reliably. In the second activation space, groups A B C & D can be completely separated, although the partitions between groups may be complex and each individual neuron only partially separates the groups, implying that these neurons do not have to be used in a specialized way.

used decision functions for interpreting neural activation patters. In addition, this particular parametric form of decision function supports a more direct comparison with current DCNN classification research (Parde et al., 2021; Yamins and DiCarlo, 2016b). It is important to emphasize that typical DCNNs simultaneously adjust the parameters of the soft-max readout layer and the parameters of the DCNN to optimize read-out performance. In the model proposed here, the parameters are estimated independently of the parameters of the soft-max readout layer in all use cases

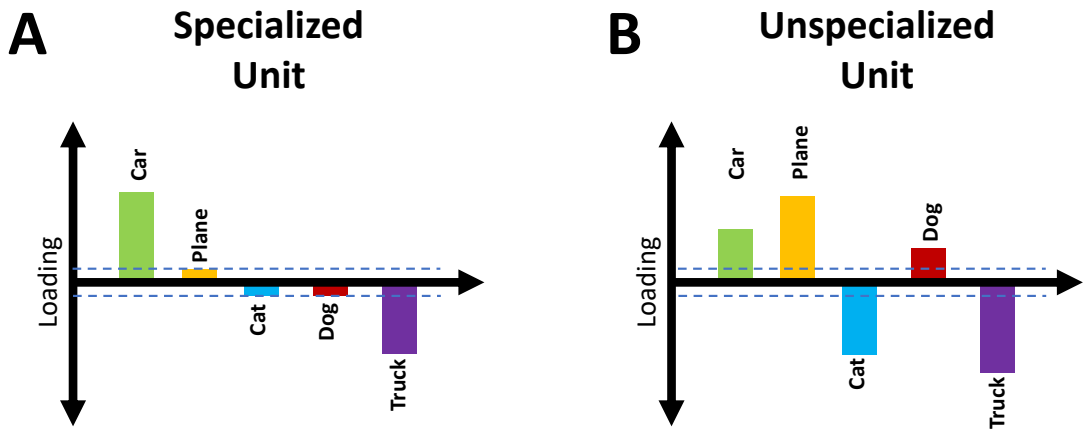

Figure 5.16. Unit Loadings: Specialized Vs. Unspecialized
This figure shows an illustration of what the selectivity of specialized units and unspecialized units might look like in terms of their selectivity. Here I use the term *loading* to refer to how much a unit activates for a category on average minus its normal activity. The dotted line represents a chance or spurious level of loading, that is effectively zero. The specialized unit has a strong positive loading for Car and a strong negative loading for Truck, this neuron might be conventionally labeled a "Car not Truck" neuron and it is only useful for separating cars and trucks. The Unspecialized unit has positive and negative loadings for all of the categories, some weak and some strong. This neuron might be conventionally labeled a "Plane not Truck" neuron, but it can participate in discrimination of all categories.

Later in this paper, I will often use the term specialization to refer to how specialized a set of neurons must be given their separability characteristics as measured by the accuracy

that a trained a soft-max readout layer can achieve when using that neural representation as input.

The crux of the emerging debate about selectivity and specialization is the issue that standard measures of selectivity do not necessarily imply that a neuron serves a specialized role in the brain Parde et al. (2021). Fig. 5.16 shows how two neurons can both be selective and how both can be semantically interpreted, while only one is constrained to serve a specialized functional role. DCNNs appear to learn the more unspecialized representations, and given their performance it suggests that unit specialization is not a requirement for strong object recognition (Parde et al., 2021). However, it remains to be seen if other high performing models with additional biological constraints learn more specialized units.

5.3.1.2 Objective

The goal of Stimulation Study 2 was to examine whether the ICL model learns specialized unit representations as current theories of cortex suggest or if it learns more unspecialized unit representations like DCNNs. To achieve this goal, I performed an analysis similar to the (Parde et al., 2021) method for characterizing the level of specialization individual units have for a set of categories. This method simply examined the discriminative power of random sets of units for a set of categories. When a representation features an explosive growth in discriminative power as the number of units increases, it indicates that all the units partially encode all the categories. The extreme of this behavior indicates a lack of unit specialization. Knowing what the extreme of this behavior would look like for a model, it can then be compared to the model's actual behavior to determine if the model is learning more specialized unit representations.

Intuitively, unit specialization exists on a continuum. On the specialized end of the continuum, each unit only participates in the representation of one category. In this case, the number of categories that can be represented increases linearly with the number of

units (Fig. 5.17). On the unspecialized end of the continuum, each unit participates in the representation of all categories. In this case, the number of categories that can be effectively represented grows combinatorially with the number of units (see Fig. 5.17).

Using this information, where a model falls on the continuum can be found by examining how well increasingly large subsets of a model's units perform at a classification task. Models with highly specialized units will reach their peak performance at a slower rate (almost linear under certain circumstances). On the other hand, models with less specialized units will reach their peak performance very quickly.

I used the Parde et al. (2021) *unit deletion curve* or UD curve, which simply measures the categorization capacity of different numbers of random units in a model to characterize how specialized the units within the representation were for a set of categories. For fully unspecialized units, the UD curve should show a strong elbow shape, as its exponential categorization capacity would cause its performance to rise quickly and saturate after only using a small number of units (see Fig. 5.18A). For specialized units, the UD curve should show a much less pronounced elbow, as its capacity per unit grows much slower (see Fig. 5.18A). In this experiment, I modified the Parde et al. (2021) procedure to compare the UD curve of the ICL model to the UD curve that it was be expected to have if its units were completely unspecialized.

While it would be possible to directly compare the normal ICL model UD curve with the curve it would have if its units were unspecialized, I instead compared the area under the curves (AUC) for both UD curves. The area under the curve for the UD curve will be denoted with the acronym UD-AUC. I chose the UD-AUC statistic for comparing the two PPU curves because it adequately captures the speed with which a model reaches its performance saturation point as the number of units increases. A model with specialized units should show a smaller UD-AUC than an identically performing model with unspecialized units (see Fig. 5.18).

5.3.1.3 Predictions

Decades of research have led to the hypothesis that high-level ventral stream neurons are highly specialized (Foldiak, 2003; Grill-Spector and Malach, 2004; Grill-Spector and Weiner, 2014). Yet experimental observations have often been based upon unit selectivity measures rather than stricter measure of unit specialization. Following Parde et al. (2021), this observation raises the concern that experimental evidence for unit selectivity may have sometimes been misinterpreted as evidence for unit specialization. The UD-AUC statistic provides a helpful tool for more clearly identifying situations where unit specialization is present. In particular, if unit specialization is present in actual high-level ventral cortex, I would expect it to show a softer elbowed UD curve. On the other hand, DCNNs show high levels of performance can be achieved with completely unspecialized units, which produce severely elbowed UD curves (Parde et al., 2021). If the ICL model produces UD curves that indicate specialization, it would suggest that biological constraints may cause the high-level ventral cortex to learn more specialized unit representations than experiments with DCNNs show is strictly necessary.

The ICL model has several mechanisms that should cause it to narrow the functional specialization of all of its units. As such I expected that the ICL model's normal-UD-AUC would be much smaller than the rotated-UD-AUC on average (see, Fig. 5.20). This result would indicate the ICL model learns a specialized unit representation in contrast to DCNNs. If the ICL model shows specialized representations, then it would be worthwhile to do a minimal exploratory analysis to see what kind of features the ICL model is learning at its top levels.

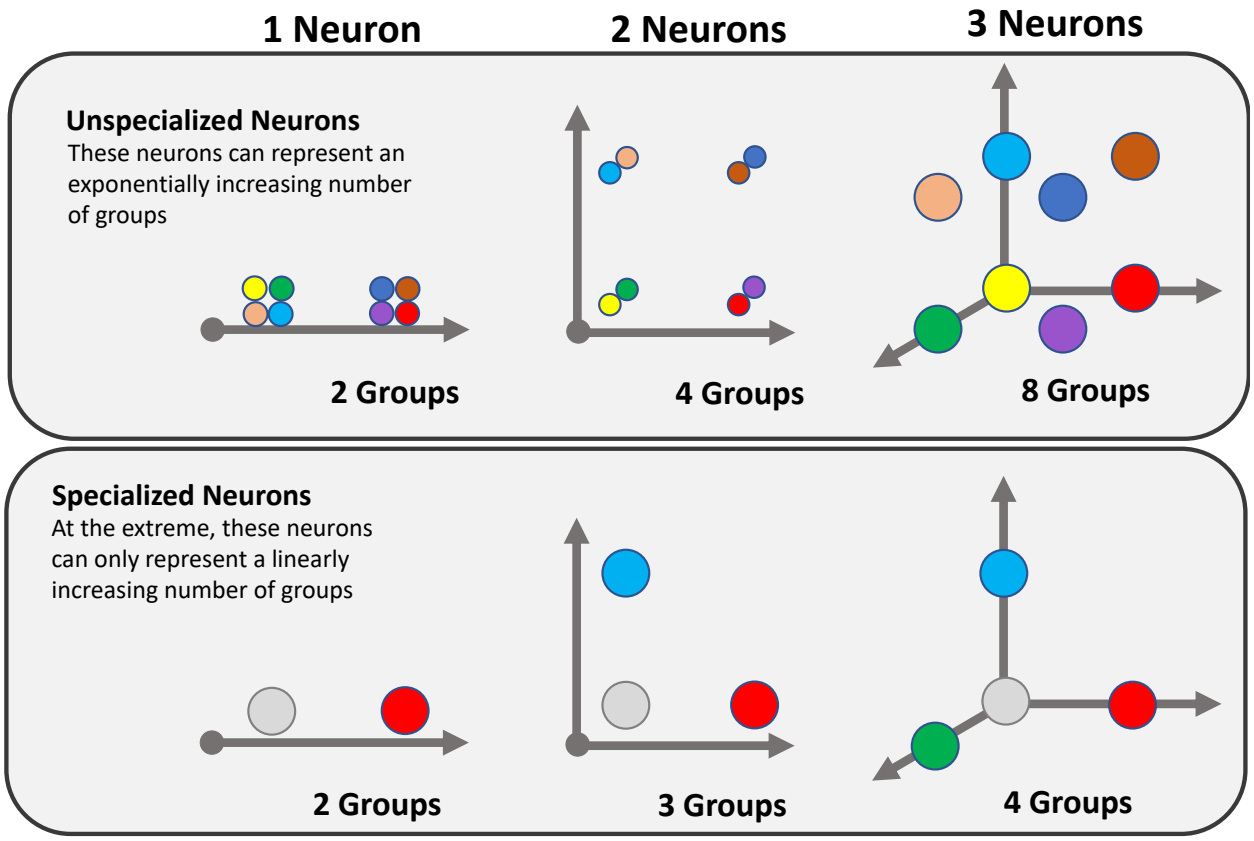

Figure 5.17. Visualization of Unit Discrimination Power: Specialized Vs. Unspecialized This figure shows how unspecialized neurons can increase the number of categories they can represent exponentially as the number of neurons grows and how the number of categories that specialized neurons can represent grows much slower. For simplicity, each neuron in this diagram is assumed to only have high and low activation values.

5.3.2 Simulation Methods

5.3.2.1 Stimuli & Training Procedures

Generally, the stimuli and the training procedures were the same as those in Simulation Study 1. Again, I used the MNIST-F dataset because it contained a large set of real-world images with large amounts of inter-item variation. Note that a readout layer was added to the ICL models used in this Simulation Study and trained separately from the ICL model to discriminate categories based on labeled images. Refer to Sec. 5.2.2.4 for a full overview

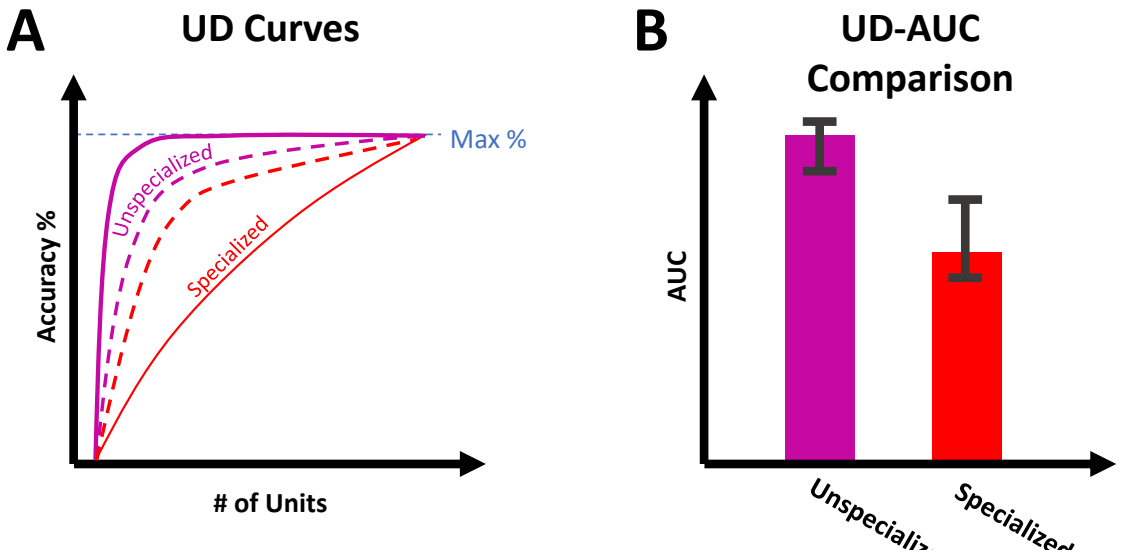

Figure 5.18. Expected UD and PPU-AUC Graphs
This figure describes the expected PPU curves and AUCs for representations with unspecialized and specialized units. (A) shows how the distribution of unspecialized unit curves (Purple) has a strong elbow and follows the top left corner of the UD graph, and how the specialized unit UD curve will be somewhere below. Similarly (B) shows how the area under the curve for unspecialized units will be much higher than the area under the curve for specialized units.

of the training procedure for the ICL model, and Sec. 5.2.2.4.3 for an explanation of the readout layer and it's separate training.

5.3.2.2 UD Curve & AUC in Detail

To measure a UD curve for a model, I randomly sampled the top-level units in groups of different sizes. Following Parde et al. (2021), I used increasing powers of 2 for group sizes (e.g. 16, 32, 64, 128, 256). To measure the discrimination performance of each random group, I trained a simple softmax classifier using each subset of units as inputs and measure its performance on the validation set for each dataset. A softmax classifier here means a learnable linear transformation followed by a softmax transformation. Because the performance for each group size is a mean of performances for different random sets of units, I will

also report the standard deviation for the performance of each group size. I then measured the UD-AUC using the mean performance values for each group size. This can be easily accomplished using the trapezoidal rule for approximate integrals.

While it may seem computationally intensive for a large number of UD curves to be estimated, this is not the case. The full ICL model will only be trained once for each dataset, and its outputs for every stimulus will be recorded. A special random rotation transformation, which is relatively inexpensive, will be applied to the activation records, to estimate an unspecialized comparison curve (this method will be detailed more in the following section). Only the softmax classification layer needed need to be retrained or run to calculate the performance for each group size.

5.3.2.3 Measurements

In my adaptation of the Parde et al. (2021) UD method, I measured two versions of the high-level representation in one ICL model. In the first measurement, I estimated the ICL model's normal UD curve. In the second measurement, I estimated the distribution of UD curves that the ICL model would be expected to have if its units were fully unspecialized.

To produce the second measurement, I transformed the representation of the ICL model to be unspecialized while maintaining its original performance. This can be accomplished by randomly rotating the axes of the high-level unit space of the ICL model. Rotating a unit-space will distribute an individual unit's loadings (i.e. linear discriminative power for each category) to all of the new units in the rotated space (see Fig. 5.19). Further, this random rotation should not affect the maximum accuracy, as a proper rotation will preserve all of the relative separability information within a representation.

Sampling from all possible rotations of the ICL model's top-level unit-space is equivalent to sampling from all linearly equivalent representations without a bias towards unit specialization. Comparing the normal ICL model's UD curve to the distribution of UD curves of

the random rotations can effectively determine whether the ICL model's constraints cause it to learn a specialized unit representation or not without the need to train a separate model.

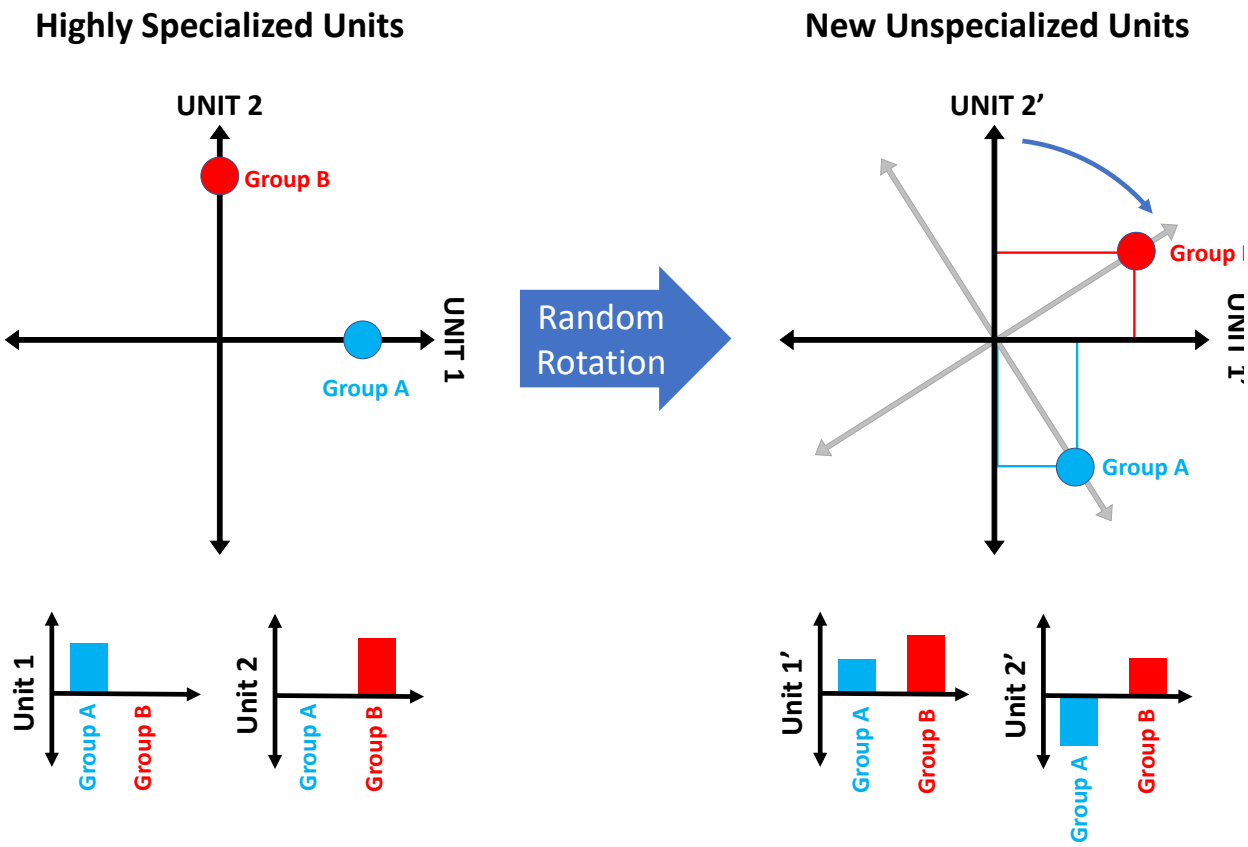

Figure 5.19. Random Rotation Unspecializes Neural Representations
This figure shows how performing a random basis rotation on the axes of a specialized unit space will result in a new unit-space that no longer has specialized units, as the rotation will distribute the selectivities or loadings of each of the specialized units randomly across all of the units in the new space.

5.3.3 Data Analysis

The purpose of this statistical analysis was to estimate the probability that the observed UD-AUC for the normal ICL model could occur by chance when its units are actually fully unspecialized. To accomplish this, I compared the UD-AUC of the normal ICL model with the distribution of UD-AUCs for the rotated versions of the ICL model, which were

guaranteed to be unspecialized while maintaining performance. From here on I will refer to these values as the normal-UD-AUC and the rotated-UD-AUC distribution. I performed a form of Monte Carlo permutation test, where the distribution of the rotated-UD-AUCs was re-sampled repeatedly. As a part of this, I tracked the percentage of times the rotated-UD-AUCs were below the normal-UD-AUC. This percentage estimates the likelihood that a model with no bias towards linearly specialized units would generate the UD-AUC of the normal ICL model by chance.

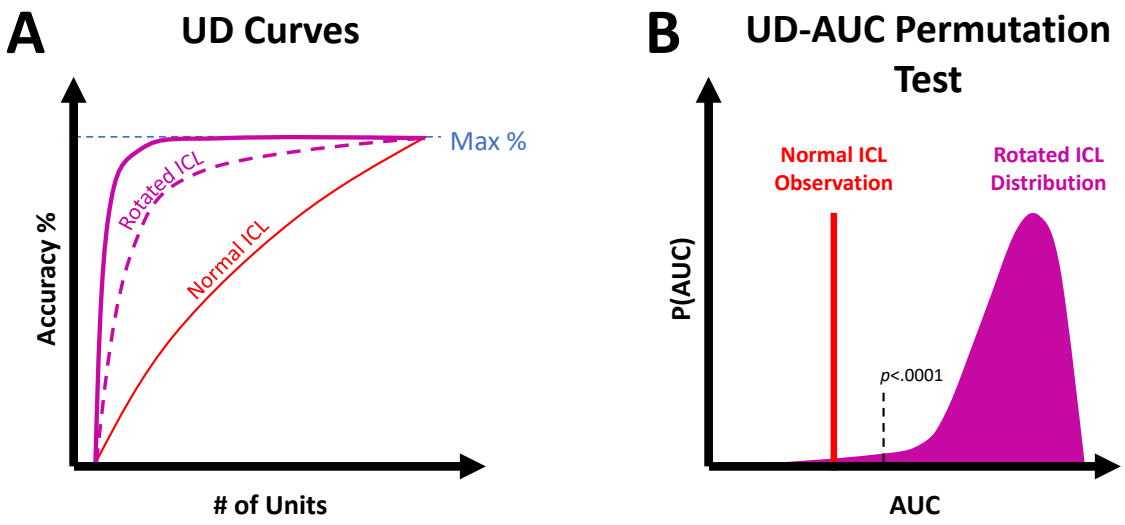

Figure 5.20. Expected Results: UD-AUC Monte-Carlo permutation test. (A) shows a sketch of the expected rotated-ICL UD curve and the normal-ICL UD curve. Note that I expect the rotated-ICL model to feature a strongly elbowed UD curve and the normal-ICL model to feature a much less elbowed curve. (B) shows a sketch of the expected result of the permutation test comparing the UD-AUC of the normal ICL model the permuted distribution of the rotated ICL models UD-AUC.

5.3.4 Results

For the main part of Simulation Study 2, I wanted to examine whether ICL learned more specialized unit representations than would be expected from a standard DCNN or DNN. To test this hypothesis, I sampled the UD curve for the ICL-Map-Optimal model and compared

its AUC to an estimate of its unspecialized AUC, using 15 samples for every point on the UD curve, 1 random rotation, and 1000 re-samplings. I found that the AUC for the normal trained version of the network was lower than its estimated unspecialized AUC distribution, but not enough to reach significance, $p < .1$ (see Fig. 5.22). As a baseline I also compared the AUC of an untrained version of the ICL-Map-Optimal model to its untrained AUC distribution. I found that the two distributions were not significantly different, $p < .19$ (see Fig. 5.21).

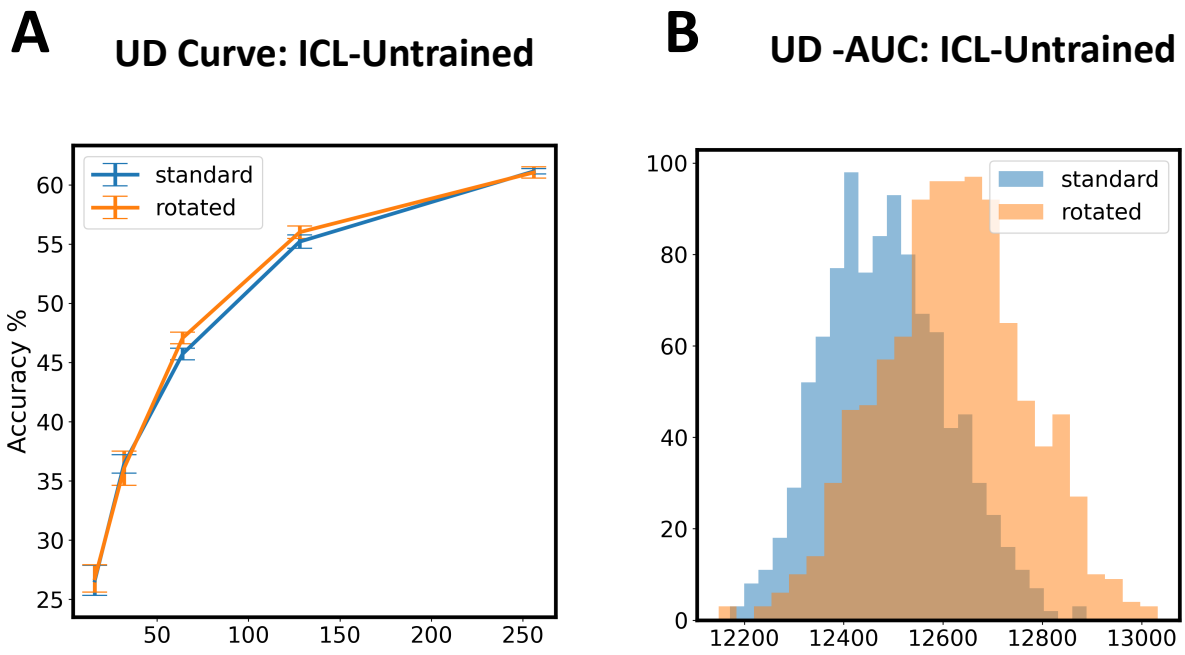

Figure 5.21. UD-AUC: ICL-Untrained
This figure shows the UD curve (A) and UD-AUC distributions (B) for the untrained MAP-Optimal model. For (A), note that there is very little separation between the untrained model's standard UD curve (BLUE) and the models rotated UD curve (YELLOW), where the error bars are 95% confidence intervals. For (B), note that the permuted standard and rotated AUC distributions overlap highly. This suggests that the untrained model's unit representations is virtually unspecialized.

As a followup, I wanted to examine if the apparent specialization in the ICL-Map-Optimal model could be due to its wide radius of lateral interaction in its top layers. To test this I

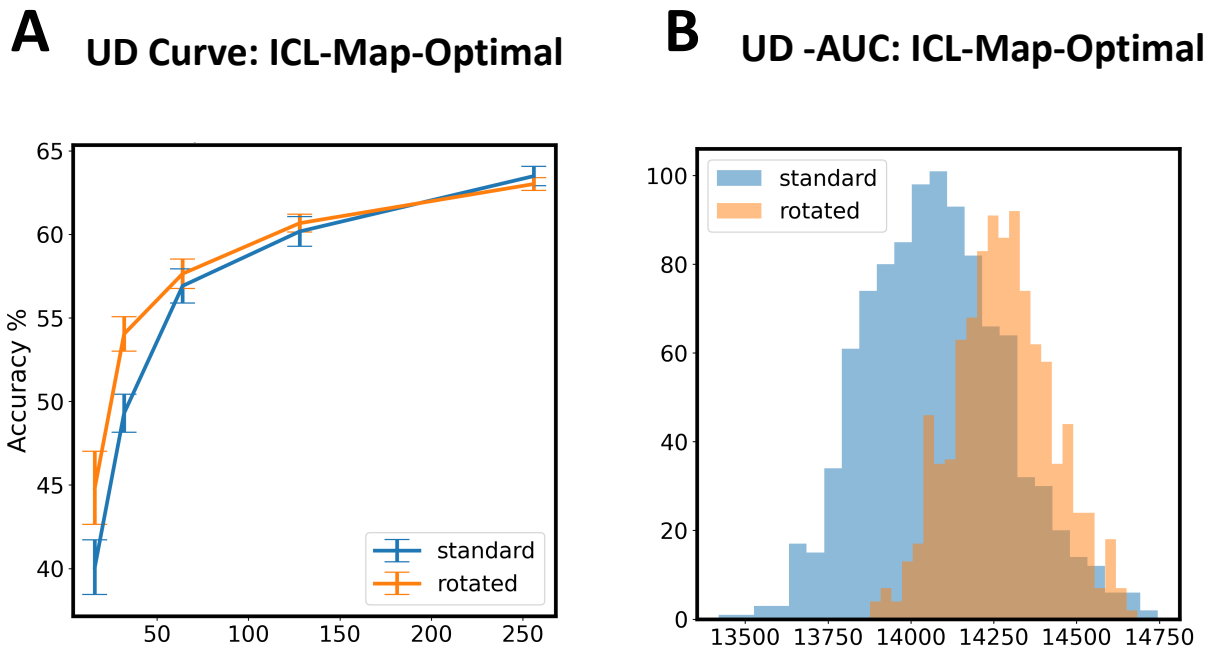

Figure 5.22. UD-AUC: ICL-Map-Optimal Trained
This figure shows the UD curve (A) and UD-AUC distributions (B) for the trained MAP-Optimal ICL model. For (A), note there is a small divergence between the model's standard curve (BLUE) and its rotated curve (YELLOW), where the error bars are 95% confidence intervals. For (B), note there is some separation between model's standard AUC (BLUE) and its rotated AUCs (YELLOW). This suggests that there may be a small amount of unit specialization compared to the virtually unspecialized rotated representation.

performed the same test using the ICL-Lat-Optimal model which features very small radii of lateral interactions to see if the effect was eliminated, and performed the test with 1 random rotation and 1000 re-samplings. I found that AUC for the ICL-Lat-Optimal model was still under its estimated unspecialized AUC distribution, this time the effect reached significance, $p < .001$ (see Fig. 5.23). However, the qualitative difference between the curves is minor.

As a further followup, I wanted to examine whether the addition of complex cell dynamics would increase the specialization. To test this, I sampled the PPU curve for the ICL-Complex model and compared its AUC to an estimate of its unspecialized AUC distribution, 15 samples for every point on the UD curve, 1 random rotation, and 1000 re-samplings. I

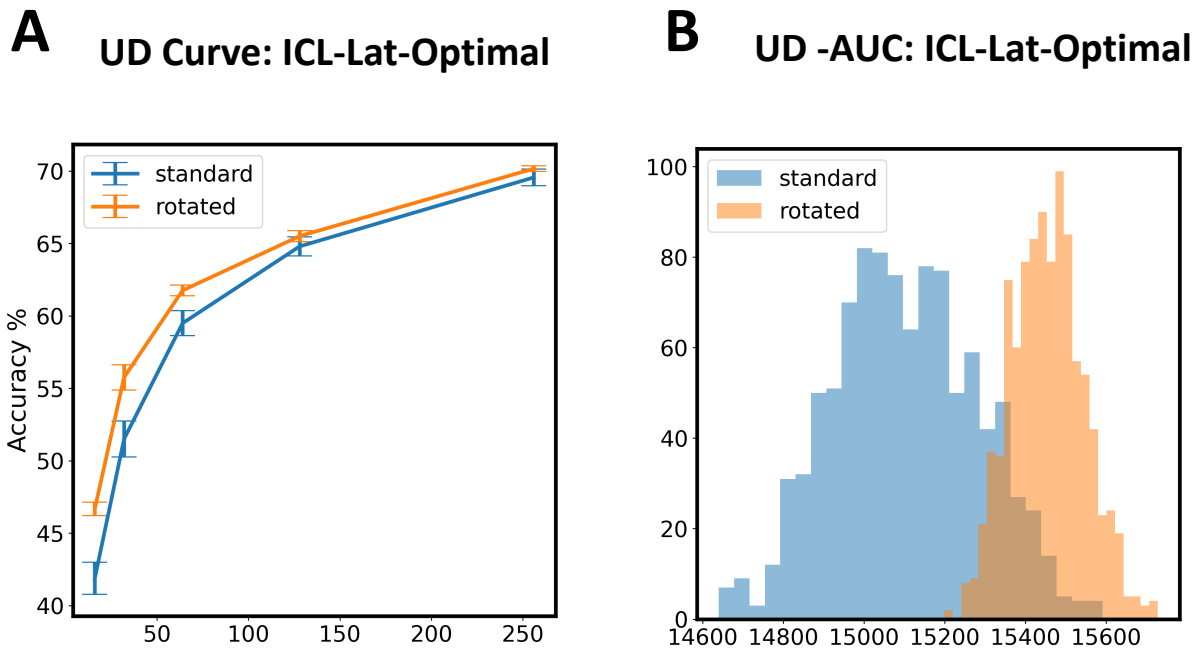

Figure 5.23. UD-AUC: ICL-Lat-Optimal
This figure shows the UD curve (A) and UD-AUC distributions (B) for the trained LAT-Optimal ICL model. For (A) note that there is a consistent gap between the standard (BLUE) and rotated (YELLOW) UD curves, where the error bars are 95% confidence intervals. For (B) note that there is an even more substantial deviation between the standard (BLUE) and the rotated (YELLOW) distributions for the ICL-Lat-Optimal model. This suggests that the ICL-Lat-Optimal model features some amount of unit specialization, though again it is still minimal.

found that the AUC for the ICL-Complex modes was under its estimated unspecialized AUC distribution, $p < .025$ (see Fig. 5.24).

Finally, I noticed that the UD curves of each of the trained models appeared to separate from their estimated unspecialized UD curves when fewer units were used. To test if this was more pronounced for smaller numbers of units, I re-calculated the ICL-Lat-Optimal model's UD curve and AUC and compared it to its estimated AUC distribution, but only calculated for 2, 4, 8, and 16 unmasked units. Additionally, because small numbers of units will behave less consistently, I increased the number of samples for every number of unmasked units to 40.

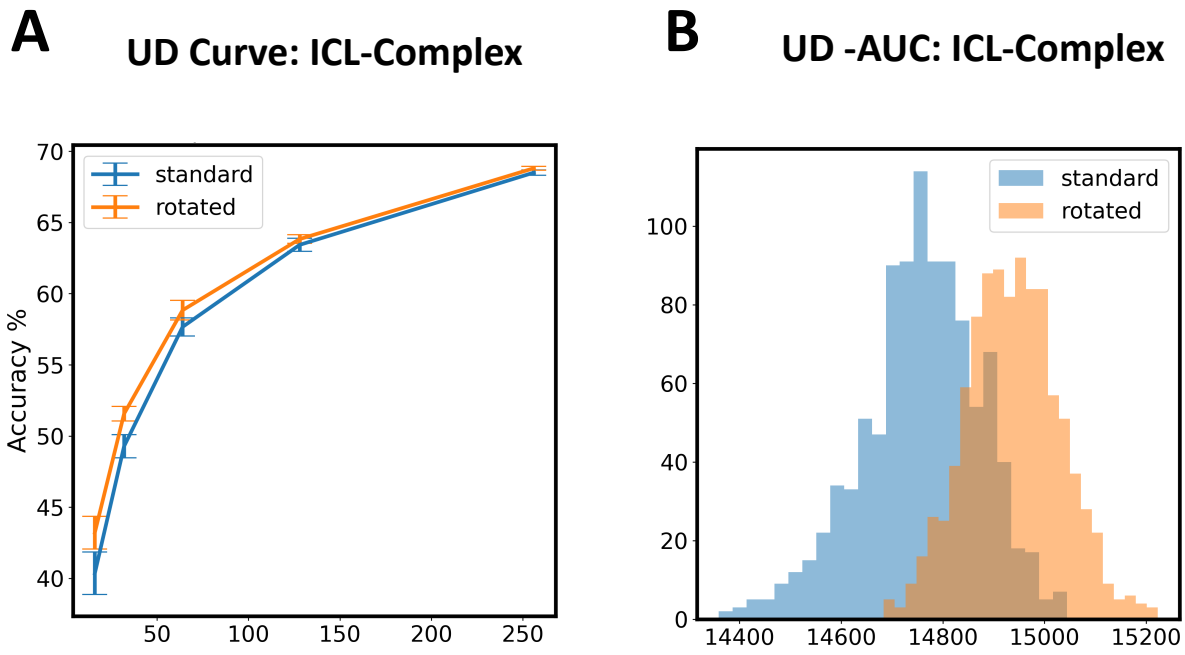

Figure 5.24. UD-AUC: LAT-Optimal-Complex
This figure shows the UD curve (A) and UD-AUC distributions (B) for the trained LAT-Optimal-Complex ICL model. For (A) note that there is a consistent gap between the standard (BLUE) and rotated (YELLOW) UD curves, where the error bars are 95% confidence intervals. For (B) note that there is an even more substantial deviation between the standard (BLUE) and the rotated (YELLOW) distributions for the ICL-Lat-Optimal model. This suggests that the ICL-Lat-Optimal model features some amount of unit specialization, though again it is still minimal.

I found that the ICL-Lat-Optimal model's AUC was not significantly under the estimated unspecialized AUC distribution, $p < .07$ (see Fig. 5.25).

5.3.5 Discussion

The past several decades of research have led to the hypothesis that high-level ventral stream neurons are highly specialized (Bednar and Wilson, 2016; Farah, 2004; Foldiak, 2003; Grill-Spector and Malach, 2004; Grill-Spector and Weiner, 2014; Schiltz et al., 2005), but DCNNs demonstrate that high performance can be achieved with apparently unspecialized units within a domain (Parde et al., 2021; Szegedy et al., 2014a). I predicted that the ICL model

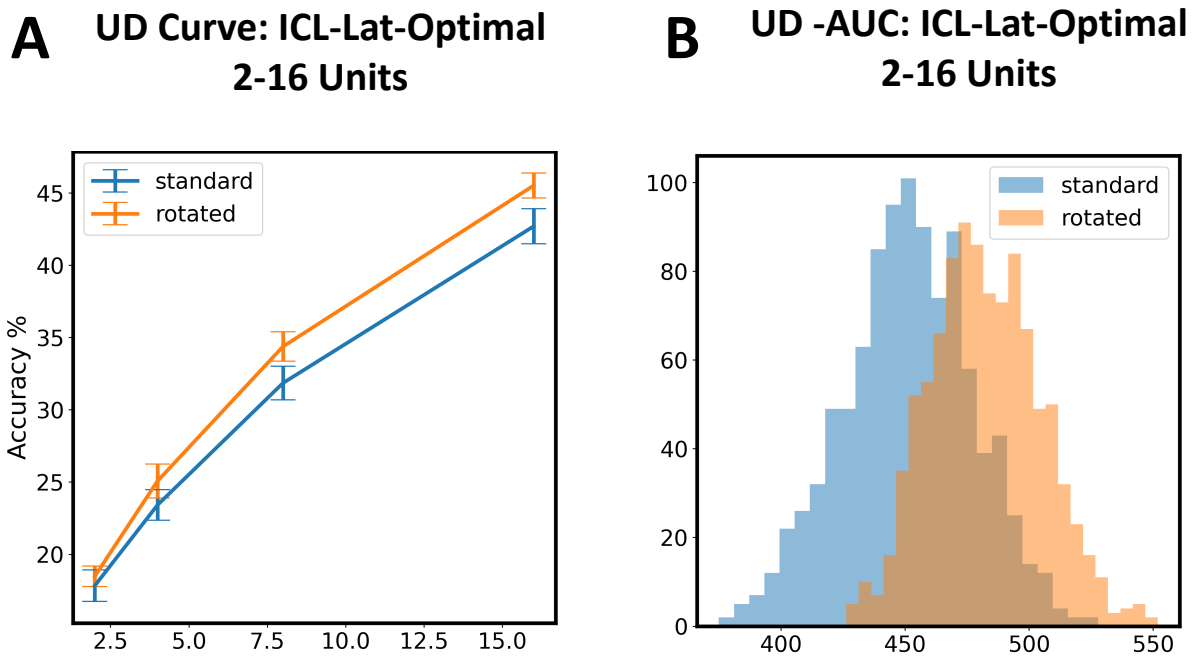

Figure 5.25. UD-AUC: Small Number of Non-Deleted Units: ICL-Lat-Optimal This figure shows the UD curve (A) and UD-AUC distributions (B) for the trained LAT-Optimal ICL model. For (A) note that there is a consistent gap between the standard (BLUE) and rotated (YELLOW) UD curves, where the error bars are 95% confidence intervals. However for (B), note that the overlap between the standard (BLUE) and the rotated (YELLOW) distributions for the ICL-Lat-Optimal model at these unit counts was not large. This result fits with the earlier results with large numbers of units taken into account.

would form specialized units more inline with current neuroscience theories. However, I generally did not find that to be the case.

In the first comparison made, I found that the ICL model with simple cell units does not seem to learn highly-specialized units (see. Fig. 5.22). Although the AUC distributions are somewhat separated, the difference between the ICL model's normal UD curve and its unspecialized curve were rather small, which suggests that there is not a qualitative difference in the level of specialization present compared to what would be expected in a standard deep neural network. Further, adding complex cells to the model did not greatly change the results (see. Fig. 5.24). Even at the lower unmasked unit numbers the models tend to very closely follow the idealized unspecialized UD curves (see. Fig. 5.25).

All together, the results of Simulation Study 2 suggest that the ICL model does not develop more specialized features than a DCNN model, which might indicate that biological constraints do not encourage unit specialization. However, these results should be taken with some skepticism as unit specialization would only really be expected when the overall representation become specialized for discriminating set of categories. However, as the results in the mapping portions of Simulation Study 1 suggest that high-level units of our implementations of the ICL model may largely represent low-level non-categorical features (despite depth of representation). In essence, this means that even if the units were becoming specialized that specialization may have had little to do with the category structure of this dataset and more to do with local edge detection.

This may seem somewhat paradoxical as all the ICL networks show highly selective neurons, like the one in Fig. 5.26. But, when I compared the top most selective neurons of the ICL model to a version that had been completely untrained the selectivities were generally indistinguishable (see. Fig. 5.27). As such, the appearance of "highly selective" neurons may be somewhat inevitable with large deep networks regardless of whether those neurons are actually functionally specialized, as a random network with no learning could not develop functional specialization. This follow-up suggests that the appearance of selective neurons in the ICL model may simply be due to the statistical tendencies of large networks rather than real unit specialization. Further, this suggests that the development of apparently categorically organized cortical maps does not imply unit specialization, as might be expected.

Given the results of the cortical map development experiments and the unit specialization experiment it would be useful to know what the computational properties of the ICL model are. Is it learning progressively more discriminative representations as more layers are added without supervision? How discriminative are its early layers and how do the feature it learns compare to early DCNN features and V1 features? If the performance does not increase

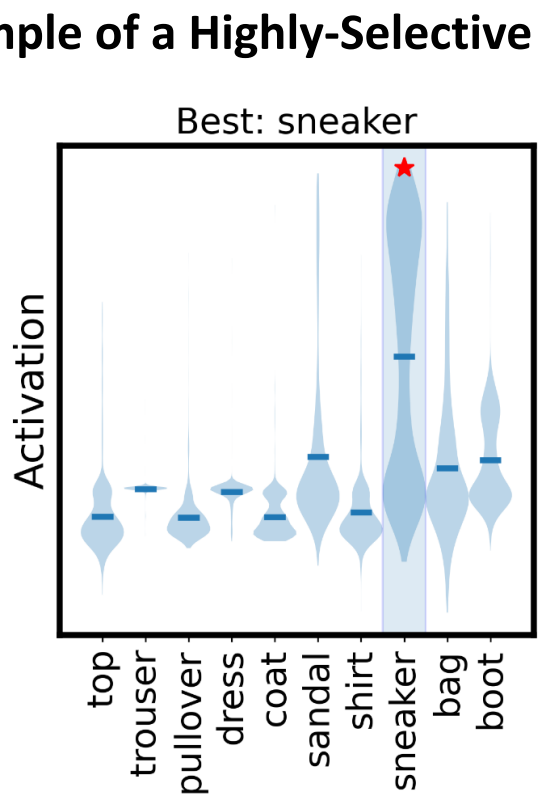

Figure 5.26. Example of a highly-Selective Neuron
This figure shows an example of the activation distributions across category for a highly selective neuron. Note that the activation distribution for "sneaker" extends much higher than all other categories and that the maximally enervating stimulus also belongs to the "sneaker" class. In neuroscience parlance this might be called a "sneaker neuron".

with each layer on classification tasks it would suggest that the unit representations are not becoming specialized because the network hierarchy as a whole simply is not becoming specialized for the task at all, beyond developing good low-level representations.

5.4 Simulation Study 3: Unsupervised Learning

5.4.1 Introduction

The ICL model was designed to support a form of unsupervised learning that does not require labeled images and does not require a mechanism for backpropagating error signals

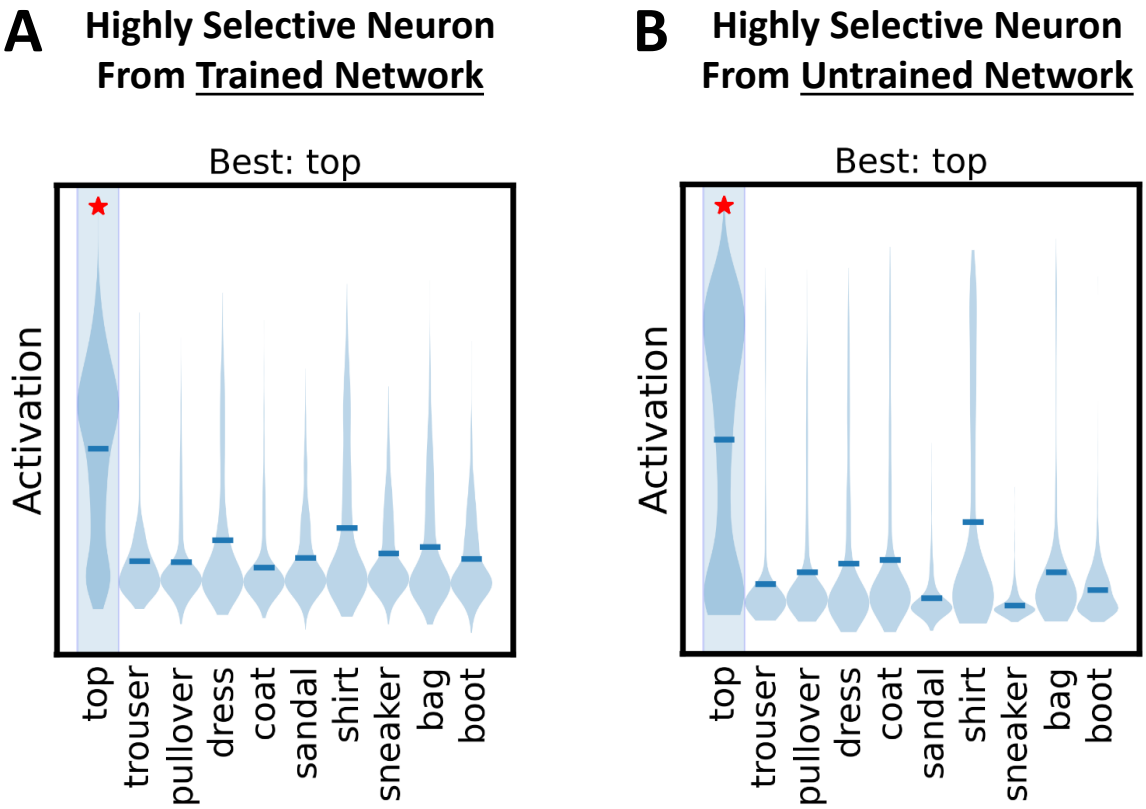

Figure 5.27. Highly selective Neurons, from Trained and Untrained Networks
(A) shows an example of a highly selective "Top" neuron from a trained ICL model. (B) shows an example of a highly selective "Top" neuron from an untrained ICL model. Note, that it is challenging to find a meaningful difference between these two neuron's activation distributions. Untrained and random network will tend to have many highly selective neurons like this. As such, the presence of such neurons does not guaranteed that units are being utilized in a specialized manner.

through multiple layers of the network. This is desirable from a computational neuroscience perspective, as human learning appears to use large amounts of unlabeled stimuli and because the backpropagation methods used in most deep neural networks are not thought to be neurally plausible (Yamins and DiCarlo, 2016b). However, the computational performance of a deep bio-inspired model like ICL has not been evaluated. If the ICL model can perform unsupervised visual category at a high-performance level, and improves its performance as it becomes deeper, it would suggest that ICL would be a promising alternative architecture to DCNNs.

The goal of Simulation Study 3 was to test whether a more neurally conservative model like ICL could learn category structure in a fully unsupervised manner (i.e. entirely without labels) and without deep backpropagation. To achieve this goal, I exposed the ICL model to a series of object recognition datasets augmented with temporal dynamics until its representations stabilized, and then I trained a separate readout network independently to examine the classification performance of each layer in the ICL model. Further I looked at the impacts of including axonal development dynamics and complex cell learning dynamics on the network's performance.

5.4.1.1 Predictions

Based on the novel mechanisms of the ICL model, I predicted that my implementation of the ICL model would learn high-level representations of the categories in these datasets that support a high level of classification accuracy, even without supervised labels.

I tuned the ICL model and the temporal augmentation scheme for each dataset in an attempt to maximize performance. At a high-level TRM-Trace learning requires that activations in the simple cell layers and the complex cell layers have a certain level of sparse activation. If too many neurons are active, then the temporal trace will experience something called whiteout where all inputs lead to indistinguishable activation patterns. Further, if the

activation patterns in the simple and complex cell layers are too sparse in their activity and the temporal patterns evolve too slowly, then the trace learning will be unable to successfully link related simple cell representations correctly. Many factors affect activation sparsity such as timescale, trace rule parameters, and radius of inhibition/excitation. Understanding what settings are successful could be useful from both neuroscience and practical standpoints.

5.4.2 Simulation Methods

5.4.2.1 Datasets

For Simulation Study 3 the first dataset I trained the model on Fashion-MNIST which contains 28 by 28 gray scale images of 10 categories of clothing items (Xiao et al., 2017) (see Fig. 5.28A). Fashion-MNIST is designed to be a replacement for the original MNIST for the purposes of prototyping and developing new neural networks, because MNIST was generally too easy for modern machine learning techniques to be validated on. The second dataset will be Imagenette which contains 160 by 160 color images of 10 real world objects categories (Howard, 2019) (see Fig. 5.28B). Imagenette is a much smaller sample of the well known ImageNet dataset designed to be easy to use, while still having the difficulty of real world object recognition built in.

5.4.2.2 Model Configuration For MNIST-F

For the MNIST-F dataset, I followed the same simulation setup as in Simulation Study 1 Sec. 5.2.2.2

5.4.2.3 Model Configuration For ImageNette

First I scaled the original input images stored in values of 0-255 RGB to between 0 and 1, then I performed a square root on these values to make the values approximately linear so that color-mixing and blurring assumptions would be better satisfied. For more information

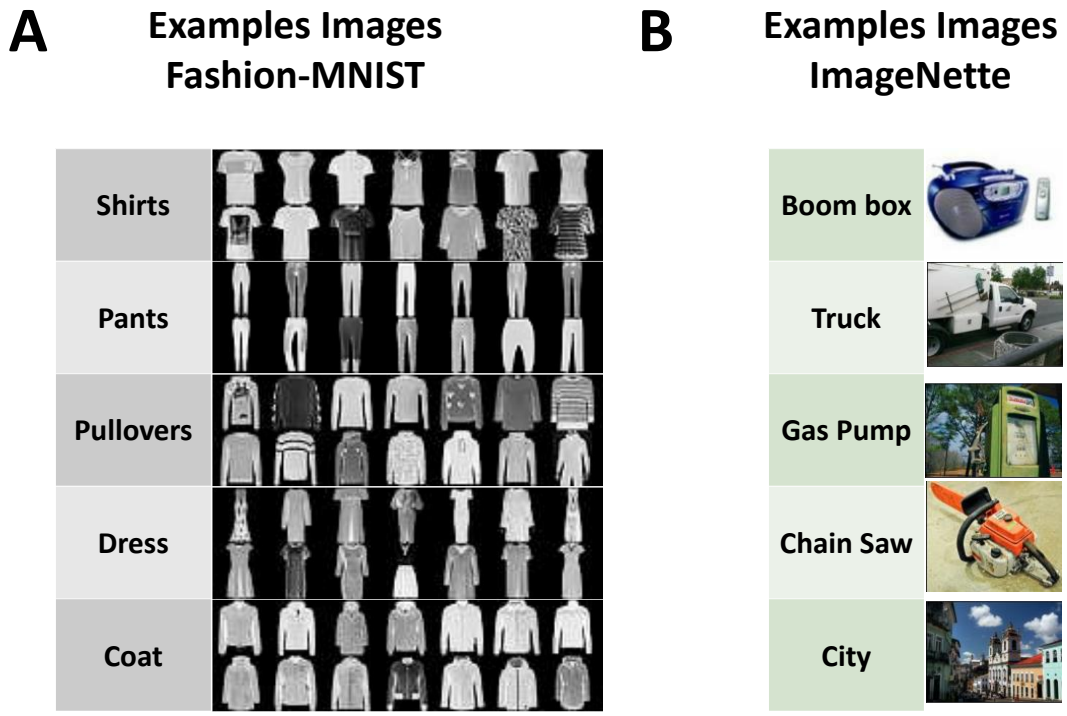

Figure 5.28. Examples of the Fashion-MNIST and ImageNette Datasets
(A) shows example images for several categories in the Fashion-MNIST dataset. (B) shows example images for several categories in the ImageNette dataset.

on why the square root is helpful it is important to know that most digital images are stored in the sRGB format, also called gamma color coding, which takes linear color value reading and applies and raises them to the power of 2.2 [1] in order to better utilize available storage by taking advantage of the non-linear nature of luminance perception in human vision (Anderson et al., 1996). Unfortunately, raising luminance values to a power breaks the assumptions of additivity usually assumed when performing many image operations such as convolutions with different spatial filters. For the ICL model, correcting for the gamma color coding lead to a modest 1% performance increase.

[1]This is actually an oversimplification as the RGB channels are passed through a matrix multiplication first and then further scaled before the power is applied

After scaling the RGB values, I re-scaled the size of the images down from 160 x 160 to 64 x 64 for the ICL model. Then I processed them using a special combination of the simple low-pass filter and a special color-openent filter that I will detail below.

The color opponent process I used subtracted each color channel from an average of the other channels that had been Gaissian filtered. Further, the width of these Gaussian filters were twice the size of the filters used in the simple high-pass filter typically use. This lower-frequency color opponent result was then spliced with the standard high-pass filter output such that every fourth pixel in a block of 4 pixels would be a color-opponent outputs and the other three would be the simple high-pass outputs. I call this pre-processing method the color splice method See Chapter 4 Sec. 4.5.3 for more information.

The color splice method was used for ImageNette because color information is processed by the LGN in an opponent fashion, and more color information tends to be contained at lower spatial frequencies than for luminance information. Further, the contribution of the color-opponent information needs to be scaled up so that V1-like simple cells will afford more representation to color contrasts. Otherwise, I found that luminance contrasts will completely dominate the learned V1 representations, leading to an information bottleneck early in the network.

For the ImageNette dataset I used 4 stacked ICL modules with a map resolution of 200 x 200 for each ICL module. For this version of the model I chose to use only the Simple Only configuration of the ICL module as testing with the easier dataset suggested that this configuration would perform best and take less training time.

5.4.2.4 Training & Testing Procedure

The benchmarking process consisted of two training stages. First, a multi-layer ICL model was exposed to image sequences from a dataset, with no labels, until all its layer representations stabilized. Second, a linear readout layer was added to the top of the ICL model

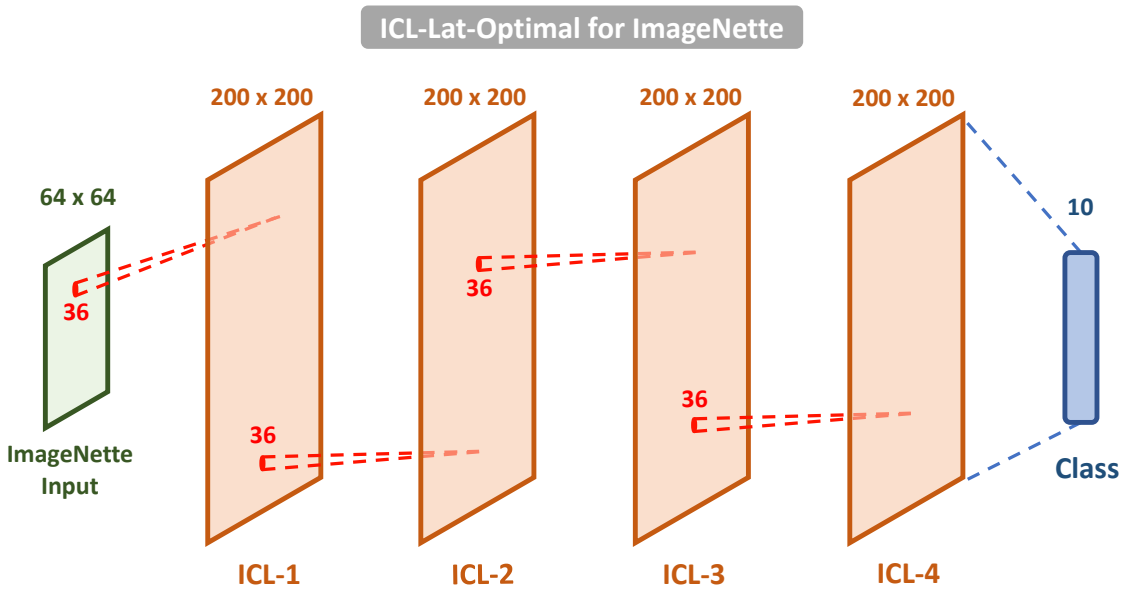

Figure 5.29. Network Diagram: ICL-Lat-Optimal: ImageNette
This diagram shows the layer setup for the ICL-Lat-Optimal model trained on ImageNette. As the Simple Only version of the ICL module was the only one tested on this dataset the windows connecting each layer (RED) are essentially fixed disks. The RED NUMBER indicates the total number of connections in each window. This layer structure was found to work reasonably well, though I was only to do minimal hyper-parameter tuning.

and trained to discriminate categories based on labeled images. While the readout layer was trained, the weights of the ICL model were held fixed, (i.e. no deep backpropagation). Then I measured the accuracy of the combined ICL and readout layer on a validation set. For an overview of the training procedure see Sec. 5.2.2.4 and for a graphic overview of the training process see Fig. 4.2.

5.4.3 Results

As part of the main question of Simulation Study 3, I wanted to understand if ICl could produce computationally useful representations (i.e. learn representations which could classify categories in its environment). To examine this I trained the ICL-Lat-Same model on two different datasets, MNIST-F and ImageNette, and tested its performance on the test

splits. For MNIST-F, I found ICL performed quite well at a maximum performance of around 91.4% (see Table 5.30). This is well below the *State of the Art* (SotA) of 96% (Tanveer et al., 2020), but well in line with average DCNN performance on this dataset. For example, a similarly sized DCNN network I implemented with batch normalization and top-level dropout achieved a similar performance of 91% (see Table 5.30). For ImageNette, ICL only manged to achieve a maximum performance of 40.2%. By comparison a similarly sized DCNN I built achieved 48% and a similarly sized Deep Locally Connected Network I built achieved 62%. Both of these comparison networks were only modestly refined for the task. The SotA for ImageNette is generally much higher, between 90%-95% (Howard, 2019), but this may be more typical for networks that process ImageNette at a much higher resolutions than those used for the models of this paper.

There is a caveat to this performance however. The highest performance for both datasets appeared to occur in the lowest layers of the network. To follow up on this I examined the performance for each layer on the MNIST-F dataset. I generally found that performance diminished for every layer added to the ICL-Lat-Same model with performances at 89.7%, 89.1%, 88.21%, and 87.9%, for each consecutive layer. Adding axonal learning appeared to have little effect with performances at 89.5%, 88.8%, 88.3%, and 88.0%, for each consecutive layer. Adding complex cell learning also had no positive effect, with performances at 89.1%, 87.7%, 87.5%, and 86.1%, for each consecutive layer.

The lack of improvement over layers with complex cells enabled suggested that they may not be improving generalization over feature variability. To examine this further I re-ran the experiments, but varied the test dataset by specific pixel shifts laterally, which gave generalization curves over image translation. Generally, the addition of complex cells did not improve generalization over translation, even when the best performing layers were chosen (see Fig. 5.33). Additionally, generalization performance appeared to go down as more layers of complex cells were added (see Fig. 5.34).

Comparison of Best Scores	Network	MNISTF	IMAGENETTE
	(ICL) 1-layer (Bio-inspired)	**91.4%**	**40.2%**
	(ICL) 4-layers (Bio-inspired)	**89%**	**37%**
	DLCN + Sup Deep Learn 4-Layers	**91%**	**62%**
	Linear Net 1-Layers		**19%**
	Basic DCNN 4-Layers	**90%**	**48.6%**
	State of The Art DCNN more than 4 layers	96%	95%

Figure 5.30. Table of Performances
This table shows a comparison of typical performances the ICL models developed for this study and several comparison models. BLUE rows indicate classification performance for the novel ICL models. BLACK BOLD performance estimates were generated as a part of this study. BLACK NON-BOLD performance ranges were reported in Howard (2019) and Tanveer et al. (2020). As a general note the ICL model performances reflect the maximum observed performances during the course of the study. In practice, the performance of the ICL models tended to fluctuate within a range of approximately one percentage point.

5.4.4 Discussion

The primary result of Simulation Study 3 shows that at least on some datasets ICL based models can perform fairly quite well compared to DCNNs of similar complexity and tuning. For example, on the MNIST-F dataset top-tier performance is only 96% and seriously focused work is usually needed to break past 90% with our basic comparison DCNN performing well below that. Both the full ICL model and the ICL-Simple model performed close to 90% on the MNIST-F dataset within their first layers typically, suggesting that even as shallow models they might have practical utility. Overall, this finding shows that ICL-like models can perform well enough to be considered worth studying further as a possible bridge between

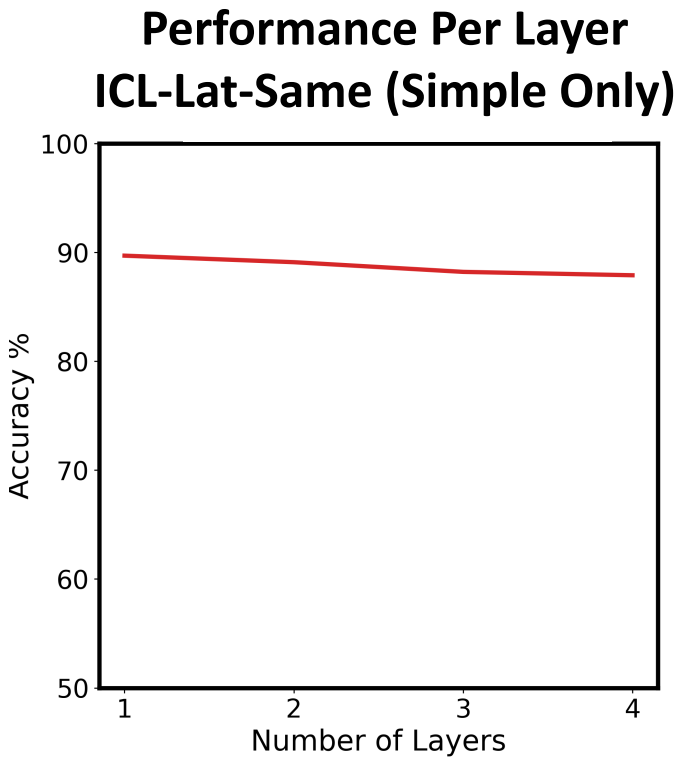

Figure 5.31. Performance Per Layer: Simple Only
This figure shows the performance of the Lat-Optimal ICL model with simple cells only as a function of the number of ICL modules it has. Note performance generally decreases as the number of modules increases.

studying the biology of the brain and contemporary deep-neural network approaches. Further it suggests that unsupervised/biologically motivated learning paradigms can be successful on difficult datasets common in today's literature.

The second most important result in Simulation Study 3 shows that performance generally decreases with the number of layers added to a ICL based model on MNIST-F. This suggests that ICL, with the its current simple cell and complex cell implementations, is either missing a critical mechanism or some kind of specialized tuning required to improve category separation as more representational depth is added. This contrasts with typical

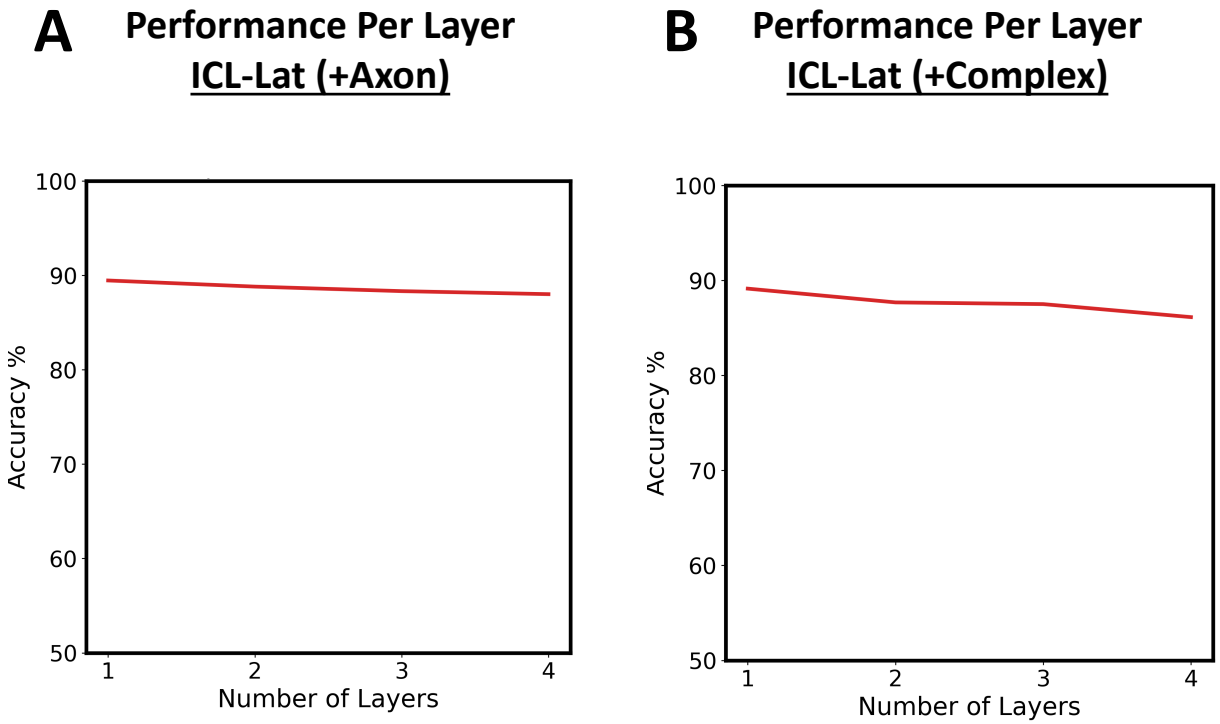

Figure 5.32. Performance Per Layer: Axon and Complex cell Learning
(A) This figure shows the performance of the Lat-Optimal ICL model with simple cells and axonal learning as a function of the number of ICL modules it has. Note performance generally decreases as the number of modules increases. (B) This figure shows the performance of the full Lat-Optimal ICL model as a function of the number of ICL modules it has. Note performance generally decreases rapidly as the number of modules increases.

deep-supervised networks whose classification performance generally increase (up to a point) when more layers are added to their representations.

The lack of improvement for the models which only included simple cell learning is not too surprising. The simple cells tend to expand an input representation into a new over complete set of features that are more complex than the original features. However, these simple cell features need some method of reduction such that features which tend to belong to the same object are conjoined in some way. This is what the complex cells are supposed to do; without this mechanism the simple cells have no way of building tolerance to image

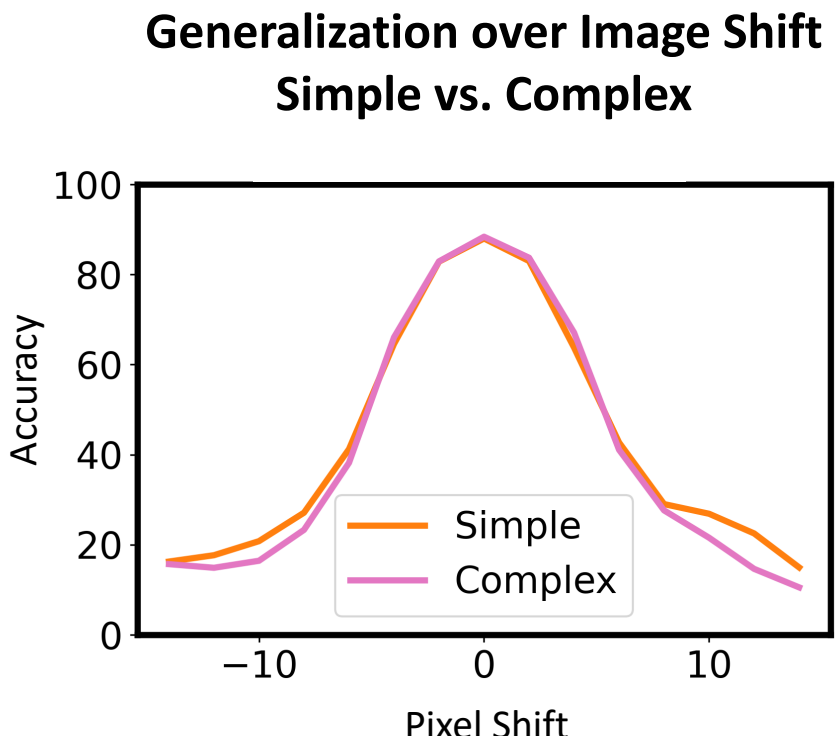

Figure 5.33. Generalization across image shifting: Simple Vs Complex
This figure compares the generalization curves over pixel image shift for an ICL model with simple cells and an ICL model with simple+complex cells. I picked the best generalization curve generated from the layers of each model (here the 4th layer for the simple only model and the 1st layer for the complex only model). Generally, the simple only model is almost identical to the complex cell model for narrow pixel shifting and slightly better for wide pixel shifts. This suggests that the complex cell layers are not building tolerance to image variability (which is their functional purpose in the architecture).

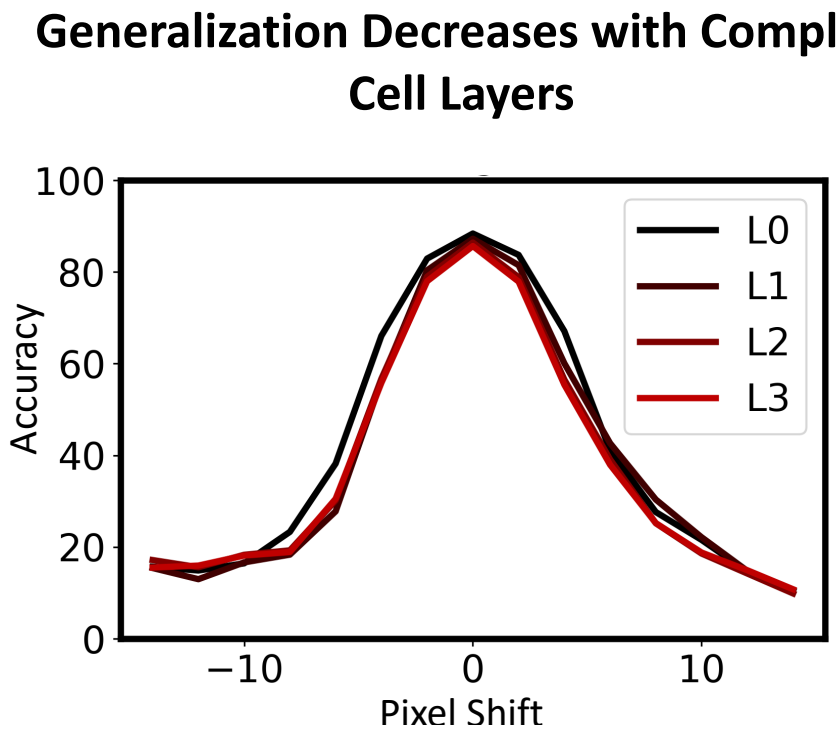

Figure 5.34. Generalization performance of ICL Complex Layers
This figure shows the generalization curve over pixel shifting for each layer of an ICL model with simple and complex cell learning. Generally, the generalization curves become more narrow and show worse performance at all levels of shift as the layer number increases. Again, this suggests that the complex cells are not performing their intended function.

variation. However, in our experiments the addition of complex cells did not help the ICL models' performance either.

The lack of improvement over multiple layers that included complex cell learning may indicate that some level outside information about category structure is needed to shape the visual pathway in a top-down fashion. This result also may indicate that observations of objects over their own visual variability is not enough to improve category separability. However, these results may simply indicate that the Trace learning used these models was insufficient to produce good pooling properties in the complex cell layers. In support of this notion, I generally found that most forms of classic trace learning produced overly redundant

pooling regimes, pooling either identical features or exceedingly correlated features rather than using the temporal signals to develop more useful pools. This result seems to be a product of existing Trace-Rules being poorly suited to learning environments with windowed local connectivity and large numbers of non-sparse simple cell activation (for a concise definition of trace rules and trace learning see Sec. 4.4). Many of these algorithms were conceptualized in fully-connected learning environments with only one simple cell activation at a time. In future studies, these observations about complex cell function within a deep architecture may be useful for designing new forms of trace-learning that are better suited to using windowed local connectivity and multiple simple cell activation's.

Additionally, axonal learning made little impact on performance in Simulation Study 3. This may happen for a simple reason. The axonal learning is directed by the correlational statistics of its axonal arbors. If the representation from the lower-level layers (which have correlations that encourage retinotopy) are simply being propagated to later layers, then the axonal learning will have little incentive to deviate from strongly retinotopic mappings. Given that the initialization for the ICL model's connectivity is already retinotopic, it is unsurprising that performance is not strongly affected, as the connectivity will not change much under these circumstances.

One central challenge in progressing models like ICL to be better hierarchical models, is that our understanding of what constitutes a good representation each layer of the ICL model is gated by our understanding of the ventral visual hierarchy. It is relatively easy to see if the simple cell representation is learning correctly in the V1 analogue of the model because the representation in V1 is very well studied. On the other hand the V2 is much less well understood and V3-IT are poorly understood in terms of their neural representations. A model which simply arrives at a good V1-like representation is not guaranteed to do the same for V2 through IT, even if the cortical algorithm for these areas in the brain is largely similar. One of the benefits of testing the performance of each layer as I did in the followup

to Simulation Study 3 is that this approach could be useful for validating if other cortical models are actually learning more sophisticated and specialized representations.

Overall, the lack of improvement over multiple layers on this study's implementations of the ICL model signals that more conceptual and algorithmic development is needed. Conveniently, the ICL platform developed in this study would serve as an excellent test bed for solving these issues.

CHAPTER 6

GENERAL DISCUSSION

6.1 Introduction

In this chapter, I will summarize the work performed in this dissertation and its contributions. The chapter will be divided into three main sections: an overview of the simulation work in context, a discussion of theoretical contributions and ramifications, future directions, and general conclusion.

6.2 Simulation Work in Context

6.2.1 Simulation Study 1: Cortical Map Development

6.2.1.1 Overview

In Simulation Study 1, I wanted to understand if the *integrated cortical learning* (ICL) model developed for this project could learn high-level cortical feature maps similar to late areas of the ventral pathway. I specifically wanted to know if the ICL model developed contiguous representations at its top level that were selective for categories in its training dataset.

The simulations generally showed that the basic ICL model (with only simple cells) developed for this project learns category selective representations that varied continuously across its top-level cortical surface. However, when I examined these maps for all layers of representation, I generally found that they did not appear to change much from the first layer to the last layer of internal representation. This suggested the model simply preserved the early Gabor-like representation, into higher layers. Further, it suggested that the apparent organization by high-level categories in its top layer may be due to biases in the dataset and to the continuous change of features over the cortical surface.

When complex cells were added to the model, some of the qualitative aspects of the maps changed, and the overall size of the category selective regions increased with more layers.

But the difference was not large. This suggested that the complex cells mildly increased the development of category based topology in the top layer.

ICL may be one of the first deep cortical map based learning architectures in cognitive modeling, and Simulation Study 1 demonstrates that these architectures are viable in that both their lower and upper layers can generate interesting cortical map organization that can be related to the kinds of topology found in visual cortex. Simulation Study 1 only investigated a limited range of learning dynamics, but the space of possible cortical map learning algorithms is vast and variation of these dynamics could lead to large differences in cortical map development. Exploring the untapped potential of deep cortical map learning algorithms, like ICL, can help us better understand the types of dynamics at play across the ventral stream.

6.2.1.2 Related Work

Our work on creating deep cortical maps was greatly inspired by the ongoing work in creating models of V1 cortical maps. One of the first to do this was von der Malsburg (1973), which introduced the idea of using narrow local excitation and wider local inhibition, combined with hebbian learning and reciprocal activation to develop orderly arrangements of orientation selective neurons. This work later inspired larger and more advanced models like LISSOM (Sirosh and Miikkulainen, 1994) and GCAL (Stevens et al., 2013). LISSOM incorporated learned lateral connections, and GCAL incorporated gain control and homeostatic mechanisms. My ICL model uses simplifications of several of the concepts in LISSOM and GCAL. Versions of ICL used in these simulation studies did not include reciprocal activation or learned lateral connections, as I found that the maps and features learned without these mechanisms were largely sufficient. Our work greatly expands the capability to simulate these types of models at a large scale and on real world data. Further, ICL addresses the biologically implausible connectivity assumptions used by most of these models models by incorporating axonal learning via the axon game and the arbor model.

More recently, the most similar study to this dissertation work on deep cortical map modeling I could find was Lee et al. (2020). In Lee et al. (2020), the researchers augmented a standard DCNN model with a self-organizing map model at its top layer that minimized a version of the wire-length objective. Their model was able to demonstrate the development of high-level cortical map features such as face and object selective areas, which is very reminiscent of the findings in IT cortex. While the authors posit that their work is a large step forward, they acknowledge that the next step would be building a model that uses self-organizing map dynamics at all levels of representation as its primary form of learning, without the use of convolution layers. The ICL model can be viewed as an attempt to build that next step of this work, and further as a step towards more biologically plausible unsupervised learning in conjunction with self-organizing map learning. Based on Lee et al. (2020) it may be interesting to place an ICL module on top of a traditional supervised DCNN network and to see if it develops a more interesting categorical cortical map than our current experiment indicated.

6.2.2 Simulation Study 2: Unit Specialization

6.2.2.1 Overview

In Simulation Study 2, I wanted to understand if the ICL models developed more specialized unit representations than would be expected from standard deep neural networks or DCNNs. Based on its unique cortical learning inspired dynamics I expected that it would develop more categorically specialized unit representations at its top level, but not necessarily an extremely specialized representation. I expected this because ICL and cortex both have competitive learning dynamics which may lead to unit specialization as a byproduct, but demonstrations with neural networks also reveal that unit specialization is not needed for strong functional performance.

To examine this, I designed a method of characterizing unit specialization in terms of how the network resisted unit deletion. Essentially, networks that depend on highly-specialized categorical units will tend suffer more when large numbers of units are deleted [1], whereas networks that depend on units which randomly distribute categorical representation will suffer less when units are deleted. In order to generate a baseline for comparison, I also developed a method for "un-specializing" unit representations while preserving their overall performance. This form of analysis is general enough to be easily applied to arbitrary neural network architectures and to real multi-channel neuronal recordings.

Using this new analysis to examine the ICL model, I generally found that it did not produce strongly specialized units compared to its baseline unspecialized representation. While the units were slightly more specialized than the baseline it did not amount to a qualitatively different level of specialization. As a follow-up we also checked to see if this apparent level of specialization could be due to the redundancy of features caused by wide lateral interactions, by retesting with the ICL-Lat-Optimal model which had short lateral interactions. Again the model displayed only a small amount of specialization compared to baseline, suggesting that the model is developing a very small amount of unit specialization. This is actually quite similar to the amount of unit specialization found in DCNNs when I conducted a similar pilot study. The presence or absence of complex cell and axonal learning did not appear to affect unit specialization substantially.

The lack of categorical unit specialization developed by the models, despite the presence of dynamics which should encourage unit specialization, could be explained in several ways.

[1] In practice, the unit deletion curves of specialized encodings can be quite diverse, depending on the exact nature of the code, the number of categories, and the number of units. For example, a specialized one-hot code with 100 neurons and 100 units would see linear degradation, while one-hot code with duplicates with 100 neurons and 10 categories would feature variable deletion curves depending on the reliability of the units. In extreme cases, highly specialized representations can feature better deletion curves than unspecialized representations, due to an excess of redundant highly reliable units. Given this, it is worth noting that the equivalent unspecialized version of a representation will tend to have a very consistent unit deletion curve, and thus a large difference between the normal representation and its unspecialized variant will still indicate the presence of specialization, even if its normal deletion curve is an extreme one

The most likely explanation is that the units did in fact become specialized, but for features that are only tangentially related to categories. As suggested earlier, I theorize that the ICL models simply propagated low-level feature representations, such as Gabor wavelets, to their top-layers. Features like these, though not adept at representing semantic categories, can be still be thought of as representing semantic categories in a distributed or unspecialized fashion. Given these shortcomings, it would be useful to perform similar tests on an ICL like model that developed more sophisticated hierarchical relationships.

The results of Simulation Study 1 and 2 suggest an interesting conundrum. The simulated high-level cortical area appears to develop contiguous category selective regions, yet its neurons are not particularly specialized for categories. This appears to be possible due to the difference between neuroscience measures of specialization (i.e. selectivity & preferred stimulus), which are typically used for visualizing cortical maps, and our deletion-curve based computational measure of unit specialization. Similar conflicts might arise when comparing both types of measure on cortical recordings in actual tissue. As such, it would be useful to use the deletion curve paradigm in Simulation Study 2 to better understand the functional unit specialization in IT cortex overall and within specific functional areas like FFA. It is possible that despite the strong domain specific topological organization of IT (i.e. faces, places, object, .etc), that its units could be quite unspecialized in a strict computational sense. The same may also be true within specific functional areas.

6.2.2.2 Related Work

The approach to unit specialization for this dissertation was largely inspired by both Szegedy et al. (2014a) and Parde et al. (2021).

In Szegedy et al. (2014a), the researchers noticed that DCNNs learned top-level neurons that appeared to have semantically meaningful selectivites, such as selecting for sun flowers or particular shapes. However, they also noted that transforming these features with a

random basis transformation also produced units with apparently meaningful individual selectivity. As such, this work advocated skepticism towards interpreting the presence of seemingly semantically meaningful unit selectivity to mean that the networks actually learn functionally specialized units.

Parde et al. (2021) approached specialization from a different angle. In their work, they examined unit deletion curves for various SotA face recognition networks, and found, amongst other things, that the networks could maintain high levels of accuracy for classifications of identity and viewing angle, even when huge numbers of their top level units were deleted. This suggested that their high-level units were not particularly specialized for either of these characteristics (i.e. small populations representing small regions of these state spaces). They generally suggest that information in these networks is largely encoded at the ensemble or space-level, which also corresponds to a distributed code in some circumstances (Foldiak, 2003). This inspired an unpublished work where I discuss how the standard classification objectives for today's DCNNs actively encourage distributed coding by default, and how it is an optimal solution for the problem they are trying to solve.

The criticisms regarding strong interpretations of neural selectivity are actually quite old. In the 80s Anderson and Mozer (1981) demonstrated that a neural network that was known a-priori to have a distributed unspecialized unit code would still display many units with semantically interpretable selectivity, simply as a matter of statistical inevitability. Further, Anderson and Mozer (1981) showed that eigan analysis and singular value decomposition of weight vectors was a more fruitful approach to understanding distributed codes and unspecialized representations.

Our work in Simulation Study 2 can be viewed as a synthesis of ideas from Szegedy et al. (2014a), Parde et al. (2021), and my own unpublished work. Effectively, perfectly distributed unspecialized representation for a set of categories should have a very specific

resistance to unit deletion[2]. I realized that if this resistance could be estimated then I could measure deviation of a network's actual resistance to that value in the form of a statistical test. Conveniently this form of analysis is very general and could be potentially applied to other neural networks and the potentially neural recordings. Finally, the results of Simulation Study 1 seem to support the view that visual cortical neurons may not actually be functionally specialized suggested by Szegedy et al. (2014a), Parde et al. (2021), and Anderson and Mozer (1981).

6.2.3 Simulation Study 3: Unsupervised Learning Classification Performance

6.2.3.1 Overview

The main goal of Simulation Study 3 was to test if the ICL model's unsupervised learning was successful and if it has potential as a more biologically founded alternative to DCNNs for cognitive modeling research. To test this I compared the performance of the ICL model developed for this project to other models such as supervised DCNNs and supervised DLCNs, in order to get a relative understanding of its performance.

For the simpler dataset (MNIST-F) the model was trained on, ICL performance was quite strong. Generally, it performed as well as a typical DCNN with the same number of layers. However, most of this performance appeared to come from the ICL model's early representation (i.e. V1-like features), while subsequent layers appeared to hinder performance.

On the more difficult dataset (ImageNette), the model performed significantly worse than a basic DCNN with the same number of layers. Here the task almost certainly required a

[2]In practice, the deletion curves of highly unspecialized representations, will depend on factors such as the number of units that actually have useful loadings for the categories and the relative strength of their loadings, as was seen in Parde et al. (2021). When I refer to a perfectly distributed unspecialized representation, I am referring to a representation where the loadings for the categories in question have been completely randomly and indiscriminately distributed across the units. It is almost surprising, given the opportunities for variability, that both DCNNs and ICL can follow unit deletion curves for perfectly distributed and unspecialized representations so closely as was seen in this study and the pilot.

more sophisticated hierarchical representation these ICL models did not develop. Although the ICL models performed poorly, it should be noted that they achieved 40% accuracy compared to the baseline DCNN which achieved 48% accuracy. State-of-the-art for DCNNs on this dataset are closer to 90% accuracy, suggesting that the issues of ICL may be related to tuning of hyper parameters similar to the DCNN I tested. I also had to hand tune and engineer the input pre-processing for the ICL model on this dataset so that it would actually learn color sensitive image features. As learned features generally perform better than hand-crafted input features, this suggests that inefficiencies in the first layer's input representation may have created an additional bottleneck for overall performance.

More importantly, the addition of complex cell learning did not improve the performance compared to the simulations which only featured simple cell learning. This suggested that the complex cell model may not be performing its core theoretical function, namely increasing tolerance to image variability. When testing the generalization performance of the ICL model for image translation, with and without complex cells, I found that the complex cells did not improve generalization over translation. Similarly, axonal learning did not seem to contribute to better performance, which makes sense if the ICL model was largely propagating its low-level representation upwards, as it would cause the axonal simulation to stay close to its retinotopic initialization. See Sec. 6.4.5 for more discussion about complex cell issues and a proposed study to address the shortcomings of the implementations used in this project.

6.2.3.2 How does ICL compare to Early work with DCNNs?

It may seem like DCNN based models have been the state of the art champion of image recognition for a long time. But in reality, the original performances that made them seem promising were quite modest improvements over earlier approaches. For example, specialized SIFT models were actually within 10% points of early DCNNs trained on the ImageNet database for both top-1 and top-5 performances (Krizhevsky et al., 2012). These early

performance gains were indicative a new fundamental capability that DCNNs offered, namely greatly improved generalization over image variation. But it took a long time to to see the benefits of that new capability emerge, as the precursors of today's DCNNs, such as NEOCOGNITRON (Fukushima, 1980), were never really considered powerhouses of object recognition performance in their time. Getting DCNNs to their current status required multiple advancements over decades of work.

Earlier DCNNs were plagued with numerous issues. One of the main problems was training multiple layers of a DCNN successfully, as was also a probel for deep networks in general. NEOCOGNITRON (Fukushima, 1980), used a simple layerwise hebbian method, and later the much more successful Deep belief Convolutional Networks used a layer-wise Restricted Boltzmann Machine learning algorithm (Krizhevsky and Hinton, 2010). But It wasn't until the revival of deep gradient descent or backpropagation, which had largely fallen out of favor in the 2000s, that DCNNs really started to rapidly increase in performance. One pernicious issue that remained was that the training signals from the top-layer would either vanish or explode as they traveled down the network. Pre-training with stacked auto-encoding and careful scaling of weight initializations helped, but did not fully alleviate the problem. It wouldn't be till the invention of batch normalization that a very accessible solution to this problem was available for a wide array of DCNN architectures (Ioffe and Szegedy, 2015).

I would argue that models like ICL are likely in their nascent phase. The fact that they can perform well at all compared basic DCNNs, which benefit from a decades of improvements, suggests that ICL-like models are worth exploring further. Unfortunately, as I found in this work, many of the same methods that have enhanced DCNN performance over the years do not help with ICL models. For instance, something that serves the role of batch normalization is necessary to stabilize learning in deep ICL models, but standard batch normalization actually hinders their performance. Further, the design of layer-wise learn-

ing methods still lag far behind those of deep gradient descent and similar global learning methods.

At a high-level, ICL has a similar problem to DCNNs before batch normalization. Where DCNNs had trouble passing stable learning signals down to their lower layers, ICL had trouble passing progressively more useful representations upward. The problem is likely caused by the statistical environment differing significantly from the first layer to the proceeding layers, which leads to unstable or trivial learning. This may be caused by or exacerbated by problems with the temporal learning of the complex cell model (see Sec. 6.4.5). New normalization processes are likely to be necessary in order to regularize the statistical landscape between the layers appropriately, and more work is needed to address the issues regarding complex cell learning.

If steady progress is made on ICL's issues, then ICL-like models will likely improve greatly over time. As unsupervised models that can handle the data-throughput of real world stimuli, ICL-like models could one day have an edge over machine learning techniques that rely heavily on supervised training and non-local learning rules.

6.3 Overview of Theoretical Work

6.3.1 Towards a Robust Theory of Unsupervised Visual Category Learning

In DiCarlo and Cox (2007) the authors introduced the idea that when objects are varied over their viewing parameters they trace a geometric surface through pixel activation space or neural activation space. DiCarlo and Cox (2007) called these geometric surfaces *Object manifolds*. With these object manifolds as the bedrock, they proposed that object recognition in visual cortex could be viewed as a series of feed forward transformations which flatten and separate tangled object manifolds in sensory space. One of the interesting predictions of this theory/framework was that the use of broadly tuned neurons which separate object

manifolds more so at the space level rather than at the unit level could be a valid encoding scheme for high-level ventral visual cortex. Object manifolds have proved invaluable as a framework for understanding how supervised deep learning in feed-forward DCNNs can be so successful.

Despite its success, a major limitation of the object manifold framework is that it did not propose a clear explanation of how the hierarchical transformations of visual cortex could be learned without explicit labeling or training. Arguably the object manifold framework is agnostic about learning methods, but its most successful implementations to date have all been supervised learning models. Given that supervised learning likely only accounts for a very small portion of visual cortical learning, it suggests that an extension of the object manifold framework might be needed.

I set out to address this limitation by proposing an extension of the object manifold framework into called the *Temporal Relation Manifold* (TRM) framework in Chapter 2.

A central idea from the TRM framework is that human labels and categories emerge from the way objects and their features evolve over time in the environment. Objects tend to be perceived as a series of non-static images due to the viewer's motion, eye saccades, the object's motion, or other changes in the environment. As such, we are exposed to a consistent statistical signal for which minor visual changes still retain the meaning or identity of objects within our visual field. Essentially, TRM uses this tendency to redefine object manifolds not as a priori constructs, but as parts of an emergent statistical object called a Temporal Relation Manifold.

In a sensory space, a point on a Temporal Relation Manifold (which corresponds to a view of an object) is most likely to transition to neighboring points on the the temporal relation manifold (which will tend to correspond to views of the same object or category). Learning the topology of the Temporal Relation Manifold leads to category learning, because views of the same object or category will tend to map to neighboring parts of the temporal

relation manifold, while views of different categories will tend to map to highly separated points on the surface of the temporal relation manifold (even if they are close in sensory space). Further, the temporal relation manifold's structure can be learned through simple observation, which makes it directly accessible to unsupervised learning techniques. This new TRM framework can be seen as a companion to other ways of looking at the ventral stream, such as Object Manifold Untangling (DiCarlo and Cox, 2007) and I-Theory (Poggio and Anselmi, 2016).

Along with the TRM framework I proposed a novel computational theory called Windowed-Temporal Auto-Untangling (W-TAU) for how an algorithm could efficiently learn the topology of or untangle the Temporal Relation Manifold for a given environment under cortex like constraints. The ICL model was developed with the intent of implementing the W-TAU theory in an explicit biologically plausible model. I would argue that the version of ICL that I developed for this project was only a partially successful implementation of W-TAU. Nevertheless, the TRM framework and the Windowed-TAU theory are a rich set of ideas. As such, alternative implementations consistent with TRM and Windowed-TAU can easily be developed and are worth exploring further.

6.3.2 Towards a General Theory of Cortical Sensory Learning

6.3.2.1 A Unified View of Cortical Learning

In "Lightning is always seen, thunder always heard" Swindale (2000) the authors review research showing that when the inputs to V1 and A1 are surgically reversed, they do something quite interesting. A1 develops maps and selectivities that are stereo-typical of V1, and V1 develops maps and selectivities that are stereotypical of A1. Further, behavior seems qualitatively similar. This supports a fascinating conjecture, that at least for sensory cortex, there may be one relatively unmodified cortical algorithm rather than several domain specific ones.

While the existence of a unified algorithm for cortical sensory learning would be a great boon to AI and brain research, it presents several challenges:

From a modeling perspective, the best algorithms we have for performing human like sensory inference are highly tailored to their specific sensory domains. In modeling, we generally see algorithmic specialization for both the domain and the level of abstractness for input representations (specialized DCNNs for vision, specialized DCNNs for sound, Transformers for Language). While this has lead to major advancements in artificial intelligence and brain understanding, these advancements have been relatively narrow and rarely integrative across sensory domains beyond a shallow extent.

From a brain perspective, research has generally focused on studying the way individual sensory pathways learn and develop as separate phenomena rather than as different end results of the same more fundamental process. This makes sense given that the features learned by cortex for different sensory domains seem so vastly different. But if these differences are more a consequence of the different input statistics for each sensory domain, then it suggests that we should be studying cortical representation of the senses in a more unitary way. However, without strong domain independent cortical sensory models and theories, unitary study of cortical sensory representations is difficult.

6.3.2.2 Integrated Cortical Learning as a Starting Point

When broken down, ICL can be thought of as an attempt to use low-level neural mechanisms such as axonal plasticity, simple cell learning, and complex cell learning, to build a cortical architecture which is domain independent, and TRM can be viewed as a theory for what architectures like ICL are computing.

Axonal plasticity shows that the structural connectivity of cortical areas can be vastly influenced by the statistics of its inputs. In contrast, the vast majority of neural network models today assume structural connectivity is constant. The pattern of structural connectivity between layers in a model is a fundamental source of its domain adaptability and

specialization. When combined with delay line coding, axonal plasticity opens up the possibility of learning even temporal input windows.

Cortical map learning shows that feature learning across early sensory cortex may use a similar system of hierarchical and lateral, Hebbian anti-Hebbian dynamics. Such learning is extremely adept at distributing neuronal receptive fields throughout a space of possible input patterns. With the right supporting structural input connectivity, cortical map learning could be a powerful domain general form of feature learning.

Complex receptive field learning. In theory, cells that learn to tolerate minor feature variations in incoming stimulus representations, like V1 complex cells are thought to do, would be useful in all modalities. Further, the ability to learn these receptive fields could let them specialize to each modality.

While not the main goal of this dissertation, ICL serves as a first attempt to build a model that is domain general, and that can adaptively specialize itself to the domain it is exposed to. ICL opens the door to modeling multi-sensory integration within a more unified framework.

6.4 Future Directions

The work on the ICL model for this dissertation is by no means comprehensive. In effect it could not be, because ICL is an attempt to make an entirely new class of models both computationally and scientifically viable for study. As a first attempt to build an ICL model, this work featured many limitations that future work could overcome.

6.4.1 Axonal Learning

The Arbor Model used for the simulations of axonal plasticity dynamics in the ICL networks has large potentials outside of this work. It is already useful for creating connectivity in

supervised or unsupervised pre-training for stacked models. But with a minor augmentations of its design, the arbor model could allow for connectivity itself to be approximately differentiable. This would allow deep locally connected models to effectively learn their own large-scale connectivity structure as part of their objective function. This development could have wide-reaching affects on the field of Deep Learning.

6.4.2 Abstractions of ICL

As mentioned earlier, there are some significant advantages to posing an algorithm in terms of minimizing objective functions via gradient descent, even if those objective functions are local in nature. Developing a version of ICL based on local layer-wise or unit-wise objective minimization could yield a more stable and easily tuned model. The current version of ICL appeared to maintain its low-level representations through 3 additional layers of representation suggesting that better tuning may be all that a model with ICL like learning dynamics needs in order to improve performance past its first layer. Gradient based ICL might make that tuning more feasible.

6.4.3 Replicating V1 Complex Cell Work

Much simulation and theory work has been conducted to understand how simple cells and complex cells might interact with one another in V1 (see Martinez and Alonso (2003) for a review of some of these theories and other related work). But most of this work has been done using very simplified models only designed to work on simple artificial stimuli. Most of them look at small populations of simple cells and complex cells with full fixed connectivity. The ICL models used in this paper are a major advance over prior models in that they allow for simulation of a V1 model with axonal plasticity and windowed connectivity, and do so at a large scale on real world stimuli. As such, ICL could form the basis of a new generation of V1 models that seek to explain more nuanced interactions between classically

studied phenomena with more biological detail and do so at larger scale than previous models allowed.

6.4.4 New Measures of Neural Specialization

The analysis used for Simulation Study 2, namely comparing a neural representations unit deletion (UD) curve to its artificially un-specialized UD curve, could be used to better understand neural specialization in real cortex. This would be very informative, as the current understanding of unit specialization in the brain is largely based on notions of selectivity and preferred stimulus, which as seen in this dissertation's experiments do not actually imply strong functional specialization. Using the unit deletion specialization test on a sampling of neurons across IT and within functional areas like FFA, with respect to relevant categories, would give an estimate of the upper bound on unit specialization in the high-level ventral visual stream.

6.4.5 Improving Models of Complex Cell Learning

As mentioned earlier, the specific implementation of the complex cell models based on the TRM framework for this project were not as successful as hoped.

From a theoretical perspective, there are three main ways that complex cells might not contribute to generalization. First, if complex cells learn pools which are too narrow, they will effectively just propagate the simple cell representation. Second, if the complex cells learn pools which are too wide, they may lose useful discriminative information more than they eliminate non-discriminative information. Third, if the pool membership is ill-chosen (i.e. does not actually conjoin correctly related features) then the complex cells will be unlikely to contribute to better generalization. Paradoxically, in the second and third situations the next level of simple cells may still "recover" enough of the original simple-cell representation in order to propagate similar unit selectivities to higher layers. Unfortunately, diagnosing if

one of these problems is occurring in the ICL model, is beyond the scope of this dissertation. However, it is grounds for proposing a small follow-up study to examine these matters.

This dissertation focused on broadly testing if the ICL models we developed would be useful as tools for future computational cognitive neuroscience research. As such, we did not test the model on so-called *toy problem* datasets which are of decreasing interest for neuroscience or cognitive science. Models that are calibrated and perform well on toy datasets rarely perform well on difficult or real-world datasets, so we opted to build our model with more difficult datsets for calibration.

Critically though, well constructed toy-datasets are powerful tools for understanding if a model is actually correctly implementing a theory it is supposed to represent. This is because a toy dataset with simple stimuli give rise to predictable responses and representations at different levels within a model, whereas, more complex datasets with more complex stimuli rarely give rise to tidy (or actionable) predictions about representation and response characteristics.

6.4.5.1 Proposal for Complex Cell Calibration Study

I propose that several candidate models based on ICL should be trained on a simple shape dataset. To start with, This simple shape dataset could include squares, triangles, and circles. The training would include temporal sequences of these shapes being shifted across the image space, using a Gaussian random walk to simulate fixations. Then there would be two kinds of testing: low-level tolerance tests and generalization tests.

For the low-level tolerance tests, the network would be exposed to a battery of Gabor wavelets, and each units responses would be recorded. Using this this stimulus battery, we would build a tuning profile for the simple cell units and the complex cell units. If the complex cell model is working correctly, the complex cell units should be more broadly tuned than their simple cell counterparts even after lateral inhibition and competition effects have

taken effect. These tuning profiles would be calculated for every level of representation in the network.

For the generalization tests, the network would have a classification layer added as output for every layer in the network and trained on centered stimuli. Next the network would be tested on battery of shifted images in order to build generalization curves for each layer of representation within the model. If the complex cells are functioning correctly (i.e. increasing tolerance, and not eliminating discriminative information), then these generalization curves should increase in width without dropping in peak performance (see. Fig. 6.1).

6.4.5.2 Alternate Implementations of Complex Cell Learning

Gradient Descent Based Learning. One reason why it is difficult to tune the complex cell model is that biologically inspired dynamics lack explicit objectives that are guaranteed to be minimized as the network learns. Gradient descent algorithms like traditional deep learning, have a remarkable ability to tune themselves to a degree and will often tolerate poorly chosen hyperparameter values. However, complex cell learning poses a challenge to being expressed as an unsupervised objective function. Essentially, a TAU compatible complex learning object would need to minimize the expected discriminability between features that are temporally close to one another, while maximizing the discriminability of features that are only little further away temporally. Further, maxpooling like operations in general limit the amount of useful gradient information, as only a single input to the maxpooling operation can have a substantial gradient.

Critically, the addition of gradient descent dynamics is also not inherently biologically implausible if each objective is manipulating local variables in a plausible way. For instance, the Arbor Model uses gradient descent to efficiently implement known forces in axonal development with only local communication.

Adversarial Learning. It is possible that a version of the complex cell model that used adversarial learning could alleviate some of the limitations of the current complex cell model.

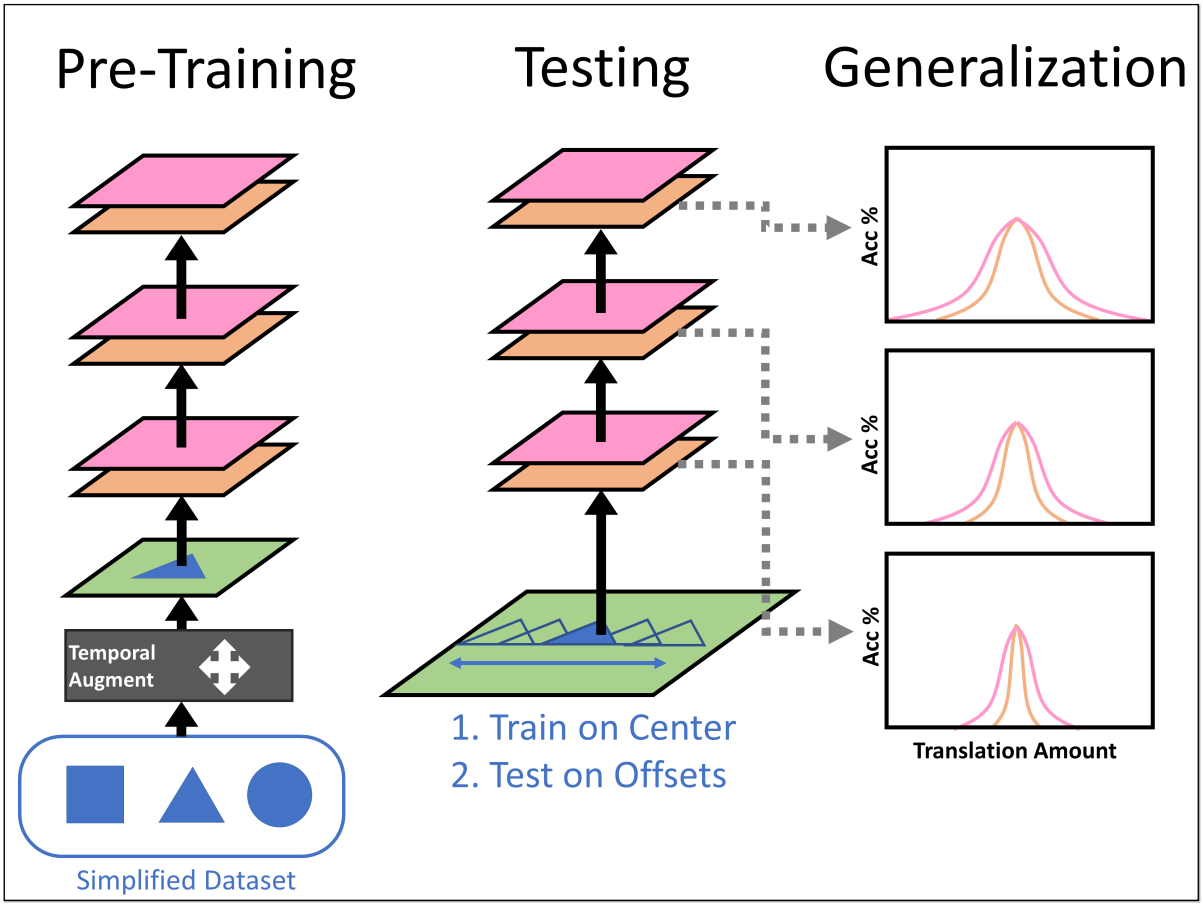

Figure 6.1. Study for calibrating sophisticated complex cell models integrated into cortical models

This figures outlines a potential Simulation Study for better understanding and calibrating the behavior of trace rule complex cell learning models that are integrated into large cortical learning models. In the pre-training phase a simplified dataset is temporally augmented using pixel shifting and presented to an ICL like model until all layers are trained. Then a test battery featuring images shifted along a single axis. The neuron which respond most strongly, will be recorded and displayed to better understand its tuning curves, for both the simple cell layers and complex cell layers in each module. If the complex cell tuning curves are wider than the simple cell curves in every layer, and the width increases in higher modules then the complex cells are likely functioning correctly at least at a low level. Discrimination performance will also be tested to see the if complex cell behavior corresponds to better generalization performance functionally. This paradigm would be useful for both studying complex cell model behavior and tuning it to achieve desired results within deep cortical models.

Essentially, one population of cells could take windowed input from a complex cell layer and act as local discriminators whose jobs detecting if the input image has actually changed locally. However, the complex cell's objective, would be to make minor perturbations of an image generated by temporal sampling look identical. It should be noted that there is no evidence for this kind of learning to my knowledge in the neuroscience literature, and it would sitting in as an abstraction for a biologically plausible form of learning.

Binary-ICL. The simple and complex cell models implemented for this project have real continuous neural activations. This makes it somewhat more difficult to tune a model to meet the various temporal dynamics and sparsity requirements of Windowed-TAU successfully. By making the the inputs to the simple cells binary and their outputs binary, the sparsity of the system would be easier to control. Further, the binary outputs of the simple cells could almost be treated as the state of a Markov process. If the windows to complex cells were altered so that only one unit within the window registered as on, each complex cell window could explicitly be treated as a simple state-machine and markov analysis could be used for fine-tuning their learning. Finally, if the output of the complex cells is binary, it would be easier to adapt the sparsity settings such that the complex cell layer produced similar activation statistics to the initial input of the simple cell layer. This would mean that stacked Binary-ICL modules would be more likely to learn progressively more sophisticated representations without stalling out, as higher layers would be exposed to similar conditions which lead to strong learning in lower layers.

The main downside to Binary-ICL is that large amounts of information would be lost from the input image to the first simple cell layer. This may make Binary-ICL less suitable for processing real-world data depending on the specific input transformation used. However, Binary-ICL may prove to be a versatile way if checking if the assumptions of an ICL-like model can successfully implement Windowed-TAU.

6.5 Main Conclusion & Impact

In this dissertation, I introduced a new platform for modeling visual cortex, that is ground breaking in a number of ways. Further, this dissertation demonstrated that the new platform offers inherently new ways for comparing theories of visual cortex to actual visual cortex via modeling. In this section, I will outline why the Integrated Cortical Learning Model is both groundbreaking as a model and critical as a first step in opening up new frontiers of cortical research.

6.5.1 Objectives Revisited

When I started this dissertation I had several main objectives. First, I wanted to introduced a high-level theory that explained how an architecture of composed of multiple modules of simple and complex cell layers could learn hierarchical representations of categories. Second, I wanted to develop implementations of these concepts, and evaluated if a model based on these concepts could be useful for studying study several cognitive neuroscience phenomena of interest, specifically, cortical map development, unit specialization, and unsupervised learning. In the chapters of the dissertation, I have largely accomplished these objectives.

In Chapter 2, I introduced the *Temporal Relation Manifold* (TRM) framework for understanding unsupervised category learning as a product of connectivity learning, simple cell-like learning and complex cell-like learning.

In Chapter 3, I introduced a new form of model that allowed deep networks to learn their connectivity in an unsupervised fashion. This new model called the Axon Game also generates cortical map like analogues. The Axon Game served as a proof of concept and a template for building the more advanced connectivity learning mechanisms employed for this dissertation known as the Arbor models.

In Chapter 4, I introduced the Arbor Model for connectivity learning, a hierarchical simple cell cortical map learning model, and a hierarchical complex cell cortical map learning

model, and several full implementations of the ideas in the TRM framework called *Integrated Cortical Learning* (ICL) models.

In Chapter 5, I tested if our implementations of the ICL models developed high-level cortical maps and specialized units, and if they learned categories in an unsupervised fashion. The results of these studies should be useful in developing better versions of the ICL model and demonstrating that ICl models have strong potential for cognitive neuroscience modeling, despite their limitations.

6.5.2 Motivation Revisited

My work started as a reaction to the realization that cortex is one of the slipperiest substances known to mankind. By this I mean that it is incredibly plastic in ways that few if any current models can mimic, and its plasticity makes detailed theoretical predictions quite difficult at the system level.

When I began this work, DCNNs had already become the dominant model in visual neuroscience. This was for good reason, as they provided an excellent platform for testing out ideas, for example whether the feed-forward object manifold view of object recognition as discussed in (Yamins and DiCarlo, 2016b), was a useful starting point for understand visual cortex. Research with DCNNs has thoroughly demonstrated that looking at visual cortex as a hierarchical series of transformations is an incredibly useful perspective to take (Khaligh-Razavi and Kriegeskorte, 2014; Khaligh-Razavi et al., 2017; Yamins and DiCarlo, 2016b). While alternative architectures, like transformers (Vaswani et al., 2017; Dosovitskiy et al., 2020), are starting to become competitive for visual tasks, its unclear if they will have the same impact on neuroscience.

Despite their success, DCNNs are actually quite inflexible in a number of important ways compared to even our most basic understanding of visual cortex. First, their features are rigidly segregated into separate feature channels, with a fixed weight-sharing assumption

that bears little resemblance to cortical constraints. Second, their pools are rigidly defined as blocks of of neighboring units within their maps, whereas their counterpart complex cells may learn their pools in a sophisticated manner. Finally, the underlying structural connections between layers are fixed in DCNNs, whereas axonal development in cortex can be phenomenally plastic.

If, DCNNs can be thought of as learning to drive a car, then visual cortex is like learning to drive, while simultaneously building the car and paving the road.

As a result of these differences, there are a number of physiological and computational comparisons between cortex and DCNNs that simple can't be made. For instance, DCNNs do not have an analogue to cortical map development, despite their study being one of the fundamental methods for assessing neural representations in the brain. Alongside this, the differences between DCNNs and cortex make certain comparisons difficult to interpret.

To address these issues, computational neuroscience needs new modeling platforms that can both incorporate ignored but fundamental kinds of cortical plasticity, and that can allow for more detailed physiological comparisons. But developing these new platforms will require experimentation, risk taking, and substantial engineering development. Further, new theories and frameworks are needed to understand how these new models can learn visual object categories without supervision. This dissertation is an important first step in that direction.

6.5.3 Engineering Plasticity

One of the central challenges of developing new architectures that better reflect the plasticity of actual cortex, is that real cortex has surprisingly few fundamental representational constants. For example, visual cortex may use the same fundamental rules as auditory cortex, but they look vastly different at a fundamental feature level. Yes, V1 and A1 have four laminae and later cortical sensory areas have six, and the wiring is very similar from area

to area, but representationally and structurally there is so much that changes depending on the nature of the inputs and due to experience. As section 6.3.2 highlights, this level of structural and representational flexibility may be critical to understanding cortical sensory function more deeply.

6.5.3.1 Introducing Axons and Structural Plasticity

Many visual models make strong assumptions about the neighborhood relationship between pixels in incoming images. For instance, convolutional models assume that input images are arranged in pixels in a strict grid formation. In contrast, deep fully connected networks and random connected networks, which were popular in the past for visual tasks, essentially assume that there is no neighborly relationship between the inputs. But a convenient aspect of both DCNNs and deep fully connected or random sparse networks is that they are straightforward to implement and they use a small number of operations, which are well optimized for today's hardware and software.

Visual cortex, appears to learn a structural connectivity to match the topological structures and relationships within its inputs, whatever they may be. This is somewhat like learning a convolutional structure with only a hint about how images are organized in the first place, from just looking at jumbles of pixels over time. Further, implementing a model with this kind of flexibility and speed close to a DCNN, is non-trivial. Unfortunately, that speed is a practical necessity for widespread adoption. It took me three wholly separate iterations and projects built around this concept before I developed a technique which was fast enough to make training times truly practical.

The axon game and its progeny the arbor model are the results of many years of work to develop models which can support cortex like structural plasticity with DCNN like efficiency. Even on their own, these models make an important contributions to library of neural network components and may yet provide excellent models for cortical neuroscience.

But farther reaching, these models are a foundational step towards building architectures with the kind of underlying flexibility seen in real sensory cortex.

6.5.3.2 Bringing Cortical Map Modeling in to the Present & Beyond V1

Simple cells allow cortex to learn sets of largely uniform features across the visual field and across spatial frequencies, with very little prior assumption. This is an impressive feat, given that DCNNs need many constraints and assumptions to accomplish the same thing (i.e. convolutional weight-sharing and explicit separation of input and feature channels).

Models such as Malsburg (von der Malsburg, 1973), LISSOM, (Sirosh and Miikkulainen, 1994), and GCAL (Stevens et al., 2013), were pivotal to advancing our understanding of how features and cortical maps are learned via simple cells in early visual cortex. In theory, these models should be a good starting point for understanding how hierarchical cortical learning effectively distributes feature representation throughout its layers. However, most models of cortical map learning were implemented with programming environments and with assumptions that make them incompatible with the demands of today's deep neural models. Deep learning of any variety is incredibly computationally demanding and requires high-throughput and efficient processing to be practical. Further, DCNNs have permanently raised the bar for cognitive neuroscience modeling, requiring that new models are at least comparable in terms of data throughput and the ability to learn on real world data. These cortical map models were never designed with this in mind, and this has limited their use as deep neural models.

With the codebase for this project, I have demonstrated and outlined how to bring the ideas from these simple cell cortical map models into the modern era of deep learning, using modern environments like tensorflow and GPU computing. Further, I demonstrated that variations of these cortical map models can work well on large real world images, and with simplified dynamics that are compatible with modern necessities like batch learning.

My work shows that cognitive models built on cortical map learning architectures are viable, and that increasing performance is likely just a matter of experimenting with different learning rules and hyper-parameter settings. With this work, I provide a platform for building the first generation of hierarchical cortical map models of cognitive function, much as NEOCOGNITORON (Fukushima, 1980) provided a conceptual platform for the powerful DCNNs of today.

6.5.3.3 Complex cells *In Situ*

Complex cells learn to pool semantically related and identical features together, in order to build more tolerant representations over multiple cortical layers. This requires learning the structure of the visual world and its myriad transformations, only by visual experience. In theory, trace learning could allow complex cells to do this using temporal observations.

Currently, there are many models of complex cell learning, such as trace rule models (Földiák, 1991; Rolls and Stringer, 2001; Wallis, 1996; Einhäuser et al., 2002; Michler et al., 2009). Complex learning offers a kind of plasticity that could greatly contribute to useful perception of categories and events. While, DCNNs are only able to use their pooling neurons (complex cell stand-ins) to build incremental tolerance to image shifts, learned complex cells could offer far more general forms of incremental tolerance, while also catering to the specific variabilities of unique categories and objects.

Unfortunately, what most complex cell models have in common, is that they are generally small scale demonstrations meant to isolate or highlight complex cell functions. This means that most of them do not work on real world data and do not scale to larger populations of neurons. It also means that most of these models were never meant to be situated between other learned layers of neurons, as would be required for deep cortical learning.

With this dissertation, I demonstrate that trace rule based complex cell learning methods can be implemented at scale and that they can learn stable pool structures with relatively

simple learning rules, within deep hierarchical models. Further, this work demonstrates that these temporal models are very compatible with batch learning and can be easily be incorporated into cortical map learning frameworks.

However this work also demonstrates that complex cell learning is not a fully solved problem from a functional and computational perspective. Though not discussed in this work, I implemented and tested many different versions of complex cell learning (trace based and others) during the course of this project. All of them demonstrated clear shortcomings for the purpose of building progressively more sophisticated hierarchical representations within a deep framework. This work has started the process of exploring which combinations of complex cell models and learning rules can contribute successfully to hierarchical learning.

Encouragingly, the ability of the ICL models built for this project to maintain high-levels of performance over multiple layers with only minor degradation, suggests that existing complex cell learning rules may only need modest revisions before they can demonstrate real performance improvements. Critically, the ICL architecture and its variations developed for this project offer the necessary platform for conducting these investigations, and for building the next generation of complex cell models to support deep cortical learning.

6.5.4 New Horizons for Cortical Comparisons

One of the major contribution of the ICL model, is in how it opens the door to radically different kinds of neural comparisons to visual cortex.

The 2010s were dominated by abstract comparisons to neural network models, such as representational dissimilarity matrices (Khaligh-Razavi and Kriegeskorte, 2014; Khaligh-Razavi et al., 2017; Yamins and DiCarlo, 2016b). But, with the help of the technologies and methods developed for ICL, the next decade may start to see more direct physiological comparisons between models and visual cortex. With more direct comparisons researchers will be able to study far more fine grained questions about neural function, and future modelers will receive far more useful information for refining future models in the process.

6.5.4.1 Cortical Maps

Up until recently (Lee et al., 2020), deep neural networks predicted very little about the actual physiological structure and layout of features on the cortical surface. This is a problem because studying the structure of feature selectivity across the cortical surface has been a major contributor to our understanding of how cortex organizes and compartmentalizes information. From low-level ocular dominance columns and orientation columns (LeVay et al., 1975; Hubel et al., 1978), to high-level functional areas such as FFA (Kanwisher et al., 1997; Sergent et al., 1992), the topological grouping of neural selectivity has inspired many theories and debates about cortical function (von der Malsburg, 1973; Sirosh and Miikkulainen, 1994; Grill-Spector et al., 2017; Wandell et al., 2007; Weiner and Zilles, 2016).

Models that build their hierarchical representations on a homolog to the cortical surface, using plausible cortical interactions, could give us important insight into what cortical maps and functional groupings actually mean from a computational perspective. But a pre-requisite for building such models is the development of deep cortical map learning architectures and the technologies needed to support them. This dissertation demonstrates that deep cortical map architectures are viable with today's technology. Further, ICL can provide a platform for building these models and training them on real world ecologically valid training data.

6.5.4.2 Unit Level Representations

Right now, it is of great interest to link the development of high-level unit representations for DCNNs to high-level unit representations found in IT cortex. However, there are some reasons to be skeptical of these comparisons. First, the top-level representations of state-of-the-art DCNNs are typically fully connected. Second, their learning schemes are completely divorced from many of the local lateral interactions that dominate cortical learning. Third, without much more severe algorithmic constraints, there is very little reason to assume that

top-down learning for categories or bottom up generative learning will cause the representations of high-level DCNNs units to match their cortical counterparts. That being said, this training should cause substantial similarities to emerge at the space-level of representation between DCNNs and IT cortex, which has indeed been found (Khaligh-Razavi and Kriegeskorte, 2014; Khaligh-Razavi et al., 2017; Hong et al., 2016; Yamins and DiCarlo, 2016b).

But in order to go beyond these abstract comparisons, we need models of deep cortical learning that are not built on rote supervised learning, and whose learning rules and constraints better reflect the local nature of cortical learning.

This dissertation demonstrates that models built on unsupervised learning and around local cortical constraints can learn deep neural representations. While I would argue that the high-level representations this project's ICL implementations could be improved, ICL provides a platform for testing and building new learning rules and constraints with a more cortically plausible foundation. With ICL and the future models it enables, we can begin the work of probing the differences between high-level neural representations between our models and IT cortex more deeply.

It may very well turn out that more abstract models, like DCNNs, are actually good predictors of unit-level representations in high-level cortex, or it could turn out that more traditional notions of highly specialized neurons will win out. But, it is also possible that new models enabled by the ICL will point the way to as-yet unimagined styles of representation hidden within the neurons of high-level visual cortex.

6.5.4.3 Computational Performance

Since the introduction of DCNNs capable of exhibiting human level classification performance on certain datasets, new process models of cognitive phenomena are often expected to exhibit human level classification performance on real world datasets as well. Many models can

predict human recognition performance from various viewing angles with only five or six parameters, but exceedingly few models can actually perform object recognition at anywhere near human levels, regardless of the number of free parameters. Thus testing with messy real-world datasets can be an important tool for highlighting promising architectures and learning rules in an increasingly crowded landscape.

But real-world data testing is a double edged sword, as the reality of model development is often messy, long, and nonlinear. Often there is a significant exploratory engineering period that can last years before new architectures and approaches show impressive results on real world data. Highlighting architectures because they are conceptually and theoretically interesting, even in the absence of strong real world performance, is still an important part of moving our understanding of visual cognition forward.

Further, it is quite difficult to predict which models will eventually become dominant and which models will fall by the wayside, even with good "real world" performance estimates. Essentially, this sets research communities up for a strong bias favoring currently computationally successful models. For example, in the first decade of the 21st century, SIFT features (Lowe, 1999; Geng and Jiang, 2009; Bicego et al., 2006) and HOG filters (Dalal and Triggs, 2005; Wang et al., 2009; Pang et al., 2011) were at the top of performance charts, while at the same time "Convolutional Neural Networks" and "neural network" approaches in general were not taken seriously by many experts in the field of human visual processing and computer vision. Yet, SIFT and HOG filters have largely fallen out of favor and deep neural networks approaches are now favored as potential models of human visual processing.

By 2020 state-of-the-art standards, ICL is not yet a highly competitive class of models. But ICL already works as well or close to as well as basic DCNNs on real-world images. As such, ICL shows strong promise that it can improve substantially with more development and iteration. Further, it clearly demonstrates that architectures based on more fundamental cortical principles can handle the data and speed requirements needed in order to train highly competitive models.

Real-world classification performance of more biologically realistic models of cortical map development and axonal development is rarely tested. On the other hand, DCNNs are very abstract implementations of the hierarchical theory of cortex, but show excellent performance on object recognition and visual understanding tasks. With this study I have shown that the ICL model can perform roughly at the same order of magnitude as DCNNs for image recognition, while providing a more biologically detailed implementation of current theories of cortex. This is a critical niche of models that needs more attention in the field of computational cognitive neuroscience.

At this point, it would make sense to compare the ICL architecture to other highly similar architectures from a computational perspective. However, there is to my knowledge, no precedent for components like those used in ICL being combined together in this manner. At an abstract level, ICL is a member of the *deep locally connected neural network* DLCNN class of models, but it shares very little in common with most DLCNNs. Most DLCNNs are trained using supervised gradient descent and do not feature connectivity learning or specific layers of neurons that serve as simple and complex cell surrogates. On the other hand, ICL is conceptually similar to DCNNs, but DCNNs are so mechanistically different that very few principles of DCNNs appear to apply to ICL models. This makes the challenge of developing and tuning the ICL architecture a unique one, as there are few models which can be used as guides for making informed development choices or for direct comparison.

Given historical context, the special niche it fills, and ICL's unique challenges, I would argue that for its first demonstration in cognitive modeling, ICL shows strong potential. But evaluating that potential further is going to require more work, more engineering, and more studies. However, this effort is well offset by the prospect that models like ICL can move us towards an understanding of the brain that more tightly anchors visual cognition to the underpinnings of visual neuroscience.

6.5.5 Addressing the Paradox of Unsupervised Category Learning

The DiCarlo and Cox (2007) object manifold framework has been invaluable for understanding how feed forward deep neural networks can decode object categories from visual images. But how the brain transforms images into perceptions of categories is only one part of the puzzle, the other part of the puzzle is how the brain learns these transformations in the first place.

Supervised deep gradient descent has provided many poofs of concept over the past decade that the layer-wise windowed structure of DCNNs and other similar models are capable of encoding successful transformations from visual images to categorical labels. But, very few scientists consider supervised deep gradient descent as a plausible explanation for the majority of visual learning. Firstly, deep gradient descent is largely considered neurally implausible (Yamins and DiCarlo, 2016b). Second, and more importantly, the amount of supervision (i.e. labeled examples) required for the most successful of these models is excessive and appears to be orders of magnitude larger than what animals or humans need in order to successfully learn visual categories. Importantly, unsupervised methods of deep learning are improving consistently, and more efficient forms of deep gradient descent that need fewer and fewer examples are constantly being developed. But it remains to be seen whether this trajectory will ever reach the level of unsupervised category learning seen in humans or whether it will ever provide a good explanation for visual category learning in humans.

With the temporal relation manifold framework, I have introduced an alternate approach to tackling the problem of unsupervised visual category learning. This new approach gives tacit acknowledgement to a conundrum that previous approaches have largely ignored. Namely, the system of categories learned by the human brain is also largely created by the human brain. As such, the categories themselves need to be viewed as an emergent phenom-

ena, one that the human brain is simultaneously learning and creating through its particular method of perception.

The categories that define Object Manifolds as separate entities must be defined external to the system learning them, essentially they are a-priori assumptions held to be self-evident. This makes the development of unsupervised learning methods difficult, because it inevitably involves looking for an objective or goal for learning, that just happens to tease events apart in the right way to support human-like categories, often without a strong theory of what informs the structure of those categories.

On the other hand, *Temporal Relation Manifolds* (TRMs) do not need external definition, TRMs are tangible observable statistical objects with measurable topology and structure. Further, the way that a temporal relation manifold will be untangled will be specific to the particular method of untangling, and can be engineered to emphasize particular kinds of distinctions. In other words, a theory for untangling TRMs is both a theory of learning and a theory of category genesis that can make specific qualitative predictions when implemented by models.

As an important qualification, the structures built by unfolding TRMs might be best described as a proto-categorical space that supports category-like discriminations, rather than an explicit system of categories like the ones we verbalize. However, this proto-categorical space would vastly improve learning for socially generated categories, as individuals would already tend to share roughly similar ways of structuring their environment.

With the right scope of application, the TRM framework can provide a valuable new viewpoint for designing and understanding visual learning systems as both category learners and as an integral part of the category creation process.

6.5.6 Parting Words

In conclusion, I developed a framework and theory of unsupervised visual learning, and using that theory I built a novel model of deep cortical learning designed to offer an im-

proved platform for studying deep cortical research in terms of bringing models closer to the neuroscience of visual cortex. With this new model, I demonstrated that deep cortical map models with learned connectivity and complex cell pools are viable with today's software and hardware. Further, I demonstrated that this platform offers new capabilities for comparing our theories of cortical learning to the physiological organization of information in actual visual cortex. As an example, ICL enabled the investigation of cortical map representation and unit level specialization in ways not offered by traditional DCNNs.

This project has been remarkably successful as an introduction for a new platform to support cognitive neuroscience modeling. But this work has also demonstrated areas of cortical modeling research that need more theoretical and modeling development. For instance, there is ample room for future work regarding the development of more appropriate complex cell learning rules for deep cortical learning. This dissertation has also raised concerns that researchers should be even more skeptical about interpreting the presence of selectivity and preferred stimuli to mean that high level neurons are at all functionally specialized.

I look forward to seeing how future models inspired by the ICL platform and the TRM framework of unsupervised learning, can improve upon the results of this dissertation, and to seeing how they can expand the borders of visual cortical research beyond what was previously thought possible.

REFERENCES

Anderson, J. A. and Mozer, C. M. (1981). Categorization and Selective Neurons. In Hinton, G. and Anderson, J. A., editors, *Parallel Models of Associative Memory*, chapter 8. Erlbaum.

Anderson, M., Motta, R., and Stokes, S. C. M. (1996). A Standard Default Color Space for the Internet - sRGB.

Bednar, J. A. and Wilson, S. P. (2016). Cortical Maps. *Neuroscientist*, 22(6):604–617.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems*, pages 153–160.

Benson, D. L., Colman, D. R., and Huntley, G. W. (2001). Molecules, maps and synapse specificity. *Nature Reviews Neuroscience*, 2(12):899–909.

Bicego, M., Lagorio, A., Grosso, E., and Tistarelli, M. (2006). On the use of SIFT features for face authentication. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

Bishop, K. M., Garel, S., Nakagawa, Y., Rubenstein, J. L. R., and O'Leary, D. D. M. (2003). Emx1 and Emx2 cooperate to regulate cortical size, lamination, neuronal differentiation, development of cortical efferents, and thalamocortical pathfinding. *Journal of Comparative Neurology*, 457(4):345–360.

Bishop, K. M., Goudreau, G., and O'Leary, D. D. M. (2000). Regulation of Area Identity in the Mammalian Neocortex by Emx2 and Pax6. *Science*, 288(5464):344–349.

Bishop, K. M., Rubenstein, J. L. R., and O'Leary, D. D. M. (2002). Distinct Actions of Emx1, Emx2, andPax6 in Regulating the Specification of Areas in the Developing Neocortex. *Journal of Neuroscience*, 22(17):7627–7638.

Blasdel, G. G. (1992). Orientation selectivity, preference, and continuity in monkey striate cortex. *Journal of Neuroscience*, 12(8):3139–3161.

Bonhoeffer, T. and Grinvald, A. (1991). Iso-orientation domains in cat visual cortex are arranged in pinwheel-like patterns. *Nature*, 353(6343):429–431.

Bosking, W. H., Zhang, Y., Schofield, B., and Fitzpatrick, D. (1997). Orientation Selectivity and the Arrangement of Horizontal Connections in Tree Shrew Striate Cortex. *Journal of Neuroscience*, 17(6):2112–2127.

Burt, P. and Adelson, E. (1983). The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540.

Carreira-Perpiñán, M. Á. and Goodhill, G. J. (2002). Are visual cortex maps optimized for coverage? *Neural Computation*, 14(7):1545–1560.

Chapman, B., Jacobson, M. D., Reiter, H. O., and Stryker, M. P. (1986). Ocular dominance shift in kitten visual cortex caused by imbalance in retinal electrical activity. *Nature*, 324(6093):154–156.

Chklovskii, D. B. (2000). Optimal Sizes of Dendritic and Axonal Arbors in a Topographic Projection. *Journal of Neurophysiology*, 83(4):2113–2119.

Crair, M. C., Gillespie, D. C., and Stryker, M. P. (1998). The Role of Visual Experience in the Development of Columns in Cat Visual Cortex. *Science*, 279(5350):566–570.

Crowley, J. C. and Katz, L. C. (2000). Early Development of Ocular Dominance Columns. *Science*, 290(5495):1321–1324.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, volume I, pages 886–893.

De Paola, V., Holtmaat, A., Knott, G., Song, S., Wilbrecht, L., Caroni, P., and Svoboda, K. (2006). Cell Type-Specific Structural Plasticity of Axonal Branches and Boutons in the Adult Neocortex. *Neuron*, 49(6):861–875.

DiCarlo, J. J. and Cox, D. D. (2007). Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8):333–341.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. `https://arxiv.org/abs/2010.11929v2`.

Dougherty, R. F., Koch, V. M., Brewer, A. A., Fischer, B., Modersitzki, J., and Wandell, B. A. (2003). Visual field representations and locations of visual areas v1/2/3 in human visual cortex. *Journal of Vision*, 3(10):586–598.

Durbin, R. and Mitchison, G. (1990). A dimension reduction framework for understanding cortical maps. *Nature*, 343(6259):644–647.

Einhäuser, W., Hipp, J., Eggert, J., Körner, E., and König, P. (2005). Learning viewpoint invariant object representations using a temporal coherence principle. *Biological Cybernetics*, 93(1):79–90.

Einhäuser, W., Kayser, C., König, P., and Körding, K. P. (2002). Learning the invariance properties of complex cells from their responses to natural stimuli. *European Journal of Neuroscience*, 15(3):475–486.

Elston, G. N. and Rosa, M. G. (1997). The occipitoparietal pathway of the macaque monkey: Comparison of pyramidal cell morphology in layer III of functionally related cortical visual areas. *Cerebral Cortex*, 7(5):432–452.

Engel, S. A., Glover, G. H., and Wandell, B. A. (1997). Retinotopic organization in human visual cortex and the spatial precision of functional MRI. *Cerebral Cortex*, 7(2):181–192.

Farah, M. J. (2004). *Visual agnosia*. MIT press.

Földiák, P. (1991). Learning Invariance from Transformation Sequences. *Neural Computation*, 3(2):194–200.

Foldiak, P. (2003). Sparse Coding in the Primate Cortex. In Arbib, M. A., editor, *THe Handbook of Brain Theory and Neural Networks*, chapter Sparse Cod. MIT PRESS, Cambridge, MA, second edi edition.

Fraser, S. E. and Perkel, D. H. (1990). Competitive and positional cues in the patterning of nerve connections. *Journal of Neurobiology*, 21(1):51–72.

Fukuchi-Shimogori, T. and Grove, E. A. (2001). Neocortex Patterning by the Secreted Signaling Molecule FGF8. *Science*, 294(5544):1071–1074.

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

Gebhardt, C., Bastmeyer, M., and Weth, F. (2012). Balancing of ephrin/Eph forward and reverse signaling as the driving force of adaptive topographic mapping. *Development*, 139(2):335–345.

Geng, C. and Jiang, X. (2009). Face recognition using sift features. *Proceedings - International Conference on Image Processing, ICIP*, pages 3313–3316.

Gierer, A. (1987). Directional cues for growing axons forming the retinotectal projection. *Development*, 101(3):479–489.

Gierer, A. and Lewis, W. (1983). Model for the retino-tectal projection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 218(1210):77–93.

Godfrey, K. B., Eglen, S. J., and Swindale, N. V. (2009). A multi-component model of the developing retinocollicular pathway incorporating axonal and synaptic growth. *PLoS Computational Biology*, 5(12).

Gogolla, N., Galimberti, I., and Caroni, P. (2007). Structural plasticity of axon terminals in the adult.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT press.

Goodhill, G. J. and Cimponeriu, A. (2000). Analysis of the elastic net model applied to the formation of ocular dominance and orientation columns. *Network: Computation in Neural Systems*, 11(2):153–168.

Grill-Spector, K. and Malach, R. (2004). the Human Visual Cortex. *Annual Review of Neuroscience*, 27(1):649–677.

Grill-Spector, K. and Weiner, K. S. (2014). The functional architecture of the ventral temporal cortex and its role in categorization. *Nature Reviews Neuroscience*, 15(8):536–548.

Grill-Spector, K., Weiner, K. S., Kay, K., and Gomez, J. (2017). The Functional Neuroanatomy of Human Face Perception. *Annual Review of Vision Science*, 3(1):167–196.

Hamasaki, T., Leingärtner, A., Ringstedt, T., and O'Leary, D. D. M. (2004). EMX2 Regulates Sizes and Positioning of the Primary Sensory and Motor Areas in Neocortex by Direct Specification of Cortical Progenitors. *Neuron*, 43(3):359–372.

Hashimoto, T. B., Sun, Y., and Jaakkola, T. S. (2015). From random walks to distances on unweighted graphs.

Hong, H., Yamins, D. L. K., Majaj, N. J., and Dicarlo, J. J. (2016). Explicit information for category-orthogonal object properties increases along the ventral stream. 19(4).

Horton, J. C. and Hocking, D. R. (1996). Intrinsic variability of ocular dominance column periodicity in normal macaque monkeys. *Journal of Neuroscience*, 16(22):7228–7339.

Howard, J. (2019). *Imagenette: a smaller subset of 10 easily classified classes from Imagenet, and a little more French*. `https://github.com/fastai/imagenette`.

Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1):106–154.

Hubel, D. H. and Wiesel, T. N. (1965). Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *Journal of neurophysiology*, 28:229–289.

Hubel, D. H. and Wiesel, T. N. (1972). Laminar and columnar distribution of geniculocortical fibers in the macaque monkey. *Journal of Comparative Neurology*, 146(4):421–450.

Hubel, D. H. and Wiesel, T. N. (1974). Sequence regularity and geometry of orientation columns in the monkey striate cortex. *The Journal of Comparative Neurology*, 158(3):267–293.

Hubel, D. H., Wiesel, T. N., and Stryker, M. P. (1978). Anatomical demonstration of orientation columns in macaque monkey. *Journal of Comparative Neurology*, 177(3):361–379.

Humphrey, A. L., Sur, M., Uhlrich, D. J., and Sherman, S. M. (1985). Projection patterns of individual X- and Y-cell axons from the lateral geniculate nucleus to cortical area 17 in the cat. *Journal of Comparative Neurology*, 233(2):159–189.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). Adversarial Examples Are Not Bugs, They Are Features. *Distill*, 4(8).

Innocenti, G. M. and Price, D. J. (2005). Exuberance in the development of cortical networks. *Nature Reviews Neuroscience*, 6(12):955–965.

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *32nd International Conference on Machine Learning, ICML 2015*, 1:448–456.

Issa, N. P., Rosenberg, A., and Husson, T. R. (2008). Models and measurements of functional maps in V1. *Journal of Neurophysiology*, 99(6):2745–2754.

Issa, N. P., Trepel, C., and Stryker, M. P. (2000). Spatial Frequency Maps in Cat Visual Cortex. *The Journal of Neuroscience*, 20(22):8504 LP – 8514.

Jamann, N., Jordan, M., and Engelhardt, M. (2018). Activity-dependent axonal plasticity in sensory systems. *Neuroscience*, 368:268–282.

Jo, J. and Bengio, Y. (2017). Measuring the tendency of CNNs to Learn Surface Statistical Regularities.

Kanwisher, N., McDermott, J., and Chun, M. M. (1997). The Fusiform Face Area: A Module in Human Extrastriate Cortex Specialized for Face Perception. *Journal of Neuroscience*, 17(11):4302–4311.

Katz, L. C. and Shatz, C. J. (1996). Synaptic activity and the construction of cortical circuits.

Khaligh-Razavi, S. M., Henriksson, L., Kay, K., and Kriegeskorte, N. (2017). Fixed versus mixed RSA: Explaining visual representations by fixed and mixed feature sets from shallow and deep computational models. *Journal of Mathematical Psychology*, 76:184–197.

Khaligh-Razavi, S. M. and Kriegeskorte, N. (2014). Deep supervised, but not unsupervised, models may explain IT cortical representation. *PLoS Computational Biology*, 10(11).

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69.

Krizhevsky, A. and Hinton, G. (2010). Convolutional deep belief networks on cifar-10. *Unpublished manuscript*, 40(7):1–9.

Krizhevsky, A., Sutskever, I., and Geoffrey E., H. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems 25 (NIPS2012)*, pages 1–9.

Le Vay, S., Wiesel, T. N., and Hubel, D. H. (1980). The development of ocular dominance columns in normal and visually deprived monkeys. *Journal of Comparative Neurology*, 191(1):1–51.

Lee, H., Margalit, E., Jozwik, K. M., Cohen, M. A., Kanwisher, N., Yamins, D. L. K., and DiCarlo, J. J. (2020). Topographic deep artificial neural networks reproduce the hallmarks of the primate inferior temporal cortex face processing network. *bioRxiv*, page 2020.07.09.185116.

LeVay, S., Hubel, D. H., and Wiesel, T. N. (1975). The pattern of ocular dominance columns in macaque visual cortex revealed by a reduced silver stain. *Journal of Comparative Neurology*, 159(4):559–575.

Levy, M., Lu, Z., Dion, G., and Kara, P. (2014). The shape of dendritic arbors in different functional domains of the cortical orientation map. *Journal of Neuroscience*, 34(9):3231–3236.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 2([8):1150–1157.

Marik, S. A., Yamahachi, H., McManus, J. N. J., Szabo, G., and Gilbert, C. D. (2010). Axonal Dynamics of Excitatory and Inhibitory Neurons in Somatosensory Cortex. *PLOS Biology*, 8(6):1–16.

Martinez, L. M. and Alonso, J. M. (2003). Complex receptive fields in primary visual cortex. *Neuroscientist*, 9(5):317–331.

McLaughlin, T. and O'Leary, D. D. (2005). Molecular Gradients and Development of Retinotopic Maps. *Annual Review of Neuroscience*, 28(1):327–355.

Meyer, M. P. and Smith, S. J. (2006). Evidence from in vivo imaging that synaptogenesis guides the growth and branching of axonal arbors by two distinct mechanisms. *Journal of Neuroscience*, 26(13):3604–3614.

Michler, F., Eckhorn, R., and Wachtler, T. (2009). Using spatiotemporal correlations to learn topographic maps for invariant object recognition. *Journal of Neurophysiology*, 102(2):953–964.

Miikkulainen, R., Bednar, J. A., Choe, Y., and Sirosh, J. (2005). Computational Maps in the Visual Cortex.

Obermayer, K., Ritter, H., and Schulten, K. (1990). A principle for the formation of the spatial structure of cortical feature maps. *Proceedings of the National Academy of Sciences of the United States of America*, 87(21):8345–8349.

Pang, Y., Yuan, Y., Li, X., and Pan, J. (2011). Efficient HOG human detection. *Signal Processing*, 91(4):773–781.

Parde, C. J., Colón, Y. I., Hill, M. Q., Castillo, C. D., Dhar, P., and O'Toole, A. J. (2020). Single Unit Status in Deep Convolutional Neural Network Codes for Face Identification: Sparseness Redefined. `http://arxiv.org/abs/2002.06274`.

Parde, C. J., Colón, Y. I., Hill, M. Q., Castillo, C. D., Dhar, P., and O'Toole, A. J. (2021). Closing the gap between single-unit and neural population codes: Insights from deep learning in face recognition. *Journal of Vision*, 21(8):15–15.

Poggio, T. and Anselmi, F. (2016). *Visual cortex and deep networks: learning invariant representations*. MIT Press.

Portera-Cailliau, C., Weimer, R. M., De Paola, V., Caroni, P., and Svoboda, K. (2005). Diverse modes of axon elaboration in the developing neocortex. *PLoS Biology*, 3(8).

Price, D. J., Kennedy, H., Dehay, C., Zhou, L., Mercier, M., Jossin, Y., Goffinet, A. M., Tissir, F., Blakey, D., and Molnár, Z. (2006). The development of cortical connections. *European Journal of Neuroscience*, 23(4):910–920.

Qiao, Q., Ma, L., Li, W., Tsai, J.-W., Yang, G., and Gan, W.-B. (2016). Long-term stability of axonal boutons in the mouse barrel cortex. *Developmental Neurobiology*, 76(3):252–261.

Qiu, A., Rosenau, B. J., Greenberg, A. S., Hurdal, M. K., Barta, P., Yantis, S., and Miller, M. I. (2006). Estimating linear cortical magnification in human primary visual cortex via dynamic programming. *NeuroImage*, 31(1):125–138.

Quinlan, P. T. (1998). Structural change and development in real and artificial neural networks. *Neural Networks*, 11(4):577–599.

Rajalingham, R., Issa, E. B., Bashivan, P., Kar, K., Schmidt, K., and DiCarlo, J. J. (2018). 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.

Ribot, J., Aushana, Y., Bui-Quoc, E., and Milleret, C. (2013). Organization and origin of spatial frequency maps in cat visual cortex. *Journal of Neuroscience*, 33(33):13326–13343.

Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–1025.

Rolls, E. T. and Milward, T. (2000). A model of invariant object recognition in the visual system: Learning rules, activation functions, lateral inhibition, and information-based performance measures. *Neural Computation*, 12(11):2547–2572.

Rolls, E. T. and Stringer, S. M. (2001). Invariant object recognition in the visual system with error correction and temporal difference learning. *Network: Computation in Neural Systems*, 12(2):111–129.

Ruthazer, E. S., Akerman, C. J., and Cline, H. T. (2003). Control of axon branch dynamics by correlated activity in vivo. *Science*, 301(5629):66–70.

Ryland, J. (2021). Modeling Axonal Plasticity in Artificial Neural Networks. *Neural Processing Letters*, pages 1–28.

Scellier, B. and Bengio, Y. (2017). Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11:1–13.

Schiltz, C., Sorger, B., Caldara, R., Ahmed, F., Mayer, E., Goebel, R., and Rossion, B. (2005). Impaired Face Discrimination in Acquired Prosopagnosia Is Associated with Abnormal Response to Individual Faces in the Right Middle Fusiform Gyrus. *Cerebral Cortex*, 16(4):574–586.

Sergent, J., Ohta, S., and Macdonald, B. (1992). Functional neuroanatomy of face and object processing: A positron emission tomography study. *Brain*, 115(1):15–36.

Simon, D. K. and Leary, D. D. (1992). Development of topographic order in the mammalian retinocollicular projection. *Journal of Neuroscience*, 12(4):1212–1232.

Simon, D. K. and O'Leary, D. D. M. (1992). Responses of retinal axons in vivo and in vitro to position-encoding molecules in the embryonic superior colliculus. *Neuron*, 9(5):977–989.

Simpson, H. D. and Goodhill, G. J. (2011). A simple model can unify a broad range of phenomena in retinotectal map development. *Biological Cybernetics*, 104(1-2):9–29.

Sirosh, J. and Miikkulainen, R. (1994). Cooperative self-organization of afferent and lateral connections in cortical maps. *Biological Cybernetics*, 71(1):65–78.

Sperry, R. W. (1963). Chemoaffinity in the orderly growth of nerve fiber patterns and connections. *Proceedings of the National Academy of Sciences*, 50(4):703–710.

Srivastava, N., Hinton, G., Krizhevsky, A., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Technical report.

Stevens, J. L. R., Law, J. S., Antolík, J., and Bednar, J. A. (2013). Mechanisms for stable, robust, and adaptive development of orientation maps in the primary visual cortex. *Journal of Neuroscience*, 33(40):15747–15766.

Swindale, N. V. (1996). The development of topography in the visual cortex: A review of models. *Network: Computation in Neural Systems*, 7(2):161–247.

Swindale, N. V. (2000). Brain development: Lightning is always seen, thunder always heard. *Current Biology*, 10(15):569–571.

Swindale, N. V. (2004). How different feature spaces may be represented in cortical maps. *Network (Bristol, England)*, 15(4):217–42.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014a). Intriguing properties of neural networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, pages 1–10.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014b). Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR.

Tanveer, M. S., Khan, M. U. K., and Kyung, C. M. (2020). Fine-tuning DARTS for image classification. In *Proceedings - International Conference on Pattern Recognition*, pages 4789–4796. Institute of Electrical and Electronics Engineers (IEEE).

Torbert, S. (2016). *Applied Computer Science*. Springer International Publishing, 2nd edition.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, 2017-Decem:5999–6009.

von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14(2):85–100.

Von Luxburg, U., Radl, A., and Hein, M. (2014). Hitting and Commute Times in Large Random Neighborhood Graphs. Technical report.

Wallis, G. (1996). Using spatio-temporal correlations to learn invariant object recognition. *Neural Networks*, 9(9):1513–1519.

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

Wang, X., Han, T. X., and Yan, S. (2009). An HOG-LBP human detector with partial occlusion handling. In *2009 IEEE 12th International Conference on Computer Vision*, pages 32–39. Institute of Electrical and Electronics Engineers (IEEE).

Weiner, K. S. and Zilles, K. (2016). The anatomical and functional specialization of the fusiform gyrus. *Neuropsychologia*, 83:48–62.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. `http://arxiv.org/abs/1708.07747`.

Yamins, D. L. and DiCarlo, J. J. (2016a). Eight open questions in the computational modeling of higher sensory cortex Brief review of recent progress. *Current Opinion in Neurobiology*, 37:114–120.

Yamins, D. L. and DiCarlo, J. J. (2016b). Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, 19:356.

Yates, P. A., Holub, A. D., McLaughlin, T., Sejnowski, T. J., and O'Leary, D. D. M. (2004). Computational modeling of retinotopic map development to define contributions of EphA-ephrinA gradients, axon-axon interactions, and patterned activity. *Journal of Neurobiology*, 59(1):95–113.

BIOGRAPHICAL SKETCH

James W. Ryland started working in Dr. Alice O'Toole's lab and doing independent projects under Dr. Richard Golden in 2009 as an undergrad. He continued working on research as a master's student. During this time, he worked on a variety of projects including fMRI visualization techniques and building neural network architectures for complex planning. While working on these projects his interest in deep convolutional neural networks (DCNNs) as a models of the human ventral pathway started to take shape.

For his first year project as a PhD in 2013, James built an extended DCNN hybrid model in order to better understand human object recognition phenomena related to recognition over different viewpoints. Starting in 2017, he started exploring whether structural connectivity like that found in the brain and in DCNNs could be learned in an effective and efficient manner in artificial neural networks. This culminated in the publication of the paper "Modeling Axonal Plasticity in Artificial Neural Networks" in the journal *Neural Processing Letters*. Finally, for his dissertation, started in 2020, he turned his attention to implementing more faithful models of simple and complex cell learning that could also model behavior similar to DCNNs. In order to do this, he developed and integrated models of axonal plasticity, cortical map learning, and trace rule learning into a unified architecture. As part of this project, he also developed the Temporal Relation Manifold Framework of unsupervised category learning for object recognition.

CURRICULUM VITAE

James Ryland

Email: jwr071000@utdallas.edu
Phone: 1-(281)-217-3501
linkedin.com/in/james-ryland-86534a43/

Education

- **PhD in Cognition Neuroscience** **2021**
  The University of Texas at Dallas
  Advisor: Richard Golden |Co-Advisor: Alice O'Toole
- **MS in Applied Cognition Neuroscience** **2013**
  The University of Texas at Dallas, Richardson, TX
- **BS in Cognitive Science** **2011**
  The University of Texas at Dallas, Richardson, TX

External Funding

- **Raytheon Dissertation Research Grant** **Sept 2020 - Dec 2021**
  The University of Texas at Dallas
  PI: Richard Golden |Senior Scientist: James Ryland

Publications

- **Peer Reviewed Journals**
  - Ryland, J. (2021). Modeling Axonal Plasticity in Artificial Neural Networks. Neural Processing Letters.1-28
  - O'Toole, A. J., Natu, V., An, X., Rice, A., Ryland, J. & Phillips, P. J. (2014). The neural representation of faces and bodies in motion and at rest. NeuroImage. 91, 1-11
- **Manuscripts in Preperations**
  - Ryland, J. (Unpublished). From Untangling Object Manifolds to Untangling Temporal Relations Manifolds
  - Ryland, J. (Unpublished). Using 20-Questions to Explain Distributed Coding in DCNNs
  - Ryland, J. (Unpublished). Visualizing Sparse and Distributed Codes
  - Ryland, J. (Unpublished). Hybrid Theories of Visual Object Recognition: Combining Convolutional Neural Networks with Classic Theories
- **Conference Abstracts & Posters**
  - Ryland, J. (2020, July). From Tangled Object Manifolds to Temporal Relation Manifolds. The 41st Annual Meeting of the Cognitive Science Society. Montreal, Canada.

- Ryland, J. (2019, July). Modeling Axonal Plasticity in Artificial Neural Networks. The 41st Annual Meeting of the Cognitive Science Society. Montreal, Canada.
- Ryland, J. (2017, July). Rethinking windows and pools: how might models grow domain appropriate local connectivity structures? The 50th Annual Meeting of the Society for Mathematical Psychology, Warwick, England.
- Ryland, J. (2015). Orientation, Congruency, and Rotary Motion: Models of Object Identification. The annual convention of the Vision Sciences Society, St. Pete's Beach, FL.
- Ryland, J. (2013). Volumetric Visualization Technique for Brain Imaging. 2013 International Joint Conference on Neural Networks. Dallas, TX.

Research

- **Dissertation: Testing the Integrated Cortical Learning Model** **Sep 2021**
  - Developed new Integrated Cortical Learning Model as an alternative to CNNs which incorporates axonal plasticity
  - Developed a theoretical framework for understanding category learning in deep cortical networks without strong top-down signals
  - Developed experimental procedures to compare novel models to human cortical representations
  - Learned Keras, TensorFlow 2.x, PyTorch, Google Colab
  - `https://bitbucket.org/jryland/icl-dissertation/src/master/`
- **Axonal Plasticity Modeling for Deep Learning** **2018-2019**
  - Developed a neural network that simulates axonal plasticity between layers
  - Developed computational theories of axonal plasticity and cortical map learning
  - Developed first network to simulate large scale axonal and synaptic cortical maps
  - Achieved previous state-of-the-art (81%) for non-convolutional methods CIFAR10
  - `https://bitbucket.org/jryland/axon-game-paper-repo/src/master/`
- **Appropriate Window Learning for Deep Learning** **2017-2018**
  - Developed novel self-organizing map and neighbor embedding method
  - Used to control structural connectivity between layers
  - Learned Python and TensorFlow 1.x, Anaconda, Spyder
- **First Year Project: Hybrid Models of Object Recognition** **2014-2016**
  - Extended convolutional weight-sharing & pooling for Rotation
  - Developed novel coordinate transformation architecture
  - Compared networks to human recognition behavior over transformations
  - Extensions featured implementations of visual attention, saccades, and saliency maps, optimized for multi-scale perception of high-res real world datasets
  - Learned to program deep CNNs from scratch in MATLAB, and Statistical analysis in R

- **Volumetric fMRI Visualization** **2010-2014**
  - Graduate RA - The University of Texas at Dallas, Richardson, TX
  - Processed fMRI and MRI data for visualization as a MATLAB GUI application
  - Created first of its kind volumetric fMRI layer editing tool for visualization
  - `www.mathworks.com/matlabcentral/fileexchange/59161-volumetric-3`

Work Experience
The University of Texas at Dallas

- **Statistical Machine Learning Teaching Lab Coordinator** **2016-2020**
  - Taught advanced students about special topics in ML
  - Helped students conduct their first ML projects
  - Taught advanced tensor derivation methods and notation
- **Intelligent Systems Design – Teaching Assistant** **2016-2020**
- **Neural Network Mathematics – Teaching Assistant** **2016-2020**
  - Introduced students to fundamentals of gradient descent, Matrix Calculus, standard ML architectures, Bayesian, and MCMC methods
  - Developed complementary independent curriculum with a highly interactive problem oriented perspective for clarifying and illustrating concepts
  - Taught advanced tensor derivation methods and notation
  - Developed new pedagogical notations adapted from tensor calculus for easily working with and optimizing neural network math and illustrating the fundamentals of optimized auto-derivation.
- **Cognitive Neural Modeling Lab – Teaching Assistant** **2016-2020**
- **Experimental Methods – Teaching Assistant** **2014-2016**
- **Statistics for Psychology – Teaching Assistant** **2013-2014**

Additional Skills

- **Programming Languages**
  Python, MATLAB, R, C#, Java, C, Processing
- **Programming Packages**
  TensorFlow 1.x & 2.x, Theano, ggplots, matplotlib, SciPy, Numpy, OpenCV
- **Misc. Technical Skills**
  Blender 3D, SPM, Unity 3D, Audacity, VR Asset Creation, Shader Writing, Arduino & Microcontroller programming

Research References

- **Richard Golden** — Program Head — (email) `golden@utdallas.edu`
- **Alice O'Toole** — Professor — (email) `otoole@utdallas.edu`
- **Peter Assmann** — Professor — (email) `assmann@utdallas.edu`