Retinotopic Preservation in Deep Belief Network Visual Learning

by

Michael Lam

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2011

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

One of the foremost characteristics of the mammalian visual system is the retinotopic mapping observed in the low-level visual processing centres; the spatial pattern of activation in the lateral geniculate nucleus and primary visual cortex corresponds topologically to the pattern of light falling on the retina. Various vision systems have been developed that take advantage of structured input such as retinotopy, however these systems are often not biologically plausible. Using a parsimonious approach for implementing retinotopy, one that is based on the biology of our visual pathway, we run simulations of visual learning using a deep belief network (DBN). Experiments show that we can successfully produce receptive fields and activation maps typical of the LGN and visual cortex respectively. These results may indicate a possible avenue of exploration into discovering the workings of the early visual system (and possibly more) on a neuronal level.

Acknowledgements

I would like to thank my supervisor for his compassion, enthusiasm, humour, and general understanding of human nature which goes much farther than one might expect. I would also like to thank my parents for their support. And finally I would like to thank my friends (yes, both real and internet ones) for keeping me sane.

Dedication

This is dedicated to my keyboard, and cute, fluffy animals everywhere. You are what make this world a better place. Keep on being fluffy.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Reading this thesis comes effortlessly, much like many of the other tasks we perform in our day to day lives. We look at the words, deduce their meaning, and convert this into information we can process, all in a faction of a second. While this may appear simple, the task of vision is in fact a very complicated process. It may be surprising to those that are not familiar with visual computation, but our visual system is performing an amazing feat nearly flawlessly for most of our waking hours. Computational vision has come very far in emulating human vision — no trivial job. However, we have not been able to reproduce the solution to the task that biology seems to have so elegantly solved. The question remains, how was this accomplished, and what are we missing?

Organization is one of the key components inherent in many systems of the body. From the simple organization of cells to the complex interaction of organs, the body is characterized by structure. Of these structures, we wish to examine the layered topographical system that exists in our visual system. In doing so, one might ask: How does our body achieve this level of organization? Is it a result of complex pre-determined genetics, or is it a result of learning and training that occurs on a neuron-by-neuron basis?

In the past decades, more and more resources have gone into the advancement of neural network research of vision. Deep learning algorithms have recently been motivated by the hierarchical cortex organization. Visual neural networks have evolved to the point where we can produce networks that can learn to discriminate handwritten digits or detect pedes-

trians in complex scenes. Can we now produce a neural model that can faithfully mimic learning in the human visual system? If not, is this because we have yet to reach the level of complexity that the brain has evolved to? Are we unable to reach the same computational efficiency, or the same visual fidelity? Or are our networks missing fundamental attributes inherent in the mammalian visual system? Have we simply yet to stumble upon the correct learning method? Perhaps the answer is a combination of all of the above.

This thesis aims to develop a biologically inspired variant of the deep belief network learning, and analyze the results of implementing such a network. We wish to demonstrate that a single learning adjustment effectively allows the network to reproduce several biological phenomena found in the visual system, and is a useful avenue of exploration. By reproducing such phenomena we hope that we may be able to explain why the visual system would implement such an approach. Our work draws from two different areas of research in an attempt to generate a better computational model of the human visual system: The biology of neuroscience, and the computation of artificial neural networks.

1.1 Thesis Structure

Chapter 2 introduces biological background information that we consider to be important to understanding the thesis. Due to the nature of the investigation into a model with strong biological ties, we discuss the physical biology of the relevant parts of the mammalian visual system as well as the main underlying concept of retinotopology.

In chapter 3 we carefully describe the Restricted Boltzmann Machine (RBM), of which the proposed model, the Proximal Restricted Boltzmann Machine (PRBM), is based. We also present a previously developed approach for creating deep architectures using stacked RBMs. Finally, we outline state-of-the-art models used to model visual learning.

Chapter 4 presents a more detailed discussion of the proximal learning variation. First, we explain how the RBM framework can be easily modified to simulate retinotopic learning. We focus on how we can update the training procedure using likelihood learning and Contrastive Divergence learning. Additionally, we discuss some limitations of the model in regards to human vision.

Chapter 5 presents the experiments conducted using a 2-level PRBM on a variety of data sets. We begin our analysis with a discussion on the properties of the each data set. We then analyze the experimental results for each layer separately, discussing the experimental procedures, along with techniques in visualization and our interpretation of the results.

Finally, in Chapter 6 we briefly reiterate the findings of our investigation, draw some general conclusions about the representational power of the PRBM model with respect to the mammalian visual system, and provide some directions for future work.

Chapter 2

Biology of Vision

Our visual system is a complex piece of machinery, and in order to fully understand how such a system works, we must understand how each of its pieces operates. The specific biological mechanisms that guide the development of the visual system are yet unknown, but a glance at the physiological operation of each part of such a system may give us enough insight into creating a more complete model that learns to process visual stimuli.

In this chapter, we will take a look at some of the primary biological components in the early processing of vision. The early areas of the visual system are those that are concerned with breaking images down into components or smaller features that make up the whole image, instead of high level tasks such as object recognition. This section of the visual system will be broken down into three main parts, all connected through the optical pathway. The first section we will cover is the retina, contained within the eye. Beyond the eye, we have the lateral geniculate nucleus (LGN), and finally the primary visual cortex. But first, it is important to take a look at the fundamental computational unit in the brain, the neuron.

2.1 The Neuron

Human cognition is achieved through special types of cells called neurons. These cells are capable of forming little processes called axons and dendrites that are projected from one

neuron to another. These processes form connections between different neurons, producing a network of interconnected neurons. The process of strengthening, and pruning of connections between neurons is the fundamental act of biological learning.

The connections are the basis of what makes a neuron special. Neurons are able to receive input from neighbouring neurons, do some processing, and then output the result to other neurons. The electrochemical interaction between the axon of one neuron and the dendrite of another neuron takes place in structures called synapses.

A synapse is a tiny gap between two neurons. Electrical activity in the pre-synaptic neuron causes a release of chemicals, called neurotransmitters, into the gap. The neurotransmitter diffuses across the gap and interacts with the post-synaptic neuron, altering its electrical activity. Figure 2.1 depicts a synapse, along with the general structure of a neuron.

Physical communication primarily occurs through electrical signals in the form of either action potentials or graded potentials. Action potentials are electrical pulses or spikes. They can be considered Boolean operators, as the they are either spiking or not. The frequency of the impulses can be used to encode intensity values. Graded potentials, on the other hand, are characterized by a difference in voltage, and thus encode a range of amplitudes directly.

We can think of a simplified model of how a neuron processes information by considering the output firing rate or strength of a neuron to be a linear combination of its inputs. Some inputs have a tendency to excite the neuron, while others tend to inhibit the firing of a neuron. Furthermore, different inputs are favoured more than others. Whether or not a given neuron will then fire an output depends on the integration of its excitation and inhibition inputs.

Additional details regarding the fundamental properties of neurons can be found in [17].

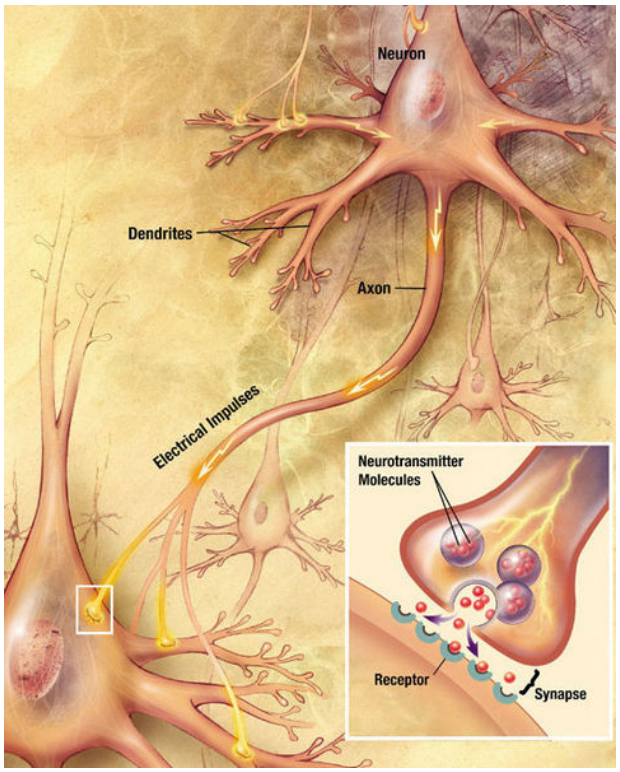


Figure 2.1: Structural make-up of a neuron. The foundation of learning, a neuron is composed of different parts that facilitate transmission of information.

2.2 Early Areas of The Visual System

Understanding how vision is processed begins in the eye. The eye is composed of several structures that facilitate vision. One of these structures, the retina, is the first sensing organ in vision and part of the central nervous system. A sheet of light sensitive tissue that lines the inside of the eye forms the retina, tasked with the main job of processing physical light that has been focused onto it, delivering that initial information down the optic nerve. The retina itself is a complex structure, composed of several types of neurons.

The absorbtion of light and transformation into electrical signals is performed by neurons in the retina called photoreceptors. Photoreceptors are classified as rods or cones, each tasked with the role of processing light under specific conditions. We will not concern

ourselves too much with the specifics of photoreceptors, rather simply note that they are the primary tool for visual input from the outside world. The electrical signals produced by these photoreceptors are graded potentials and travel down a variety of cell types as they are aggregated in the end by neurons called ganglion cells. In general there is roughly a 100-to-1 ratio of photoreceptors to ganglion cells [9]. The axons of these ganglion cells form the optic nerve and project further into the brain.

Other interspersed neurons in the retina contribute to how the photoreceptor signals are collected and transmitted to the ganglion cells. These interneurons are classified into lateral cells, such as the horizontal and amacrine cells, and direct cells, such as the bipolar cell. Horizontal and amacrine cells allow signals to transmit laterally within the retinal scope, while bipolar cells allow signals to pass from the photoreceptors straight to the ganglion cells.

The developmental growth and cumulative properties of all of these cells are of utmost importance to the biological view of the visual system. As a result of the role that the interneurons play, the photoreceptors and ganglion cells are mapped retinotopically to each other.

Retinotopy means that the pattern of light-activating photoreceptor cells in the retina corresponds spatially to the activation patterns of ganglion cells. Adjacent neurons have receptive fields that include slightly different, but overlapping portions of the visual field. For example, two nearby dots falling onto our retina will also stimulate visual-cortical neurons that are near to one-another[1] (see Figure 2.2). As one might expect, a ganglion cell tends to activate only when a proximal photoreceptor is excited. In later chapters we will see the implications of this important retinotopic mapping feature.

The ganglion cells are our first set of neurons that extend deeper down the visual system. Important visual information flows from these cells out of the retina to the lateral geniculate nucleus (LGN). We can see a rough overview of the flow of information depicted in Figure 2.3.

[1]There are exceptions; discontinuity exists between the left and right hemifields.

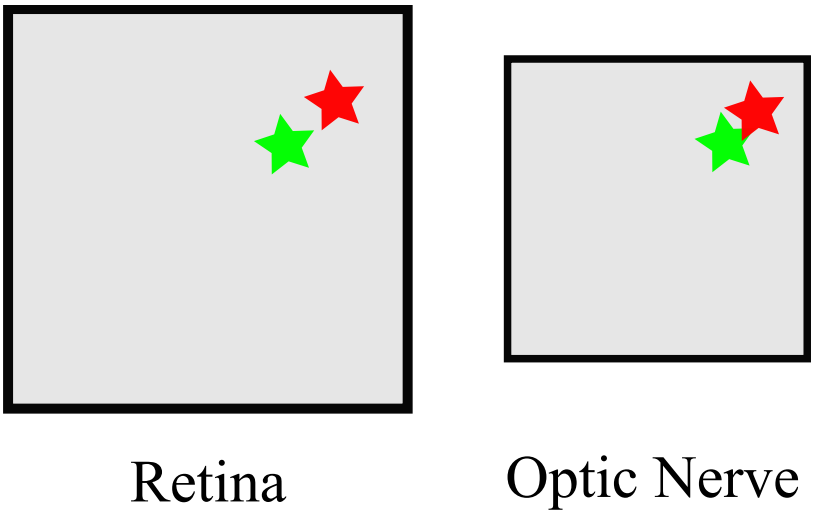


Figure 2.2: An example of retinotopic mapping. Input into the retina travels through the visual pathway topologically. Stimuli presented side-by-side will be encoded in a similar fashion at deeper layers in the visual system.

2.3 Lateral Geniculate Nucleus

One of the first major processing centres of the optical pathway is the lateral geniculate nucleus, or LGN. Ninety percent of retinal axons terminate at the lateral geniculate nucleus [23], making this a major centre for vision. Ganglion cells project in an orderly fashion to points in the LGN, so a retinotopic representation of the visual field is maintained from the retina. Physically, the LGN lies between the retina and the primary visual cortex. Neurons in the LGN send their axons to the primary visual cortex through a direct pathway. As this thesis is focused on the neuronal learning and response of the visual system, we will dwell less on the physical make-up and biology and more on the neural responses.

When we consider neural responses of visual neurons, we usually look to the concept of a receptive field. For the remainder of this thesis, we will define a receptive field as a representation of both the area and stimulus that is more likely to illicit a response from a neuron[2]. Each neuron has its own receptive field that can be thought of as some

[2]In some texts there is a distinction between receptive fields and tuning curves; receptive fields depict

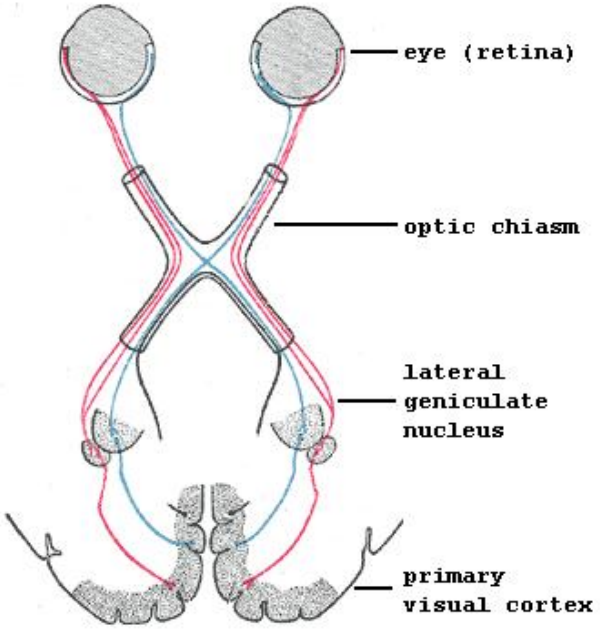


Figure 2.3: The visual pathway in a human brain. Information for vision starts at the retina and is sequentially processed by each section of neurons as it moves deeper into the visual system.

combination of activations of neurons connected to it. Neurons receiving input directly from a single source (such as photoreceptors) however have no receptive field. Simple examples of receptive fields are shown in Figure 2.4.

Neurons in the LGN often have approximately circular receptive fields with specialized central and surround regions [17]. These fall into two classifications: on-centre cells, and off-centre cells. On-centre ganglion cells become excited when light stimulates the central region of their receptive field, and are inhibited when light stimulates the annulus that surrounds it. In contrast, off-centre ganglion cells become excited when light hits the surround area, and are inhibited by light falling on the centre of the receptive field. A simple example of these LGN receptive fields can be seen in Figure 2.5.

the area in which a neuron receives input from (its field of view), while tuning curves illustrate the actual neural responses.

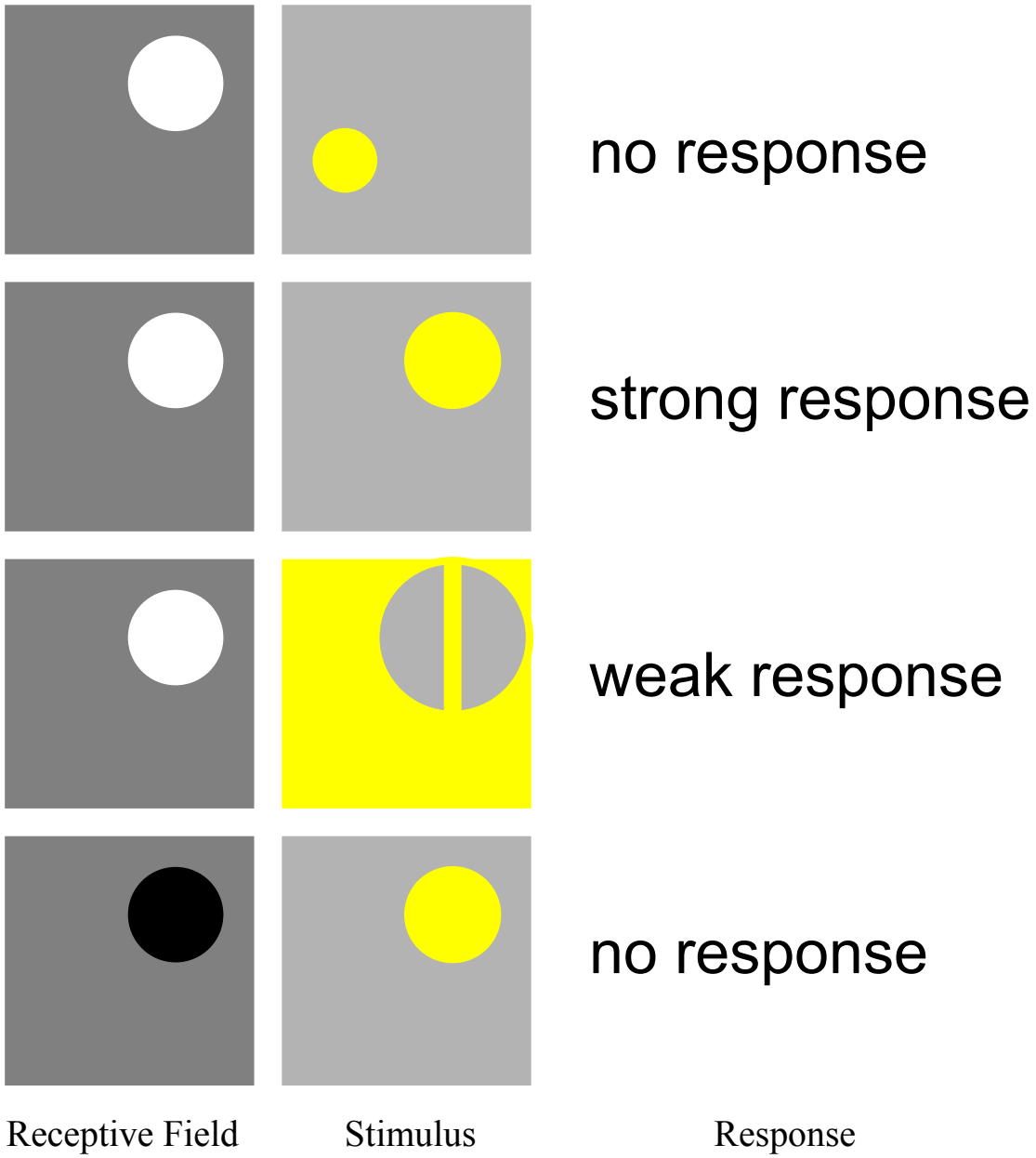
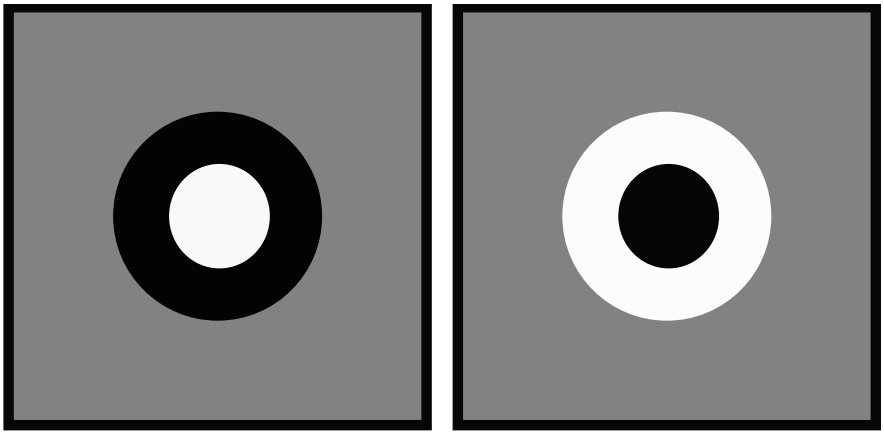


Figure 2.4: Visual receptive fields concept. Each row shows a two-dimensional receptive field, a stimulus, and the resulting response. In the receptive fields, grey is neutral, white is excited by light, black is inhibited by light. For the stimulus, yellow/bright indicates the area being illuminated. The examples here do not necessarily represent real cells, but simply explain how receptive fields are being interpreted in this thesis.



(a) On-centre (b) Off-centre

Figure 2.5: Biological LGN Receptive Fields. These centre-surround receptive fields are typically the types of patterns one might find the neurons in the LGN responding to.

2.4 V1

The primary visual cortex, or V1, is where a majority of the information coming from the LGN is received. It is the earliest cortical visual area, and one of the better understood.

Like the LGN, our knowledge of the visual area V1 stems largely from electrophysiological probing of animals such as ferrets, cats, and monkeys. In particular, neuronal response patterns of the early visual system have been established and documented in many of these experimental studies [21, 26, 27]. While some of these aspects include structural features such as the ratio of V1 neurons to LGN neurons [21], more significant findings include the actual neuronal responses.

In particular, research done by Hubel and Wiesel has demonstrated that the receptive fields in the V1 cortical region are composed of very specific activation patterns [12]. They found that most cells beyond the lateral geniculate nucleus responded best to stimuli that had linear properties, such as lines and bars. Since then, further research has gone into classifying and mapping responses of different types of cells in the visual area V1. It is now clearly known that the primary visual cortex has a more complicated gamut of receptive

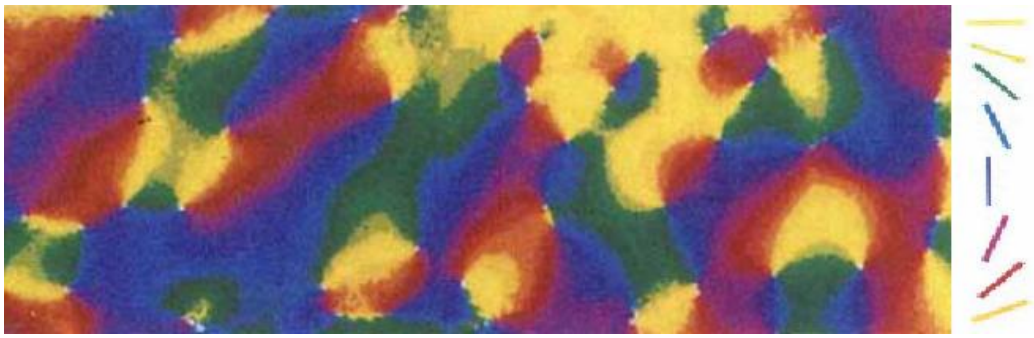


Figure 2.6: An example of a non-simulated orientation map taken from a paper by Bonhoeffer et al. [6]. This activation map was created by probing and imaging the neural activity of a cat while it was being shown oriented visual stimuli. Colours in the image represent neural affinity towards different angled stimuli. Different locations in the image represent different neurons.

field types than found in the LGN.

The V1 neurons that respond to oriented lines possess a distinct distribution pattern. This pattern forms a clustered activation map. If we colour code each neuron with the strongest neural response for various orientations, then we arrive at a diagram, called an orientation map, seen in Figure 2.6. Neurons that encode similar orientations tend to cluster together. The cells that exhibit these different patterns generally fall into the classifications of simple cells and complex cells [16].

Simple cells have receptive fields that are generally characterized as filters that are spatially local and oriented. For example, a V1 simple cell neuron might fire rapidly when presented with a light grating oriented at 45°, but not at all to a grating at 135°. Neurons that encode similar orientations tend to cluster together into sections, partitioning V1. Hubel and Wiesel also proposed that simple cell receptive fields are typically composed of the summation of multiple offset centre-surround receptive fields.

Complex cells also have receptive fields characterized by oriented edge detectors, however these cells are more tolerant to variations, such as phase shifts, or are excited regardless of where the edge occurs in the receptive field.

Chapter 3

Neural Networks

Theoretical models of the visual system often abstract the notion of neurons and connections into a graph-theoretic network of nodes and edges. These models are called artificial neural networks. We will focus our attention on a particular kind of neural network, the restricted Boltzmann machine, followed by a series of extensions that make the restricted Boltzmann machine more suited to processing vision.

3.1 Restricted Boltzmann Machines

The restricted Boltzmann machine (RBM) is a constraint satisfaction network [1], capable of learning hidden, underlying constraints that characterize a domain. This learning is achieved simply by having the network train on various examples from the domain.

Structurally, an RBM is a bipartite undirected network model. It consists of two layers of stochastic variables. These variables (called neurons, nodes, or units) can adopt binary or real-valued states. Real valued neurons would typically be used as the input neurons to encode light intensity (as the graded potentials of the photoreceptors would do in a biological system), while binary valued neurons would be more suitable for determining whether or not a neuron is firing (an action potential). These states depend on the states of adjacent nodes and their associated connectivity weights (or synaptic connections). We

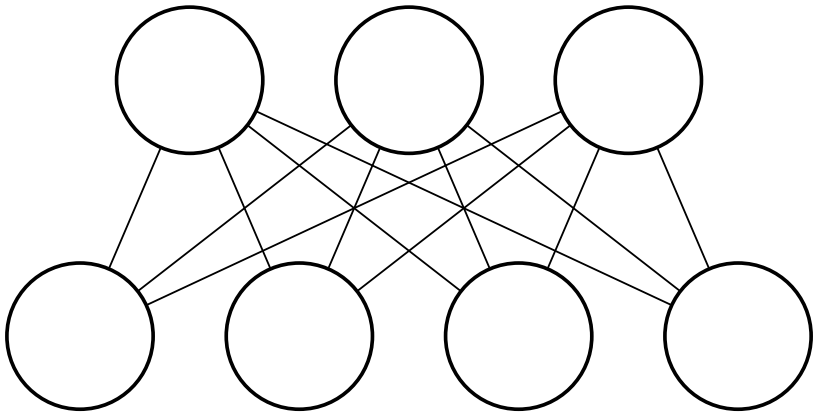


Figure 3.1: Restricted Boltzmann machine. An RBM contains two variably-sized layers of neurons or nodes. Each circle represents one of these nodes, while the lines between the nodes indicate connections. Layers are fully connected to one-another, but contain no connections within a layer.

will later see in more detail how these weights affect the network, however in general these weights describe weak constraints between the association of two neurons.

The two layers themselves are often labeled the *visible* layer and the *hidden* layer, purely for descriptive purposes. We refer to the visible layer as the layer that is directly influenced by the environment (i.e. is the interface between the 'outside world' and the model), while the hidden layer receives input through its connections with the visible layer. The hidden units generally act as free variables, and their output is used to explain the constraints underlying the visible input. If we look at the structure of the connectivity, each neuron in a specific layer is completely connected to the neurons in the adjacent layer, however, intra-layer connections are prohibited. Figure 3.1 depicts the connective structure of a simple RBM.

RBMs can be ideal for certain tasks that, instead of enforcing strict rules such as board-game rules, can be described using a large number of weak constraints. These constraints incur a cost to the model when broken, but are never strictly enforced. The justification for such a model is that each rule contributes to a plausibility or likelihood of the final result. When we model something like visual responses we can arrive at much more robust

solutions [1], as a visual scene itself can typically be broken down into much simpler features which represent these weak constraints.

We can think of using these constraints as a way of combining different visual features that, while possibly violating some implicit constraints, are still good interpretations of the input. The constraints characterize the distribution of inputs (visual scenes) and the relationships between individual neurons that represent the characteristics implicit in visual scenes — presumably features such as edges formed from pixel intensity, or perhaps more complex aspects.

3.1.1 Energy of an RBM

Each global configuration of the network can be assigned a numerical value, or an energy. This energy can be represented as a function of the node activations, as well as their connection strengths. Let the activity of the visible and hidden nodes be represented as vectors v and h respectively, and the synaptic connections between these neurons as a weight matrix w. Using this notation, we can compute an energy value from the activity of the nodes,

$$E(v,h) = -\sum_i \sum_j w_{ij} v_i h_j + \sum_i b_i v_i + \sum_j c_j h_j, \tag{3.1}$$

where v_i and h_j represent the activity of the ith visible node and jth hidden node, respectively. The strength of the synaptic connection between those two neurons is indicated by w_{ij}, and b_i and c_j are bias or threshold terms affecting those neurons. These bias terms state how much the energy of the system is altered by simply turning a single neuron on. The energy of a specific state vector can be denoted as $E(v,h)$ for specific visible and hidden vectors v and h.

We can also use (3.1) to determine the difference in energy between turning a given visible node v_k on or off

$$\Delta E_k = \sum_j w_{kj} h_j + b_k. \tag{3.2}$$

Similarly for hidden nodes, simply replace the v's with h's, and use b's for the bias terms.

Thus far we have simply defined energy to be a sum of active nodes, and adjustable parameters such as weights and biases. However if we constrain the network such that some neurons must either be on or off, then the network will attempt to find a minimal energy configuration that is compatible with the given input. We call this process of forcing certain nodes to have a particular state 'clamping' the network. Once we have clamped such a network, we can use the energy function (3.1) to determine the probability of being in a certain global state. The probabilistic model (i.e. the probability of the network having visible vector v and hidden vector h) for an RBM is defined by the Boltzmann distribution

$$P(v, h) = \frac{1}{Z} \exp(-E(v, h)/T) \ , \tag{3.3}$$

where Z is the partition function (the total of the possible states, normalizing the probability), v and h denote vectors of visible and hidden variables, $E(v, h)$ is the energy function of that particular state of visible and hidden variables, and T is a temperature parameter (more on this later). Minimal energy occurs on likely observations, so we have an inverse relationship between energy and probability. Lower energy states have a higher probability of occurring. Thus we should drive down the energy of the states for input that we expect to see, or that the network is trained on. Likewise, we should drive the energy up for states that are highly improbable, or rarely observed in the training environment.

Energy in such a network can be represented as a value anywhere between $-\infty$ and $+\infty$. If our objective was to minimize the energy of the network, we might imagine the network would simply try to increase all the weight values w so that turning every node on reduces the energy. However, the bias terms counteract this possibility by also increasing the network's energy as each neuron is activated, leading us to the common global minimization task.

What this entails is that a lower energy is produced when both a visible and hidden unit are on simultaneously, and they have a positive weight value between them. If the weight is negative, then turning both units on increases the energy. Ultimately, reducing the energy term becomes a complex balance of finding the right combination of weights, biases, and nodes to turn on such that the reduction in global energy sustained from their connections exceeds the increase in energy afforded by each active neuron's bias term.

3.1.2 Equilibrium

Like in a simple state machine, we could look for a set of states that the network falls into after running indefinitely, and this would constitute our equilibrium. However, our neural network operates on non-deterministic principles. Given an initial state, the sequence of states that follow may be highly variable between runs. Since there is no closed set of states that is visited as we run the network infinitely, we must identify a different equilibrium. Instead of looking for a specific set of states, we can consider the set of all states, and observe how often the network visits each state. By doing this, we can see that the *distribution* of the visited states is consistent in the network. That is, as we run the network to infinity, the frequency at which it visits a particular state will be stable across multiple runs. We call this the equilibrium distribution or thermal equilibrium.

An RBM uses simulated annealing to avoid local minima by introducing a non-deterministic aspect to the configuration probability. This is done by using the energy from equation (3.2) to set the state of node k to 1 regardless of its previous state using the logistic function

$$p(k = 1) = \frac{1}{1 + e^{-\Delta E_k / T}}, \tag{3.4}$$

where T is the temperature parameter. This function denotes the probability that node k will be active. By applying the sigmoidal function (seen in Figure 3.2) to node activations, we can see that it is possible to produce different sets of states even given the same input. This built in non-linearity means that after running the network for long enough at a certain temperature, the probability of a distribution of states is only dependent on the global energy. This is considered thermal equilibrium for the network, and is characterized by a distribution of state visitation frequency.

Since we are working with distributions of states, we will introduce a specific notation $V_\delta = v^\delta$ to represent a specific configuration of nodes. Note that that notation v^δ is *not* equivalent to v_i, as the former indicates vector v in state δ, while the latter indicates the specific visible node indexed by i. Therefore we can combine these notations to produce v_i^δ which represents visible node i in state δ. In general, if we do not need to index specific nodes, we will use the notation V_δ instead. Also note, this same notation applies to the hidden units, using H instead of V and h instead of v. Furthermore, we will assume that

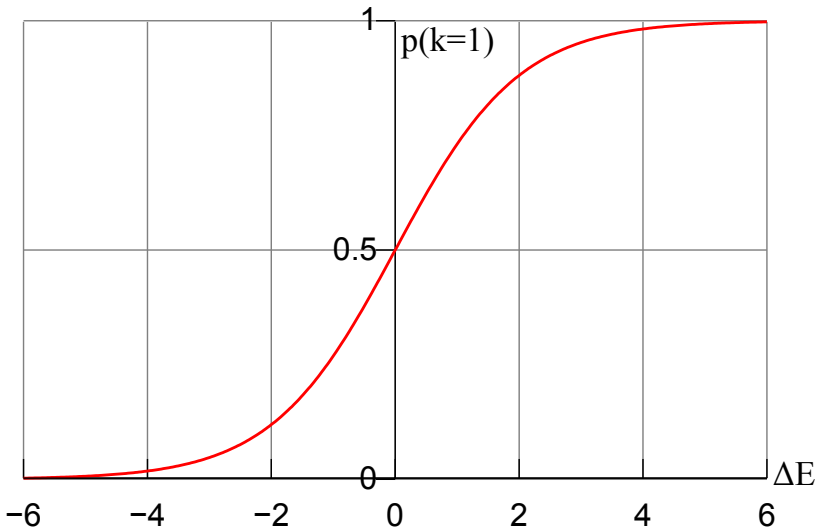


Figure 3.2: Logistic function for $T = 1$. The additional temperature parameter T can affect the steepness of the function. Higher energy values have a greater chance of turning node k to 1.

each such node configuration can be accessed by only a single index. For example, if we suppose the are N configurations of the visible nodes, then $\{V_\alpha | \alpha = 1, \ldots, N\}$ is the set of all configurations of visible nodes.

Due to the non-deterministic state generation, there is no one state or set of states that constitutes an equilibrium for the model. As the network continues to run, certain nodes will change state according to (3.4) regardless of the current state. However, if the network continues to run freely, the energy will approach a minimum. If the network is untrained, this minimum may be meaningless, but the network will move towards states with minimal energy regardless. Again, the distribution of the states as the network runs to infinity (which is dictated by the energy of the model, and therefore the weights) is the equilibrium distribution. Figure 3.3 shows a simple example of state distribution and adjusting the distribution based on some learning.

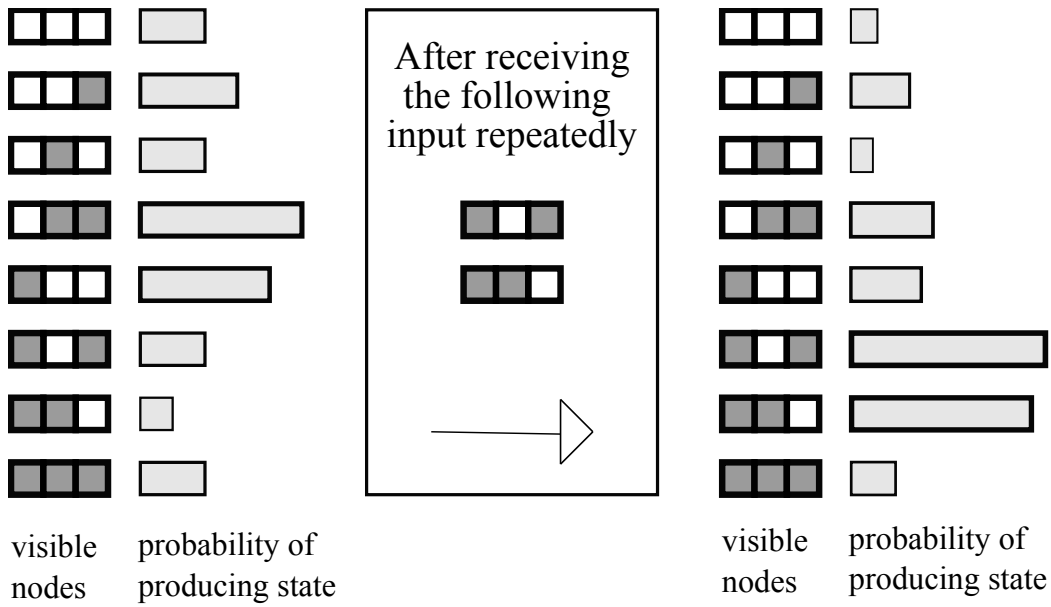


Figure 3.3: Example of learning from visible node distributions. Given a free running network, suppose the energy is distributed in such a way that the bars show the relative distribution of states that will be produced. This demonstrates what an equilibrium distribution may look like. After doing some sort of energy adjustment (in this case learning that it should be producing the same distribution as the inputs), the new energies produce the visible distribution on the right. Note that at any isolated time, the network could jump to any state as it is not bound to stay within a subset of the total amount of states (though the probability of such a jump would depend on the energy of that state).

3.1.3 Learning Rule

Suppose we had two input vectors (vectors of visible nodes) that were each presented to the network half of the time. If the network was designed to learn to model its environment, we would expect the network at equilibrium to be reproducing only these two vectors. Using this idea, we may consider modelling the environment using a probability distribution over all the possible input states of the visible units. In the previous example, we would expect such a distribution to give half of its weight to each of the two input vectors, and no weight to any other possible inputs. When the network is capable of producing a probability distribution (over all input states) that is identical to the probability distribution of the training (or input) data, then we consider the network to have a perfect model of the environment. In order to measure this difference between the internal model of the network and the environment distribution, we use the Kullback-Leibler (KL) divergence measure

$$G = \sum_{\alpha} P(V_\alpha) \ln \frac{P(V_\alpha)}{P'(V_\alpha)} \tag{3.5}$$

where α iterates over all the visible state possibilities, $P(V_\alpha)$ is the probability of being in the α'th visible state when clamped to the environmental distribution, and $P'(V_\alpha)$ is the probability of being in the same visible state when the network is running unclamped. This is analogous to $P(V_\alpha)$ being the probability of V_α occurring in the environmental distribution. Since we are only observing the visible states, $\sum_\alpha P(V_\alpha)$ will be 1 as we are fixing the states in this case, and also the distribution over the visible units will be independent of the weights (if the visible units are set to the environment then they will not change).

Kullback-Leibler divergence gives a number that represents the distance from one distribution to another. This number increases as the difference between distributions increases, and is only 0 if the distributions are identical. The KL divergence can be altered by changing the energy of the system, and thus changing the probability of different configurations. We ultimately want our network to have a perfect model of our input domain. The visible or hidden neurons cannot be altered to adjust the energy of the system, however we can change the weights in order to optimize the KL divergence. Therefore if we wish to minimize the KL divergence, we need to perform gradient descent of the KL divergence in

the space of the connection weights. We achieve this by taking the partial derivative of G with respect to each of the connection weights. As stated earlier $P(V_\alpha)$ is independent of the weights, so $P'(V_\alpha)$ is the only term that depends on the weights. Hence, we first need to differentiate $P'(V_\alpha)$, defined as

$$P'(V_\alpha) = \sum_\beta P'(V_\alpha, H_\beta), \tag{3.6}$$

where V_α and H_β are states of the visible and hidden neurons respectively and β indexes all of the possible hidden states. According to the Boltzmann distribution, we have

$$P(V_\alpha, H_\beta) = \frac{1}{Z} e^{-E_{\alpha\beta}/T}, \tag{3.7}$$

where $E_{\alpha\beta}$ denotes $E(V_\alpha, H_\beta)$. Therefore we can observe

$$\sum_\beta P'(V_\alpha, H_\beta) = \frac{\sum_\beta e^{-E_{\alpha\beta}/T}}{\sum_\delta \sum_\beta e^{-E_{\delta\beta}/T}}, \tag{3.8}$$

where the numerator is proportional to the probability of state V_α given the priors of the hidden states, and the denominator sums over all possible visible and hidden states. As stated earlier, we know the energy of a system in the state $E(V_\alpha, H_\beta)$ can be represented by

$$E_{\alpha\beta} = -\sum_i \sum_j w_{ij} v_i^\alpha h_j^\beta + \sum_i b_i v_i^\alpha + \sum_j c_j h_j^\beta. \tag{3.9}$$

Differentiating the Boltzmann distribution from (3.7) with respect to the weight w_{ij}, we get

$$\frac{\partial e^{-E_{\alpha\beta}/T}}{\partial w_{ij}} = \frac{1}{T} v_i^\alpha h_j^\beta e^{-E_{\alpha\beta}/T}. \tag{3.10}$$

Using the equations (3.6), (3.8), and (3.10), we differentiate $P'(V_\alpha)$ (using the quotient rule) with respect to the weights to get

$$\frac{\partial P'(V_\alpha)}{\partial w_{ij}} = \frac{\frac{1}{T}\sum_\beta e^{-E_{\alpha\beta}/T} v_i^\alpha h_j^\beta \sum_\delta \sum_\beta e^{-E_{\delta\beta}/T} - \frac{1}{T}\sum_\beta e^{-E_{\alpha\beta}/T} \sum_\delta \sum_\beta e^{-E_{\delta\beta}/T} v_i^\delta h_j^\beta}{(\sum_\delta \sum_\beta e^{-E_{\delta\beta}/T})^2}, \tag{3.11}$$

which simplifies to

$$\frac{\partial P'(V_\alpha)}{\partial w_{ij}} = \frac{\frac{1}{T}\sum_\beta e^{-E_{\alpha\beta}/T} v_i^\alpha h_j^\beta}{\sum_\delta \sum_\beta e^{-E_{\delta\beta}/T}} - \frac{\frac{1}{T}\sum_\beta e^{-E_{\alpha\beta}/T} \sum_\delta \sum_\beta e^{-E_{\delta\beta}/T} v_i^\delta h_j^\beta}{(\sum_\delta \sum_\beta e^{-E_{\delta\beta}/T})^2}. \tag{3.12}$$

We want our equation to take this form so that we can express it in terms of the probability of being in certain states. Using substitutions from equations (3.6) and (3.8), we can further simplify to

$$\frac{\partial P'(V_\alpha)}{\partial w_{ij}} = \frac{1}{T}\left(\sum_\beta P'(V_\alpha, H_\beta) v_i^\alpha h_j^\beta - P'(V_\alpha)\sum_\delta \sum_\beta P'(V_\delta, H_\beta) v_i^\delta h_j^\beta\right). \tag{3.13}$$

This derivative term can then be used to take the gradient of G, remembering that $P(V_\alpha)$ is not affected by the weights (since the visible units are clamped to state α, and thus are not influenced by the hidden units).

$$\frac{\partial G}{\partial w_{ij}} = -\sum_\alpha \frac{P(V_\alpha)}{P'(V_\alpha)} \frac{\partial P'(V_\alpha)}{\partial w_{ij}} \tag{3.14}$$

$$= -\sum_\alpha \frac{P(V_\alpha)}{P'(V_\alpha)} \frac{1}{T}\left(\sum_\beta P'(V_\alpha, H_\beta) v_i^\alpha h_j^\beta - P'(V_\alpha)\sum_\delta \sum_\beta P'(V_\delta, H_\beta) v_i^\delta h_j^\beta\right) \tag{3.15}$$

Using the definition of conditional probability,

$$P(B|A) = \frac{P(A,B)}{P(A)}, \tag{3.16}$$

we know that

$$P(V_\alpha, H_\beta) = P(V_\alpha)P(H_\beta|V_\alpha), \text{and} \tag{3.17}$$

$$P'(V_\alpha, H_\beta) = P(V_\alpha)P'(H_\beta|V_\alpha). \tag{3.18}$$

Since in the ideal equilibrium, the probability of the hidden states given the visible will be the same regardless of being clamped or running freely, we have

$$P(H_\beta|V_\alpha) = P'(H_\beta|V_\alpha). \tag{3.19}$$

After substituting (3.19) into (3.17) and (3.18), we can re-arrange to get

$$P'(V_\alpha, H_\beta)\frac{P(V_\alpha)}{P'(V_\alpha)} = P(V_\alpha, H_\beta). \tag{3.20}$$

Substituting (3.20) into the gradient term in equation (3.15) (remembering $\sum_\alpha P(V_\alpha) = 1$) gives us

$$\frac{\partial G}{\partial w_{ij}} = -\frac{1}{T}\left(\sum_\alpha \sum_\beta P(V_\alpha, H_\beta) v_i^{\alpha\beta} h_j^{\alpha\beta} - \sum_\delta \sum_\beta P'(V_\delta, H_\beta) v_i^{\delta\beta} h_j^{\delta\beta}\right), \tag{3.21}$$

where the first probability term specifies the average probability of the visible node i and hidden node j being on when the environment is clamped; we will denote this clamped probability as p_{ij}. The second probability term is the same probability when the network is running freely, denoted as p'_{ij}. Using this notation we can simply write the gradient as

$$\frac{\partial G}{\partial w_{ij}} = -\frac{1}{T}(p_{ij} - p'_{ij}). \tag{3.22}$$

Thus if we wish to minimize the KL divergence using gradient-descent, we can observe the probability of $v_i h_j$ being "on" in the clamped and free equilibrium states, and adjust the weights proportionally to the difference,

$$\Delta w_{ij} = s(p_{ij} - p'_{ij}), \tag{3.23}$$

where s is the learning rate of the network.

Thus far we have established a learning rule for the weights, but have yet to touch the bias terms b and c that are attached to each visible and hidden neuron respectively. However, at this point we can make one simple observation that allows us to adapt the exact same learning procedure for learning weights to learning biases. If we imagine that each neuron is connected to an imaginary 'bias node' that is always in the "on" state, then we can consider the bias term for each neuron as the weight to this node, and these 'bias weights' can simply be treated as learning another weight to a neuron that is always on. The imaginary node itself plays no part in the neural learning, as it retains a constant state. We can use this information to complete our set of learning equations,

$$\Delta b_i = s(b_i - b'_i), \tag{3.24}$$

$$\Delta c_j = s(c_j - c'_j), \tag{3.25}$$

where b_i is the bias term for visible neuron v_i, and c_j is the bias term attached to hidden neuron h_j.

3.1.4 Contrastive Divergence

To get an estimate of the gradient, it is necessary to run Gibbs sampling until it converges on the equilibrium distribution. However, doing so is generally computationally intractable. Maximizing the log likelihood of the data is equivalent to minimizing the Kullback-Leibler divergence between the visible data input distribution Q_0, and the equilibrium visible data distribution Q_∞ [11].

$$\frac{\partial(Q_0||Q_\infty)}{\partial w_{ij}} = \langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^\infty . \tag{3.26}$$

Here, the operator $(\cdots||\cdots)$ is the KL divergence, Q_0 represents the initial probability distribution of the visible states in the clamped phase, and Q_∞ represents the equilibrium probability distribution of visible states after the network has been unclamped. The angle-bracket terms $\langle v_i h_j \rangle^0$ and $\langle v_i h_j \rangle^\infty$ denote the expected sampled values of the product $v_i h_j$, initially and at equilibrium. In general it is not particularly feasible to arrive at the equilibrium state Q_∞, as this requires many iterations of sampling back and forth between visible and hidden states (Gibbs sampling).

Instead of minimizing the Kullback-Leibler divergence of the data distribution and the equilibrium distribution $(Q_0||Q_\infty)$, we instead simply minimize the difference between $Q_0||Q_\infty$ and $Q_1||Q_\infty$. The intuition behind this approach is that because we know our desired distribution is composed of our training input, we want to discourage network behaviour that results in producing states that are not like the input. In turn we want to favour behaviour that produces states close to the input.

Computationally we can see that the intractable term $\langle v_i h_j \rangle^\infty$ cancels out, and we are left with

$$\frac{\partial}{\partial w_{ij}}(Q_0||Q_\infty - Q_1||Q_\infty) = \langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^1 + \frac{\partial Q_1}{\partial w_{ij}} \frac{\partial Q_1||Q_\infty}{\partial Q_1}, \tag{3.27}$$

where the final term in the equation is effectively ignored because "it is small and seldom opposes the resultant of the other two terms" [11]. We know that by taking one step closer to the equilibrium distribution, $Q_0||Q_\infty$ is always greater than $Q_1||Q_\infty$ (KL divergence is greater, unless Q_0 is equal to Q_1), so that the difference is never negative. This process is shown in Figure 3.4.

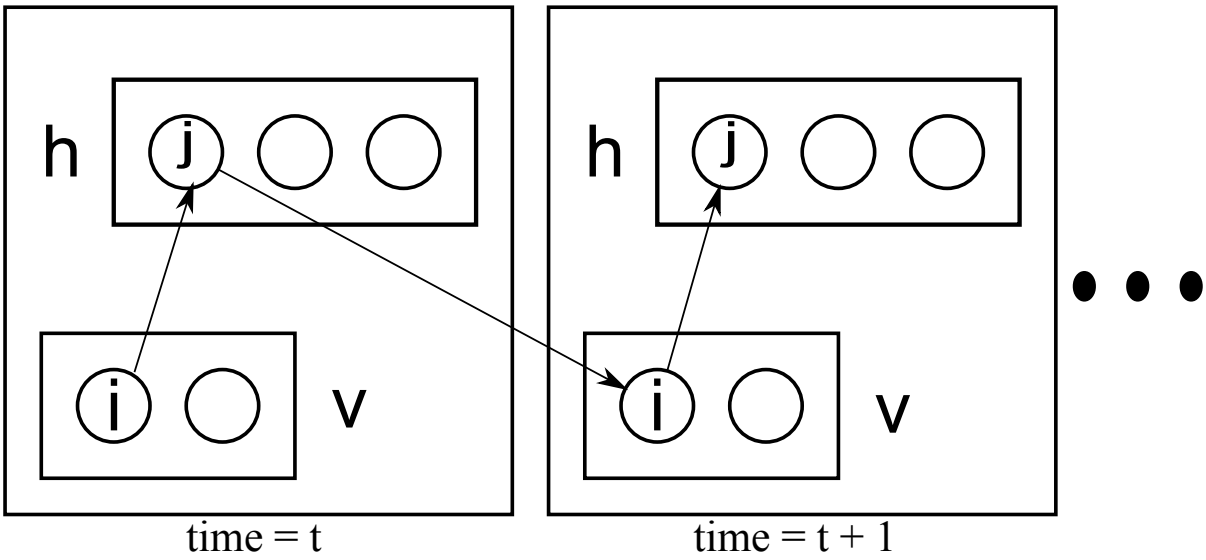


Figure 3.4: One step in contrastive divergence learning, depicting a single interaction between visible neuron i and hidden neuron j. At time t, we sample the state of the hidden units, and use that information along with that of the clamped visible units to determine correlation between neurons. At time $t + 1$, we sample the visible units based on the hidden units, and re-sample the hidden units based on the new visible units to determine the correlations after one step of Gibbs sampling. Depending on the approach, this process could be repeated for several more sampling steps.

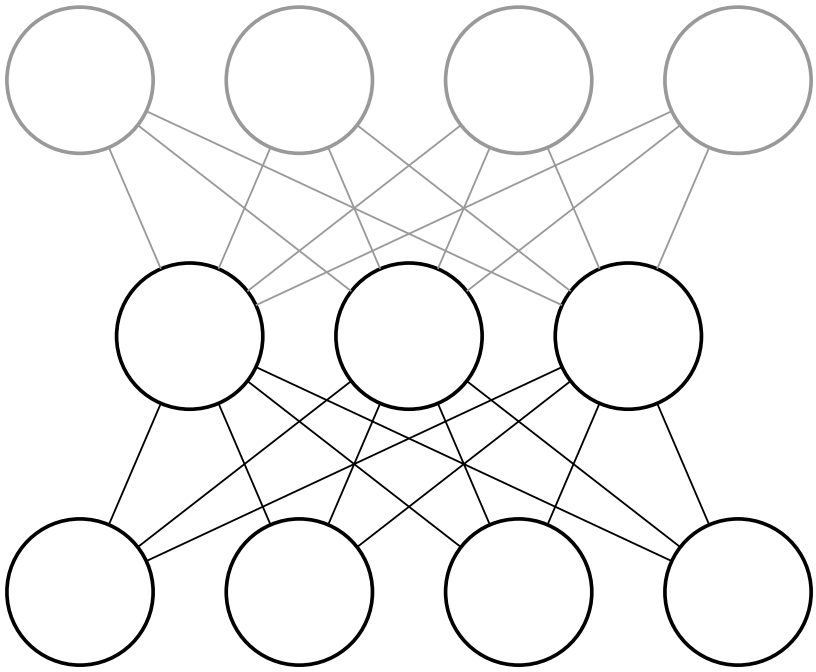


Figure 3.5: Deep belief network. A three-layered DBN composed of two RBMs. The highlighted layers depict one such RBM (the other RBM would be composed of the top and middle layers).

3.2 Deep Belief Networks

Deep Belief Networks (DBN) are probabilistic models composed of multiple layers of stochastic variables. To build our DBN, we need only stack several RBMs on top of each other (see Figure 3.5). Each new layer of hidden variables act as higher-order constraints upon the previous layer. That is, a layer's hidden units become the visible units of the next layer. We can do learning on a layer-by-layer basis using a greedy approach. By assuming the weight matrices for a generative and recognition model are tied (use the same value either way), we can use the standard RBM learning algorithm. Once we learn the weights for one layer, we effectively freeze them, and force the next layer to learn based on the now transformed data.

We continue to use the weight update equations (3.23), (3.24), and (3.25) to determine

how to change the weights and biases for each layer. These equations are modified slightly to incorporate contrastive divergence. Thus, to determine Δw, Δb, and Δc we use

$$\begin{aligned} \Delta b_j &= b_j + s(\langle h_j \rangle_{\text{orig}} - \langle h_j \rangle_{\text{rec}}) && (3.28) \\ \Delta c_i &= c_i + s(\langle v_i \rangle_{\text{orig}} - \langle v_i \rangle_{\text{rec}}) && (3.29) \\ \Delta w_{ij} &= w_{ij} + s(\langle v_i h_j \rangle_{\text{orig}} - \langle v_i h_j \rangle_{\text{rec}}) \ , && (3.30) \end{aligned}$$

where s is the learning rate and $\langle \cdots \rangle_{\text{rec}}$ is the expectation over the reconstructed data after an iteration of Gibbs sampling.

Deep belief networks provide us with several useful properties for modelling vision. First of all, they provide us with the layering that occurs in the visual system. Second they provide us with the feedforward and feedback unsupervised learning system. In addition to these properties, our goal is to explore the biological viability of using a modified DBN.

Earlier it was stated that neurons can be real or binary valued. The equation (3.31) was derived for binary neurons, however real valued neurons function in a nearly identical manner. The difference is how we express energy:

$$E(v,h) = \frac{1}{2}\sum_i v_i^2 - \sum_i \sum_j w_{ij} v_i h_j + \sum_i b_i v_i + \sum_j c_j h_j. \tag{3.31}$$

Consequently, this effects how we determine the probability of a neuron taking on a value. Our probability of observing a visible neuron given the hidden units is

$$P(v_i, h) = \mathcal{N}\left(\sum_j w_{ij} h_j + c_i, 1\right), \tag{3.32}$$

where $\mathcal{N}$ is the normal (Gaussian) distribution. For a hidden neuron, the equation is the same except the h's and v's would be swapped, and we would use the hidden unit bias b.

3.3 Neural Network Models of the Visual System

Incorporating aspects of sparsity and regularization to a deep learning network is an idea that has been approached many times in the past, as sparsity has been a long researched

topic in the area of visual representation. A number of recent algorithms have demonstrated that aspects of the mammalian visual system (primarily V1) can be modelled as a DBN or even an RBM [25]. However, greedy, layer-wise training of these models does not, by itself, produce the sparse retinotopic mapping that we observe in mammalian vision. Some of the recent architectures incorporating learning from a deep belief network include sparse restricted Boltzmann machines (SRBM) [18], convolutional restricted Boltzmann machines (CRBM) [22], and topographic independent component analysis [14]. These methods are successful at producing compelling V1-type receptive fields typical of the human visual system, but do it in a way that may not be biologically plausible, or use an approach that is not parsimonious.

3.3.1 Sparse Restricted Boltzmann Machines

Sparse restricted Boltzmann machines are a modification of standard RBMs that incorporate an additional regularization term into the model's learning. The purpose of the regularization term is to enforce sparsity by penalizing hidden-unit activity. The approach follows the same contrastive divergence learning procedure outlined in section 3.1.4 for RBMs, as well as using the same update equations (3.23), (3.24), and (3.25).

The difference occurs during the training steps. Once the KL divergence gradient term with respect to the weights is determined using contrastive divergence, the gradient of the additional regularization term is also used to further update the parameters in a separate step. These operations are repeated until convergence.

The new objective function (akin to KL divergence, plus a regularization term) is defined as

$$G_{\text{new}} = G + \lambda \sum_{j=1}^{n} \left| p - \frac{1}{m} \sum_{i=1}^{m} \langle h_j | v \rangle \right|^2 , \tag{3.33}$$

where G_{new} is the new objective function to optimize, G is the original function from (3.5), m is the number of training vectors, n denotes the number of possible hidden units, λ is a regularization constant, and p is a constant that controls the sparseness of the hidden neurons. We can see that the additional term uses a sum over all of the hidden units, so it cannot be computed using only local information.

It is suggested that an SRBM is capable of producing receptive fields similar to those found in V1 and V2 [18]. While aspects of sparsity appear to play important roles in biological vision, we also want to consider other important biological premises. SRBMs apply sparsity through a global term that sums up hidden unit activity, however there is biology does not favour this type of global learning and information span (neuron dendrites extend only so far). In fact, to the contrary, neurons seem to only use local information available to them. Also, while SRBMs (and any deep belief networks created from them) have sparse activations, their receptive fields will not be arranged in any retinotopic fashion. Much like a standard RBM, the receptive fields and hidden units may be permuted freely to produce identical networks.

3.3.2 Convolutional Restricted Boltzmann Machines

Convolutional restricted Boltzmann machines (CRBM) represent another class of modified RBMs. A convolutional RBM functions similarly to a standard RBM, however the weights in a CRBM are shared between all locations in the visible layer [22].

One standard method for extracting spatially local features from an image is called filtering. Hidden units are grouped into filter banks, where each group is characterized by a filter kernel. A filter kernel is a rectangular matrix of coefficients that represents a spatially local visual feature. The hidden units then encode the convolution of the input and the kernel. CRBMs use this idea of filtering to determine the receptive field of each kernel. Each hidden unit bank represents a set of small local receptive fields that are learned. These receptive fields are invariant to some linear transformations, such as translation or scaling, depending on the domain used (the kernels produced in the papers [22, 8] were translation invariant).

The most basic CRBM consists of an array of visible units and K "clusters" of hidden unit arrays. Each of the K groups is associated with a filter kernel. This kernel is associated with a single weight such that all the hidden units in a group share this activation weight. We can think of each hidden unit in group k to have identical connection weights to visible unit i. The energy function for a CRBM is similar to the energy definition for an RBM

(3.1) and is defined as

$$E = -\sum_{k=1}^{K}(w^k * v) \cdot h^k + b\sum_i v_i + \sum_{k=1}^{K} c^k \sum_j h_j^k, \tag{3.34}$$

where w^k, c^k, and h^k represent the weight, biases, and hidden unit activations for filter k respectively, $*$ denotes the convolution operation, and $\cdot$ denotes a matrix dot product. Higher layer representations of the input are learned by stacking CRBMs into a multilayer network analogous to a DBN, with an additional deterministic step between each layer for the purposes of scaling.

Not surprisingly, it was found that a sparse representation was required to achieve desired results. One method for achieving this was to use the regularization term from 3.3.1 to force a small amount of hidden neurons to activate.

The results produced from a stacked CRBM look like the receptive fields of V1, however the main drawback with adopting filters is that the set of weights described in a specific filter must be shared among many connections in the network. This means that a neuron must make, for all intents and purposes, functionally identical connections to potentially thousands of other neurons. In other words, the connection weight must be duplicated between many pairs of neurons. These weights must also be updated in the exact same manner synchronously. This identical learning (or generation of synaptic connections) must occur for every neuron in each filter. We find that there is no known biologically plausible mechanism that could account for these identical weights. Again, the sparsity term uses global information to constrict neuron activation.

3.3.3 Topographic Independent Component Analysis

Independent Component Analysis (ICA) is a statistical blind source separation technique. By applying a neighbourhood function to ICA, Hyvarinen et al. produce a variant of ICA that also utilizes a topography constraint [13].

Instead of modelling learning in multiple layers, this network model uses topographic ICA learning for the first layer and then uses fixed pooling weights to the second layer

to achieve topography. The pooling of the second layer influences how the first layer is learned since the only learning that occurs in the model is at the first layer. The weights to the pooling layer do not change.

The results produced by topographic ICA are very convincing, and appear to be more biologically plausible than the previous mentioned approaches. We find however that the approach jumps straight to mapping V1 receptive fields, skipping the LGN which we believe is important to model. Furthermore, the model learns receptive fields and then topographically pools the receptive fields together using a fixed weighting scheme to produce the pooled receptive fields, instead of learning how to pool receptive fields together. This is analogous to biologically already having a well formed V1 neural configuration before beginning to learn representations between the retina and LGN.

Chapter 4

Proximity Modulated Training

4.1 Retinotopic Topology

A standard RBM or DBN has no notion of spatial topology since each node is typically connected to every node in the next layer, with no regard to position. As a result, every node can be permuted (along with the corresponding weights) to produce another equivalent network. Every permutation of the network of a standard DBN is identical in representational power.

To better model the physical biology of the visual system, we can add a proximity constraint in the network's learning process. As discussed earlier in chapter 2, the process of embryonic development results in a retinotopic organization of the ganglion cells in the optic tract. This organization continues into the V1 visual cortex. When the optic tract reaches the LGN, the ganglion cells begin to make connections to the neurons in the next area. If we consider neurons extending axons out to produce these connections, then two neurons that are closer together can connect more readily, and with greater ease, than neurons that are distant from each other. Hence, the retinotopic proximity of two neurons should be factored into the learning rule.

The model can use this constraint to take advantage of the strong two-dimensional topology already existent in the pictures that are used for training. This topology can be

understood as local interactions between pixels; input, or pixels near each other, should have a high degree of correlation compared to those that are far apart. In other words, important early visual features are more likely to arise (though not exclusively) in local pockets, rather than between pixels on the opposite ends of an image. Since pixels of a visual scene are stored in a grid structure, We can describe this topology in terms of a simple 2D matrix for a given input location, where proximity is considered as the Euclidian distance between input entries.

4.2 Incorporating Retinotopic Proximity

Proximity is implemented into the contrastive divergence training paradigm by modulating the learning rate between pairs of nodes according to their retinotopic proximity to each other. In particular, the learning rate for the connection between visible node i and hidden node j is multiplied by a normalized distance function based on Euclidian distance (L2 norm). For the remainder of this thesis, unless otherwise specified, this function is a normalized Gaussian function that peaks when visible node i and hidden node j are in the same retinotopic location, but falls off asymptotically to zero as the two nodes become retinotopically further apart.

The proximity function, depicted in Figure 4.1, can be adjusted in width from layer to layer, or even individually for each neuron. For our experiments, the proximity functions all used the same width within a given layer. In essence, these proximity functions act as a physical bias for axons and dendrites to make local connections, in the absence of any directed (developmental) growth. While traditionally we might use the learning rate as a means of expressing the speed at which neurotransmitters are exchanged, we may also use this biased learning rate as a measure of the quantity of connections, or the physical requirements in transmitting further, or even a bias towards maintaining closer synapses over distant ones.

The proximity condition appears only in the update equations for the connection weights, changing the weight update function in (3.30) to

$$\Delta w_{ij} \quad = \quad w_{ij} + s(\langle v_i h_j \rangle_{\text{orig}} - \langle v_i h_j \rangle_{\text{rec}}) p_{ij} \tag{4.1}$$

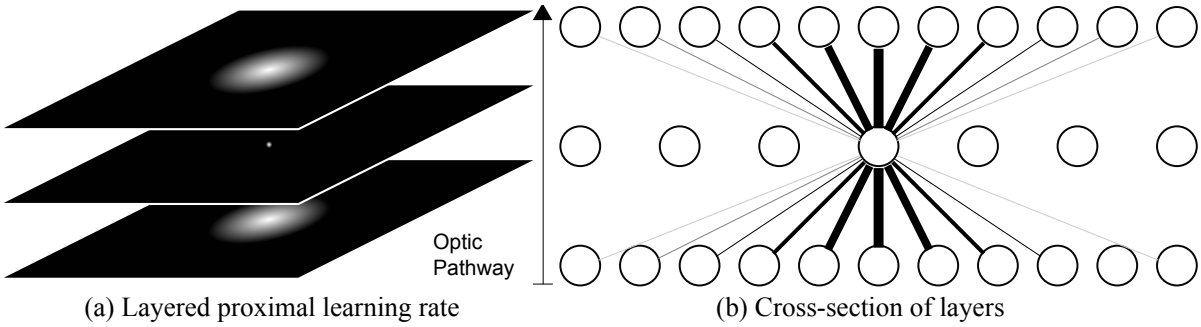


Figure 4.1: Retinotopic proximity functions. A three layered network, where each layer is drawn as a 2D array of neurons, is shown in (a). The bottom-most layer represents the visible layer, while layers on top represent subsequent hidden layers. One node is highlighted as a dot in the first hidden layer. The shading of the second hidden layer and visible layer reflects the retinotopic proximity to the highlighted neuron. In (b) we show a cross-section of the three layers, where each neuron is represented by a node, and each connection by an edge. The thickness of the graph edges implies the learning rate.

where p is a matrix that stores the proximity factor for each pair of nodes (p_{ij} is the proximity modification between nodes i and j). Neurons in this setting still act only on local information, and lack the ability to "share" connection strengths with other arbitrary units. Learning the strength of an individual connection to any node within any layer remains an entirely individualized process. Additionally, there are no other sparsity constraints that require observing the entire network state. By modifying the weighted learning rates, all the weights continue to move in their respective gradient directions, however only a certain subset of weights will be altered significantly.

The structure of the network itself is not an overly complex issue, though we do have some aspects we must consider. Moving between layers we must consider either growing to a larger space domain, shrinking to a smaller one, or remaining in the same space. This is naturally accomplished through changing the number of neurons present in each layer. As a consequence, distance between neurons in different layers is measured and scaled according to the visible layer. A layered depiction of a network is presented in Figure 4.2. The specifics of this scaling operation will be elaborated on in the experiments chapter.

Also note that as we move upward in the hierarchy (deeper into the network), unless we continue to constrict the proximity window, the receptive fields become larger in the sense that the image regions that have a significant effect on their activity grow.

4.3 Summary of Learning Algorithm

Algorithm 1 shows a summary of one step of the learning procedure for a single batch of input data on a single layer. This algorithm would generally be repeated for several iterations (epochs), much like contrastive divergence, before training the next layer.

Algorithm 1 Proximity Contrastive Divergence Learning (ProxCD).

```
For each epoch e
    For each input batch i
        Generate a proximity-rules matrix for each neuron
        Project vis → hid
        Gather neuron stats
        Project hid → vis (perform Gibbs sampling)
        Project vis → hid (again)
        Gather updated neuron stats
        Update the weights and biases, modulated by the proximity rules
    End For
End For
```

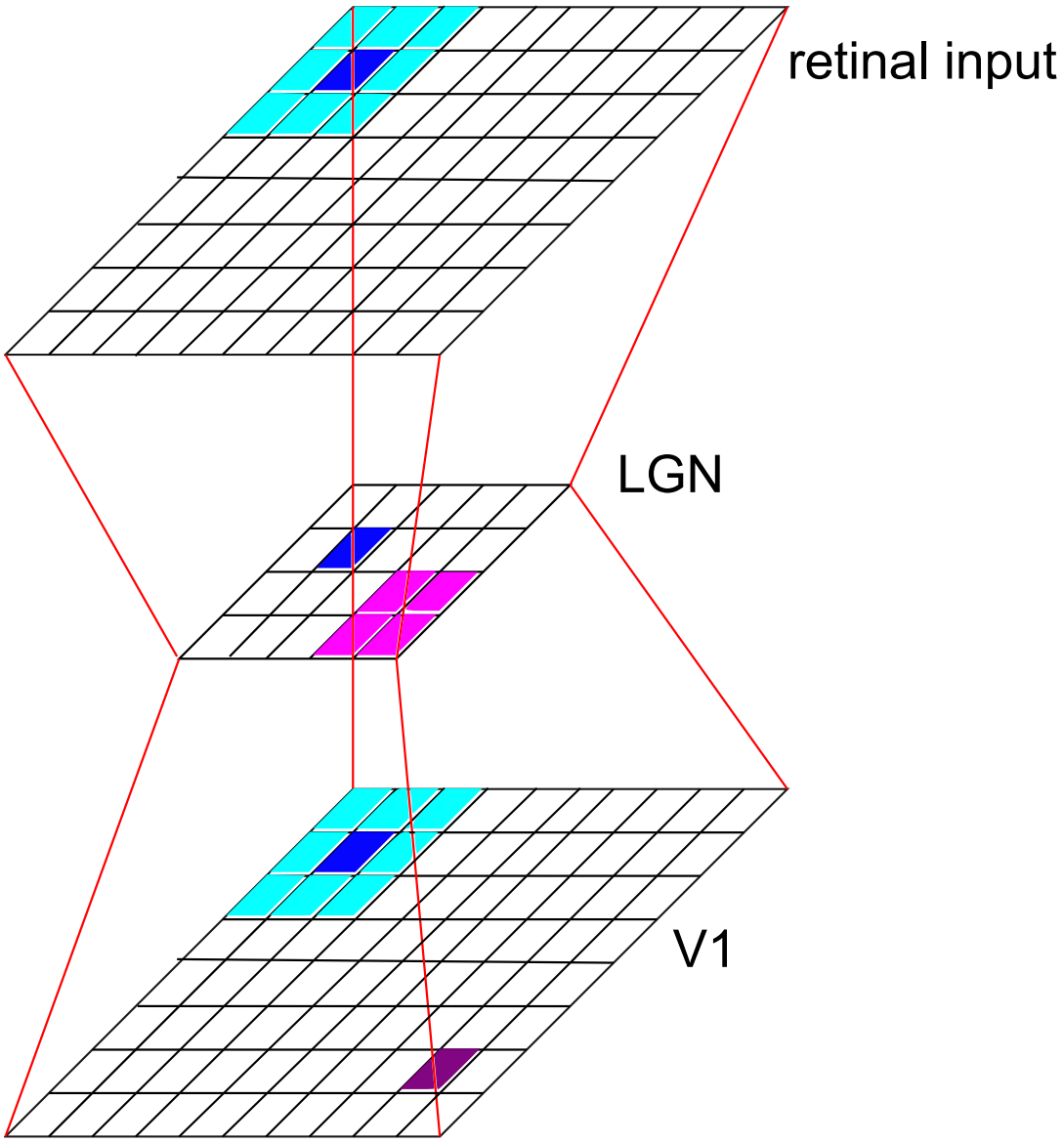


Figure 4.2: Layered proximal neuron mapping. Another three layered network is depicted, where each neuron is represented as a cell in a 2D array. Two separate neurons are highlighted for clarification. The LGN neuron illustrated in blue on the left side of the network is mapped to the neurons that are also illustrated in blue residing in the neighbouring layers (both up and down). Likewise, the purple V1 neuron on the right side of the network is mapped to the purple LGN neurons. These proximal connections go up and down the network in a similar fashion.

Chapter 5

Experiments

The purpose of these experiments is to demonstrate that by using proximity-modulated contrastive divergence learning (which will be referred to as ProxCD henceforth) in a deep belief network, we can produce a neural network that learns to model a system that is similar to one found in the early areas of the mammalian visual system. We show experimentally that under ProxCD learning, our artificial neurons produce the same type of responses and patterns that are typically observed in biological neurons. We also empirically compare the results of contrastive divergence (which will be referred to as CD) training to those of ProxCD training with a qualitative focus on the biological plausibility of each implementation. The comparisons show how the results support ProxCD as a more biologically fit alternative to modelling the human visual system.

The overall high-level architectural design of the multi-layer deep network uses the following pattern: an input layer of neurons, compressed to the LGN layer, and expanded to the final V1 layer. Our intuition for applying this kind of bottleneck feature on the number of available neurons draws heavily from the actual biology of the system. As we know from chapter 2, there are many photoreceptors that retinotopically map to each ganglion cell. We also know that upon arrival to the LGN, each ganglion cell is then further mapped retinotopically to a large number of V1 neurons [28].

For each separate testing instance, despite the fact that the human visual system is binocular in nature, we used only a simple network comprised of a single input layer,

LGN layer, and V1 cortical layer. We reason that each of the two LGN sections (one in each hemisphere) beyond the optic chiasm, encodes only one hemi-field of our vision (with contributions from both eyes). Similarly, each V1 section takes input from only its respective LGN section. We can use these biological facts to assume that in our single network, our input layer draws only from the one side of the visual field for which the LGN encodes. As this 'one-eyed' setup is incapable of representing input from the two eyes simultaneously, it precludes us from being able to investigate phenomena such as ocular dominance columns and other effects of binocular vision. However, this does not significantly affect the properties that we will be demonstrating and discussing in this thesis. There is some research also suggesting that separate eyes do not have to be trained simultaneously to produce matching orientation maps [10]. We will discuss these missing features in the conclusions chapter.

One interesting question that arises during experimentation is how do we accurately gauge how well a system can 'see like a mammalian system'. Deep belief networks are nice, in that they allow us to propagate states backwards through the network to 'see what the network believes', however there is no such mechanism that exists in the human visual system. A human brain simply processes the visual information present in V1 (that is sent further along the visual pipeline) in concert with stimuli from the other visual areas. It does not send the information back down to stimulate retinal cells[1]. We will look at techniques for visualization of the results along with our analysis of the layers themselves.

In this chapter, we will be using the terms 'input layer', 'LGN layer', and 'V1 layer' interchangeably with their respective network equivalents: the visible, first hidden, and second hidden layers. Additionally, when we discuss the training of a specific layer, we are talking about setting the connection weights from the previous layer to that layer. So when we are "training the LGN layer" we are discussing the network that involves the input layer and the LGN layer. Similarly, when we are "training the V1 layer", we are discussing the weights between the LGN layer and the V1 layer.

Beyond simply configuring the network, we also needed to determine what kind of

[1]V1 does send information back to the LGN. However, this does not help use visualize the receptive fields.

input a biologically plausible model might learn from, and what kind of input might help us determine if our network is learning what we expect. The most obvious choice, besides baseline tests, is a compilation of natural scenes. However, because the visual system has a unique capacity in what it can learn, and a simulated experiment is much easier to control than an animal such as a cat, we include several unique experimental data sets that are described in the next section.

In the following section we present the data sets that are used in the experiment. In the following two sections after that, we present our analysis concerning the LGN layer and the V1 layer.

5.1 Training Data Sets

Our goal is to show how the addition of this biologically inspired proximity function to the learning algorithm affects the growth of the weights in a deep belief network. In order to achieve this goal, we need to develop training data that would properly allow us to gauge the ability of the network to function in a way that is similar to the mammalian visual system. To train these networks, we need to examine types of input the visual system would typically be trained on. Due to the wide variety of possible responses and learning techniques that literature has pointed out, we created several different and unique training data sets.

For our training data we use an ensemble of six distinct data sets. In every training set, our inputs are 2D images treated as intensity-only, single-framed data. In other words, there is no colour used, nor are there any relationships between images or the order they are used as input. In addition, we feel that while the statistical properties of natural images belonging to different categories is relevant for scene and object categorization tasks [32], the model is not expected to be goal oriented, such as being a better classification network. Hence, the exact statistics of the visual scenes are not of overt importance.

We include a series of simple baseline training sets consisting of homogeneous grey-scale images (of varying intensities), homogeneous zero-intensity images (simply a black image being presented repeatedly), as well as sets of pure random Gaussian noise. These

training sets are simply used to compare against as a measure to see whether properties are emergent from the learning algorithm, the images, or a combination of both.

Beyond the simple tests, our main consideration is learning from natural images. The natural images data set used is the Orchard data set, collected courtesy of Professor Jeff Orchard. The training set consists of 400 images of various natural scenes. Natural scenes are considered generic images that would typically be seen by a human during an average day. These include nature scenes, pictures of people, objects, animals, and typical structures, with no particular emphasis or theme.

The question as to what specifically constitutes "natural images" is a debatable topic, as humans are exposed to a wide variety of visual stimuli. This has changed even more dramatically in the last century than it has over our evolutionary time period. However, if the mammalian visual system itself has not significantly changed between species, then it is unlikely that the learning method would take a wild detour in response to an increase in urban or interior scenes and a decrease in nature scenes. Either way, there is a clear difference in information redundancy and complexity between both urban or natural scenes, and the previously described uniform and random data sets.

Each image is a 240×240 pixel JPEG image. Examples of these images are shown in Figure 5.1. Samples are drawn from this set of 240×240 images by randomly choosing one of the 400 images, and then randomly choosing an appropriately-sized block from the image (matching the dimensions of the input layer). We also produce miniaturized Orchard training sets at 20×20 pixels, 50×50 pixels, and 100×100 pixels, however at small resolutions the pictures appear more like random noise than actual typical visual stimuli, even to a human observer.

According to a study done by Meister et al. [20] it also appears as though the visual system is stimulated by bursts of retinal activity (retinal waves) during early development. Since these retinal waves are being presented to the visual system before it is exposed to stimulation from the outside world, we believe they are an important aspect in shaping the training of our visual system. Thus, in addition to the natural world, we want to test the effect of training on these retinal waves in order to compare our networks to the study.

To create our synthetic retinal waves, oriented, repeating sine waves are generated

Figure 5.1: Samples from the natural images 'Orchard' training set (N set) depicting a wide range of visual stimuli and edge information that we are likely to encounter in the real world.

at various frequencies and converted into images. The orientations are randomly chosen angles drawn uniformly from $[0°, 360°]$. The images are then randomly shifted to new phases. Each image is created at a resolution of the image size required for visual input. See Figure 5.2 for examples.

Our final data set is more unorthodox, in an attempt to explore constrained training similar to the study done by Blakemore et al. [5]. We attempted to create a visual universe mostly composed of vertical and horizontal edge stimuli. The training set is composed of 19 images gathered from Flickr, and scaled down to 200×200 pixels. The lack of image variation in the training set is simply due to the lack of unique patterns that are likely to occur taking random pieces of the pictures in such a constrained universe. Sample images from this set are shown in Figure 5.3. As with the natural images, randomly selected sub-images are extracted and used for network training.

For the remainder of this thesis, we will sometimes be using the following shorthand notation to denote the different data sets: N for natural images, O for orthogonal-constrained images, W for retinal wave images, R for random Gaussian noise images, G for grey-scale images, and Z for homogeneous zero-intensity images. The term "X-trained network" may also be used occasionally, where X may denote any of the previously mentioned training sets. See Table 5.1 for reference.

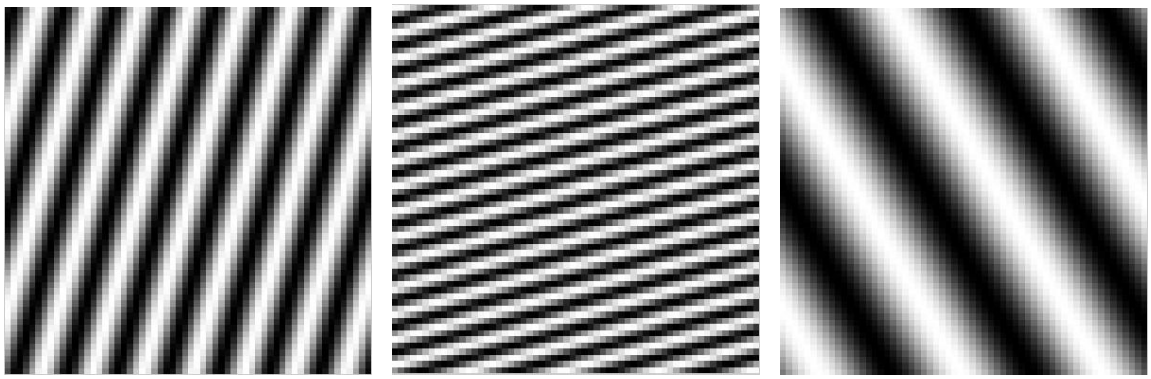


Figure 5.2: Samples from the retinal-waves training set (W set) simulating a series of wave-like signals passing through the retina.

Figure 5.3: Samples from the orthogonal images training set (O set) depicting mostly edges found vertically and horizontally.

Table 5.1: Data Set Shorthand

Symbol	Image Set
N	Natural images
O	Ortho-constrained images
W	Retinal-wave images
R	Random Gaussian noise images
G	Grey-scale images
Z	Zero-intensity images

5.2 Training the LGN Layer

5.2.1 Implementation

For our experiments, we used Matlab to implement the CD and ProxCD learning methods for deep belief networks. To approach our training comparisons and analysis, we produced several networks trained using the different data sets outlined in the previous section. Beyond the variation in training sets, our implementations of the networks consists mainly of setting up structural parameters such as layer sizing and proximity sizing, and learning parameters such as rate and duration.

Network Architecture

The structure of the network's weighted connections is determined by the size of both the input layer of neurons, and the LGN layer of neurons. If we want to think about this purely from a biological-structural perspective, the LGN layer should generally be made smaller in size than both the input retinal layer, and the subsequent V1 cortical layer.

For our networks we use a visible layer composed of 3600 neurons in a lattice of 60×60, connected to a hidden layer composed of 900 neurons in a 30×30 lattice. The visible layer is treated as a real-valued RBM, while the hidden layer is binary. If the input image

is larger than the visible layer, then we randomly sample an appropriate sized sub-image from the data sets, simulating random and momentary gaze fixation.

The chosen size of our input field does not allow a great deal of compression without losing too much information. Compressing by a ratio of $100:1$ (60×60 visible neurons into 6×6) results in a layer that is too small to analyze. However, we found empirically that the network is not hyper-sensitive to the degree of compression. The types of receptive fields produced remain the same judging from the difference between various networks ran using $120 \rightarrow 30$, $120 \rightarrow 60$, $60 \rightarrow 30$, and $60 \rightarrow 15$ (input layer to LGN layer). Computationally and analytically it is simply far more feasible to model a small portion of the retinotopic space. In addition, it is somewhat arguable to what degree at which resolution plays a role in the early stages of visual development in the first place, as the visual acuity of newborns is poor [2].

Training Duration

Training is done in phases called *epochs*. During each epoch, the network is trained once with each training image in the training data set. Typically we train the LGN layer for at least 20 epochs, though usually more. The networks specifically presented in this thesis are trained for 30 epochs using a specific learning schedule that will be discussed in the next section.

Learning Rate

The learning rate(s) for our networks varies over epochs. To initially train most of the networks, we use a learning schedule of $s = 0.05$ for the first 5 epochs, $s = 0.50$ for the next 5 epochs, and beyond that we use a learning rate of $s = 1.00$. This is used as a typical approach to make sure the learning does not diverge initially. Once we have established weights, the learning rate is increased to speed up convergence. The proximal learning technique is not particularly sensitive to the exact fine-tuning adjustments of the learning rate, but regardless this learning schedule was still used for both CD and ProxCD methods so that we could have a comparable set of networks.

Proximity Function

Probably the most influential factor in determining the contents of the receptive fields produced by the network is the size of the Gaussian proximity function. In the ProxCD method, we utilized a narrow proximity function. We used a Gaussian with a standard deviation of 1/60 of the length of the LGN neural space, and normalized so that the maximum value is 1.00 and the remaining values fall between 1.00 and 0.00.

5.2.2 Visualization

One of the main difficulties in interpreting the learned structure is the lack of a definitive measure for the quantitative assessment of what is actually being represented in the neural net. As a result, our conclusions are based on the examination of different figures, which include interpreted receptive fields, and orientation maps.

Many of the connections that run through the visual system are reciprocal. That is to say, there are feedback connections associated with feedforward connections. In our model, these feedback connections are useful for assessing the accuracy of the network's ability to reproduce the given input. The receptive fields themselves do not require the recognition weights, however, when we wish to see what the network "believes", we must utilize the generative weights. It may not be so clear in a biological sense to be using feedback connections from the LGN cells to the retinal input, however they give us a convenient and simplified way of determining what each neuron in the LGN layer responds to.

We know the receptive field of a given neuron in the LGN layer since we know the weights between the input layer and the LGN layer. The first hidden layer is directly connected to the input layer and we can use this property to perform a simple visualization of the weight vectors in this layer. By outputting the weights connecting to a neuron in a retinotopic manner, we can simply get a glimpse of the type of image (input) that the given neuron responds to. In essence, what we see is the 'optimal' input that would most likely excite a neuron (turn it on).

5.2.3 LGN Generated Receptive Fields

Natural Images

As we can see from Figure 5.4, the receptive fields produced in our LGN layer resemble the biological version seen in Figure 2.4 in section 2. The pattern of the receptive fields produced by ProxCD illustrates a central region that is either excitatory or inhibitory (light or dark) with an opposing surround region. The receptive fields in these cases are also localized to a small region near the "physical" location of the neuron. This is in direct contrast to CD learning, where the receptive fields are global in nature. Figure 5.4 shows this contrast in locality.

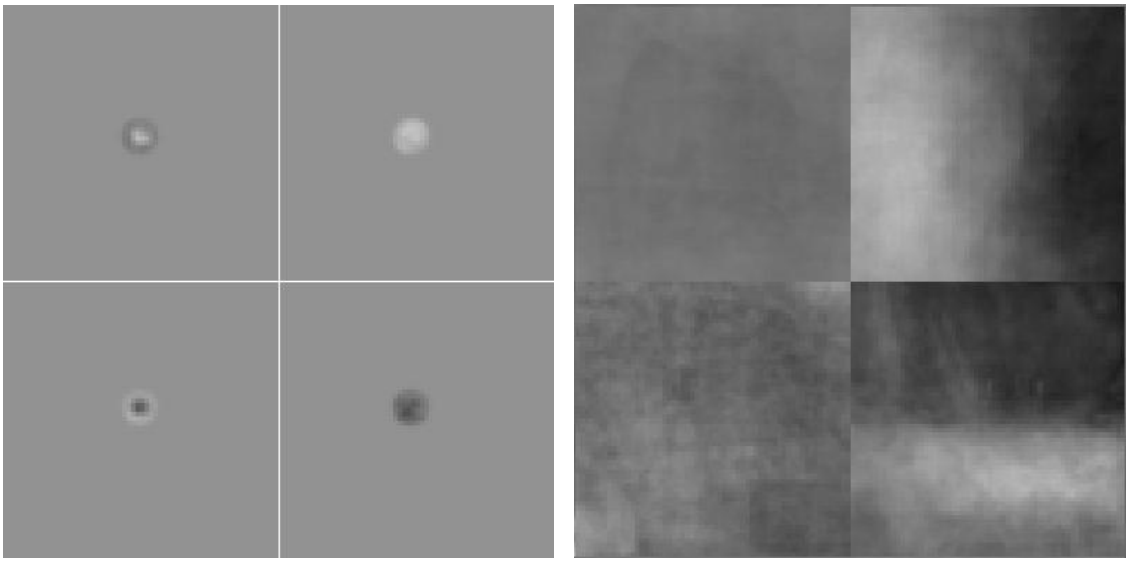
Retinal Wave Images

It is not surprising that the proximal receptive fields produced by retinal wave image training are very similar to those produced by natural image training. If we consider the sampling of natural images (we take small pieces of each image as input), we are most likely to capture input that consists mainly of uniform intensities and edges of all orientations, which is exactly the kind of information that retinal waves encode. This is a possible indication that even without input from the external environment, retinal waves can help train the visual system to adapt to the natural environment.

If we compare classic CD learning in Figure 5.5 (b) to that of Figure 5.4 (b), it is clear that the overall global composition of the images have a large effect on the type of receptive fields that are produced when we do not use a proximity function. If we instead contrast ProxCD retinal wave image learning in Figure 5.5 (a) to natural image learning in Figure 5.4 (a), we can see that proximal learning is more invariant to the exact nature of the image and focuses more on feature composition.

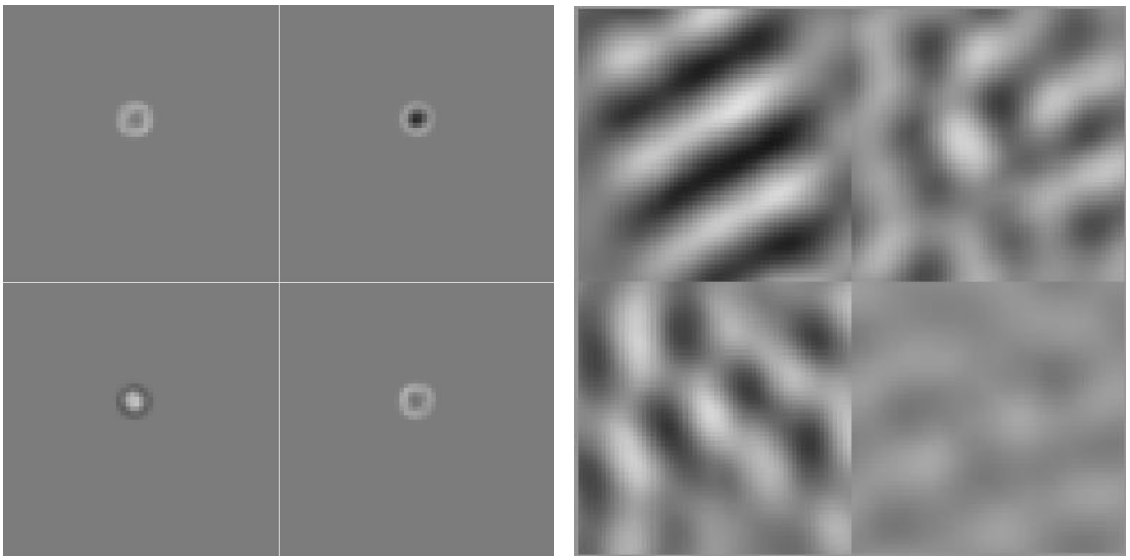
Constant, Zero, and Random Images

The receptive fields for the random input seem almost trivial (shown in Figure 5.6 (a)) as they appear to be small randomized regions themselves. However, there is something



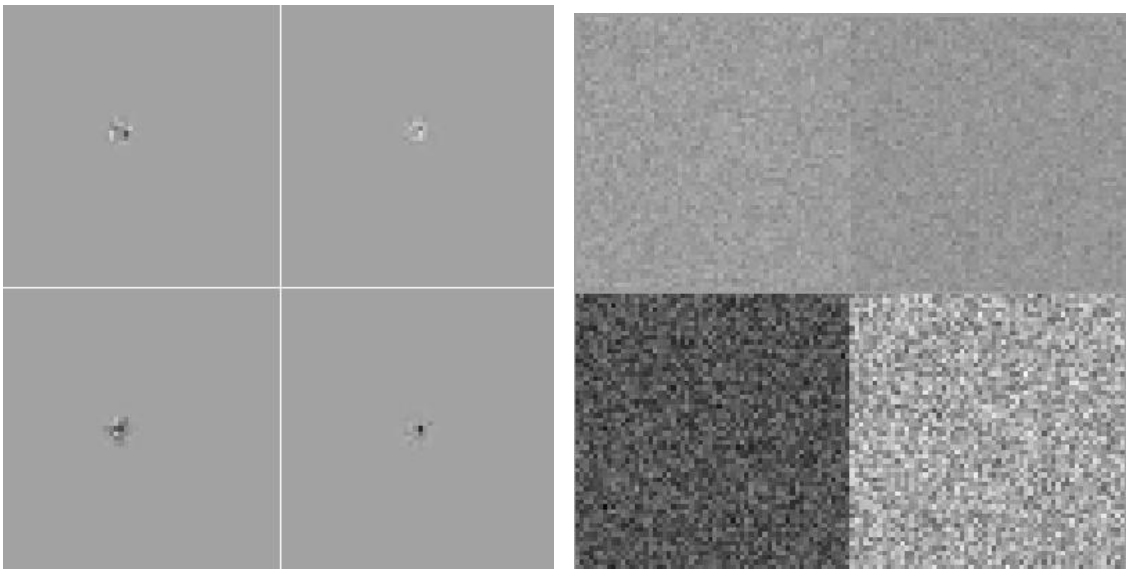
(a) ProxCD natural image learning (b) CD natural image learning

Figure 5.4: Sample of LGN layer receptive fields for natural image training. (a) shows localized, distinct on-centre and off-centre receptive fields, along with a few solid blobs we call intensity detectors. These centre-surround receptive fields occur for all neurons while using ProxCD learning. (b) shows the global type of receptive fields produced by a similar DBN only using the standard CD learning method (no proximity constraint).



(a) ProxCD retinal wave learning (b) CD retinal wave learning

Figure 5.5: Sample of LGN layer receptive fields for retinal wave training. Similar to natural image training, (a) shows localized centre-surround receptive fields and intensity detectors. (b) again shows global receptive fields produced by using the standard CD learning method.



(a) ProxCD random image learning

(b) CD random image learning

Figure 5.6: Sample of LGN layer receptive fields for orientation constrained and random images. ProxCD still shows localized receptive fields, however they appear to take on properties distinct to each domain. Standard DBN receptive fields are similar, only on a global scale.

to be noted when contrasting these receptive fields to those produced from training on the constant intensity images (the receptive fields are not shown, since they look nearly identical to natural or retinal wave training). If we observe the random image set and the constant image set, they are both devoid of any sort of significant patterns that may be worth encoding. However, the constant image training produces centre-surround receptive fields similar to the ones produced by training from natural images and retinal waves. One might imagine that the centre-surround pattern is generated by the Gaussian shape used to modulate learning, however we also see this occurring while using the plateau function (ie. box-car function, defined as being one within radius r, and zero elsewhere) on constant input. Since centre-surrounds are still produced after varying the proximity function, we can conclude it is not due to the Gaussian function properties.

As expected, the receptive fields for a zero-trained network are all local inhibitory areas. Any activity should be suppressed in order to produce an all-zero image.

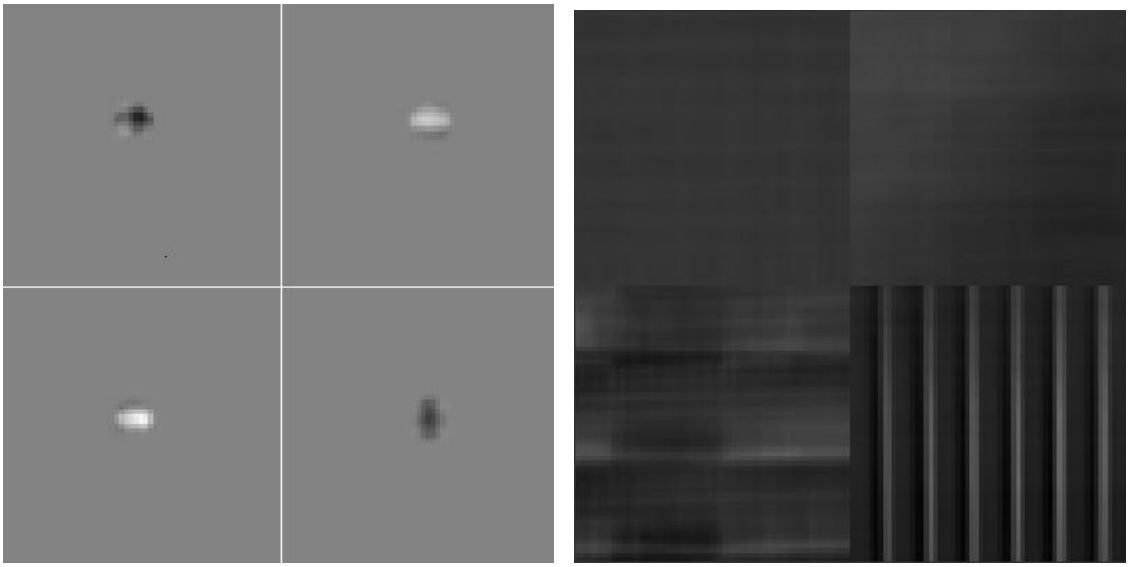
We can see from Figure 5.4 (a), Figure 5.5 (a), and to a lesser degree Figure 5.7 (a), that on-centre and off-centre cells tend to form regardless of what kind of training data we provide except for random noise and zero-intensity images. The receptive fields produced by random noise training are simply miniature patches of noise.

Orientation-constrained Images

Interestingly, when we observe the receptive fields for the orientation constrained universe in Figure 5.7, we find that while we produce centre-surround patterns, they are skewed towards the vertical and horizontal direction. These could be the ideal shape for a receptive field that is of a single orientation, and these begin to resemble little Gabor filters. We should expect a phenomenon like this to occur since we are reducing the amount of information that has to be encoded in each region.

Encoding Factors

Typically after running the learning algorithm for 20 to 30 epochs on any non-uniform training data, our receptive fields end up as a mixture of on-centre fields, off-centre fields,



(a) ProxCD ortho-edge image learning (b) CD ortho-edge image learning

Figure 5.7: Sample of LGN layer receptive fields for orientation constrained images. ProxCD shows localized receptive fields, however they take on properties distinct to the constrained domain. Receptive fields show an affinity towards encoding horizontal and vertical lines. Standard DBN receptive fields are similar, only on a global scale.

and blobs that we will denote as 'intensity detectors'. Intensity detectors are generally not present during the initial learning with a very low learning rate, and all the receptive fields are on/off-centre. However these do start to appear after prolonged learning. These intensity detectors likely handle encoding areas of uniform intensity, or adjust for varying brightness. In order to encode more detailed regions, the centre-surround receptive fields can be combined additively to create the contrast required for edges. When encoding resources are limited, it seems that encoding edges is done by combining multiple centre-surround receptive fields together, as depicted in Figure 5.8. This concept becomes important later when we look at V1 receptive fields (which are indeed a combination of LGN receptive fields).

The mean weights of each centre-surround receptive field for the networks tended to be near-zero compared to the individual weights themselves. The maximum and minimum magnitude of the weights for each receptive field ranged from ± 0.3 to ± 1.5 (0.3 to 1.5, though there were positive weights and negative weights), while the mean of these weights typically ranged between 0.015 and -0.015, with most falling within ± 0.010. The intensity detectors tended not to have zero means. This was typical of all the trained networks using ProxCD after 20 epochs of training. Since the centre-surround receptive fields are near zero-mean, the summation of multiple on/off-centre receptive fields does not affect the overall intensity of the image, only adding edge and difference information (high to low/low to high). This appears to further support the implicit edge encoding as a lot of neurons then contribute to changing local contrast without affecting regional intensity.

Centre surround receptive fields appear to be a simple way for a system to sparsely encode visual data where the data is mostly composed of solid intensity or low frequency data, with the occasional edge feature, given small concentric overlapping windows.

One of the major variables affecting the receptive fields is the width of the proximity function, or the Gaussian function's standard deviation. Bringing the width down to a pixel in size has the more obvious outcome of simply becoming a simple mapping where each node trivially handles only nodes that fall directly on top of it (there is no interaction or overlap between adjacent neurons' receptive fields). The other extreme is an extremely large window. We find that these tend to encode sections of receptive fields that are

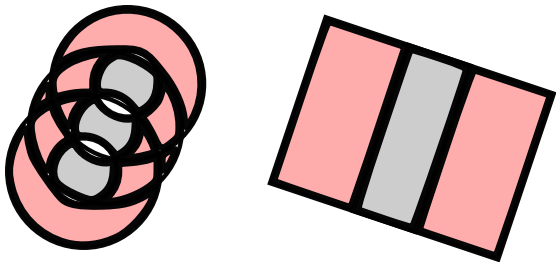


Figure 5.8: Edges can be encoded by combining multiple centre-surround cells at different positions. We can see in this image that the the summation of on-centre cells at various locations (on the left) is similar to a simple oriented bar (on the right).

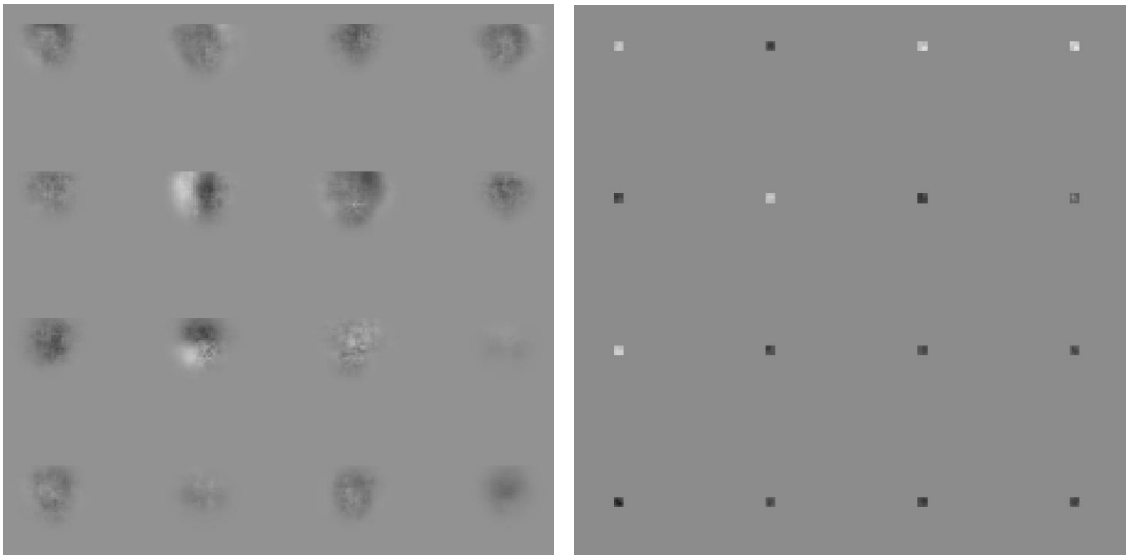
typically found with CD learning. The receptive fields from both of these cases can be seen in Figure 5.9. Interestingly, there seems to be a specific range of proximity function widths that yields on/off-centre fields. This width appears to be a function of how much overlap occurs between receptive fields.

The specifics of the Gaussian function is not the primary cause of the on/off-centre fields. Since producing these receptive fields from constant intensity training appeared as if it could be an artifact of the shape of the Gaussian, we also used a plateau, or box-car function that covered a similar retinotopic range. We found that by using this simplified function, the network yielded the same receptive fields.

It turns out that ProxCD does not require such a strict training schedule. It is capable of producing comparable results regardless of using a small learning rate at the start. This can likely be attributed to the smaller search space that each individual neuron is constrained to. On the other hand, standard CD learning is more sensitive to the learning schedule and tends to be more suited to using the schedule that was listed earlier. It would appear that ProxCD is more adaptable to a relaxed training regiment.

5.2.4 LGN Reconstructions

We can propagate the input states through to the LGN layer and back to the input to regenerate the expected input. This process allows us to gauge how well the network is



(a) Large ProxCD standard deviation of 5.0 (b) Small ProxCD standard deviation of 0.1

Figure 5.9: ProxCD with various Gaussian function widths trained on natural images. ProxCD with a large enough function generates receptive fields that resemble miniature CD receptive fields. With a tiny enough function, ProxCD simply attempts to map pixels.

able to encode and internally represent various scenes.

As a note, the visual fidelity of the reconstructions is very low compared to the original input. This can be largely attributed to the significant bottleneck that occurs at the LGN layer. The ratio of possible states in the input layer to the LGN layer is $q_1^{(60\times60)}$ vs $q_2^{(30\times30)}$ where q_1 is the number of possible states each neuron can take in the input layer, and q_2 is the same for the LGN layer. If $q_1 = q_2$ then the ratio is $4 : 1$. However, q_2 is 2 (neurons are binary in the LGN layer), and q_1 is generally taken to be greater than 2. In our experiments, our input consists of a 60×60 pixel image, where each pixel can take on 256 different intensities. This has to be fed through the 30×30 neuron LGN layer, where each neuron is binary (2 different states). The resulting ratio is around $10^{8400} : 1$ in terms of possible state combinations. If we used $q_1 = 16$ intensities instead, then the ratio would be around $10^{4000} : 1$.

It should seem obvious, but networks that have been trained on specific input will generally be good at reconstructing said input. Wave-trained networks surpass other networks in reconstructing retinal waves. The networks trained on natural images, varying-intensity, and ortho-constrained images all reconstruct natural images at the same relative quality (see errors in Figures 5.10 and 5.11). It is not surprising that the centre-surround receptive fields of these networks are very similar to each other. The network trained on retinal waves is significantly worse at reconstructing natural images, and tends to accentuate hard edges, or to over-intensify regions. It may be that the W-trained network tries to interpret the input as a wave, thus producing the expected low-frequency local wave pattern of dark to very bright. However, more tests are needed to verify this hypothesis.

Homogeneous zero-intensity training and random noise training yield the least interesting reconstructions. These networks generate a blank image or nearly complete noise (respectively) when attempting to reconstruct anything as seen on the right of Figures 5.10 and 5.11. The receptive fields of these networks would have suggested very little orderly information is being encoded to begin with.

Networks trained on either natural images or greyscale images appear to have the most average responses (no particular strong or weak points) in terms of their ability to reconstruct any given input. In a majority of the test images, the ortho-constrained

networks fall slightly behind N-trained and G-trained networks largely due to the O-trained networks affinity towards the cardinal axes. Examples of reconstructions from various networks are in Figures 5.10 and 5.11.

Reconstructions of the highest frequency retinal-wave images are poor in all cases. These images contain eight full wave periods. It is possible to attribute this to the compression factor, as it requires roughly three properly positioned centre-surround receptive fields to produce one period of the wave (in a single direction). Again, further tests are required to make conclusive claims.

5.3 Model V1 Cortical Layer Training

5.3.1 Implementation

Building on the model outlined in section 5.2, we can extend the network to another layer, thus creating a deep representation of the visual system. Like the previous layer, our implementation of the networks consisted of setting layer and proximity sizing, as well as learning parameters. The main difference in dealing with our V1 layer is how we visualize it.

Network Architecture

The structure of these connections is determined by the size of the LGN layer of neurons and the V1 layer of neurons. At this point, the size of the LGN layer has already been determined so it is simply a matter of choosing the size of the V1 layer. Again based on our understanding of the neurons in the LGN and the V1 area in the visual cortex, we know that there are many more V1 neurons in the visual cortex as compared to the LGN. Thus our strategy was to expand the number of neurons in the new layer compared to the LGN layer. We found that empirically, much like the last layers, the ratio of expansion was not terribly important.

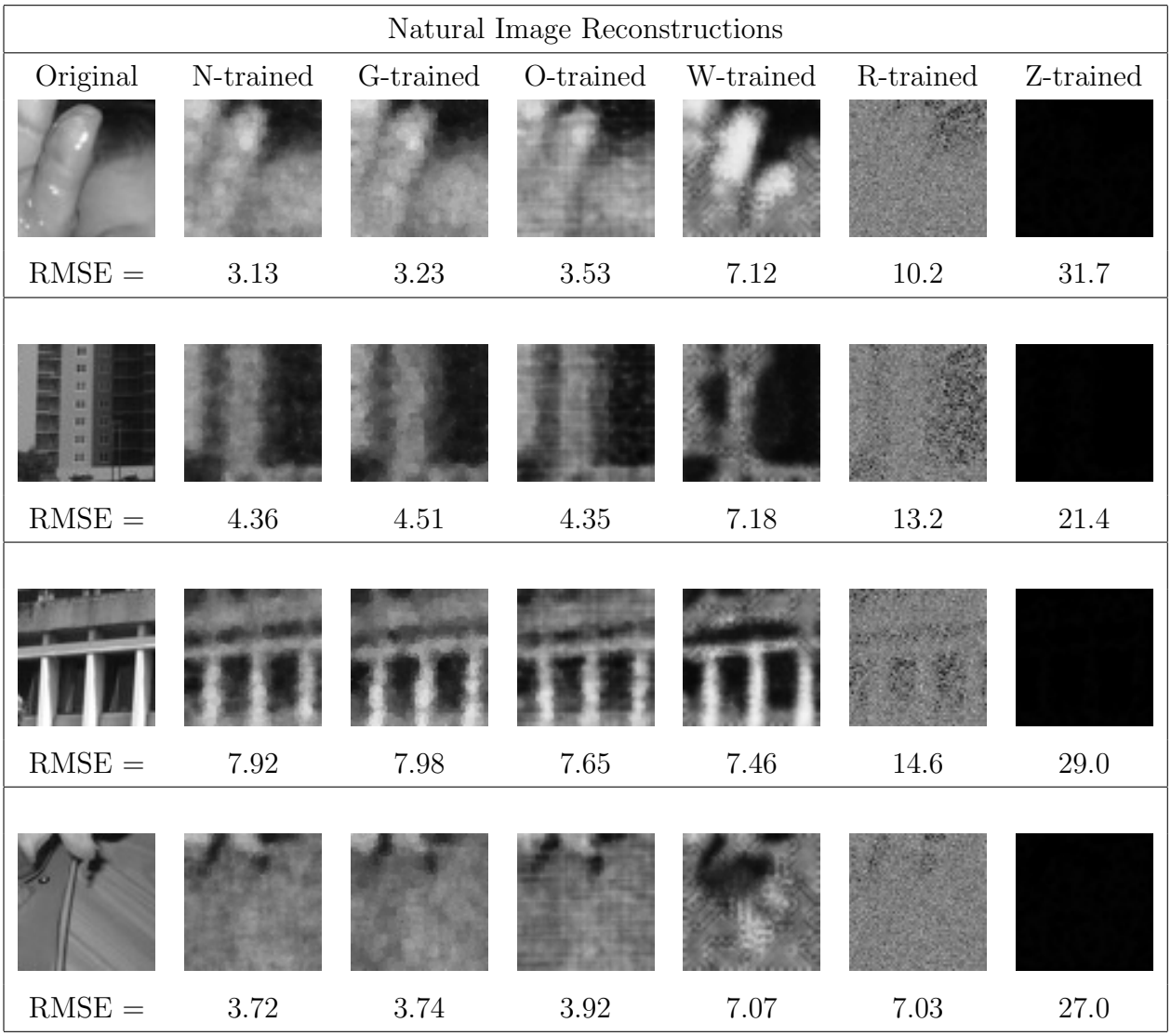


Figure 5.10: Natural image LGN reconstructions. Note the W-trained network's improved ability to reconstruct the third image down compared to the others. It would appear that this image favours wave-like reconstruction due to its content.

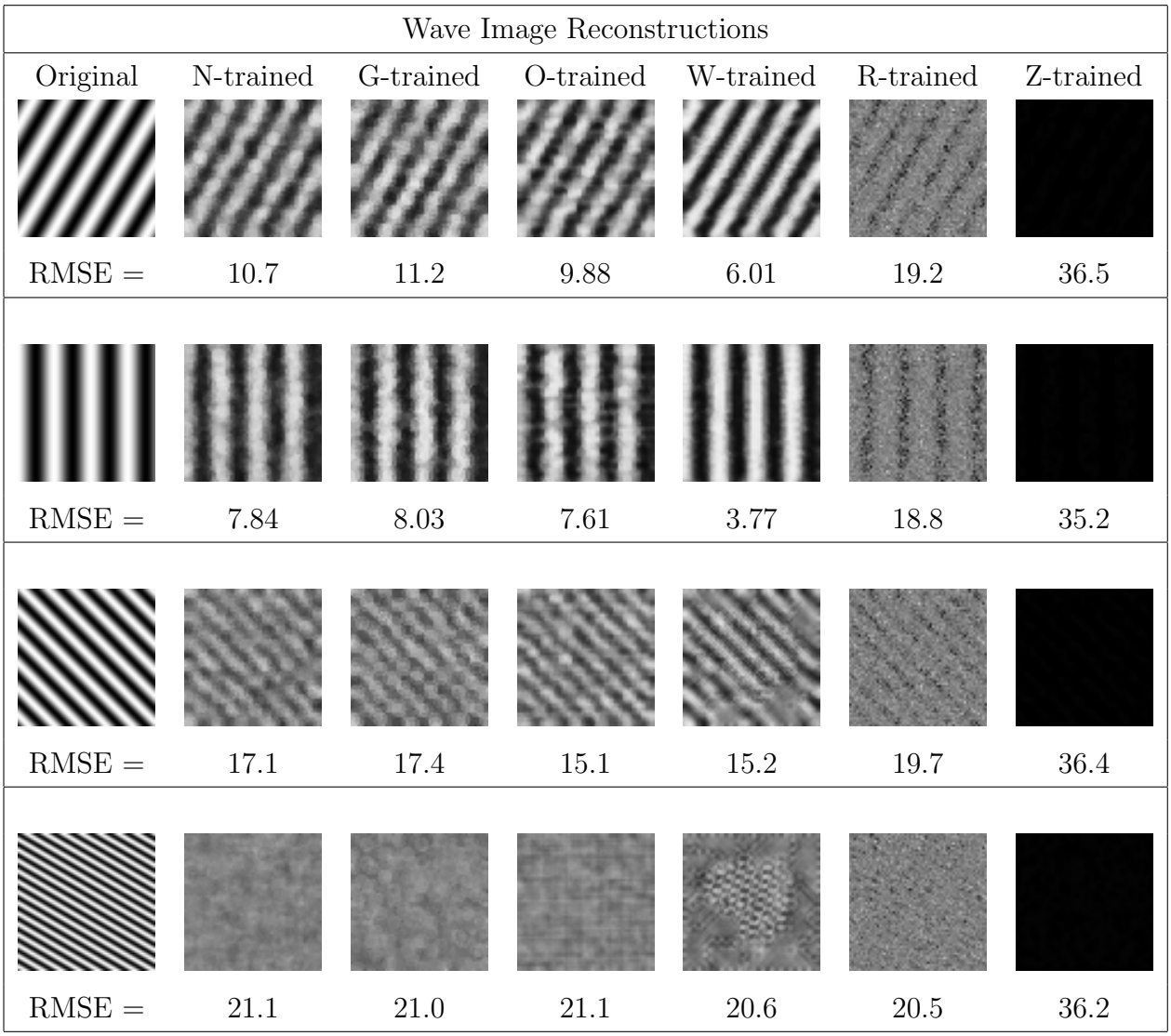


Figure 5.11: Retinal wave LGN reconstructions. The fourth image down indicates that there is likely a resolution limitation imposed by the amount of neurons that were used for encoding.

For the deep networks displayed in this thesis, we used a V1 layer composed of 14400 neurons in a 120×120 matrix. We have additional results from a network utilizing 3600 neurons in a 60×60 matrix, however all results discussed in this thesis were from the 14400 neuron network unless otherwise specified. The new layer, like the previous hidden layer, was treated as a binary PRBM.

Training Duration and Rate

The Training schedule of this layer mimicked the approach used for the first layer. To reiterate, the schedule used a learning rate of $s = 0.05$ for the first 5 epochs, $s = 0.50$ for the next 5 epochs, and $s = 1.00$ for further training.

Proximity Window

While at this stage, we chose to use the same proximity window size as the previous layer (a standard deviation of 1/60 the length of the V1 neural space), this actually means that the V1 layer has an expanded receptive field compared to the LGN layer. Each V1 neuron draws heavily from its nearby LGN neurons, which in turn draw heavily from their nearby inputs. This cascading effect causes an outward expansion of the input nodes captured by deeper layers. Due to the inequality in the number of neurons between the V1 layer and the LGN layer, our V1 proximity function has a standard deviation of 1/30 the length of the LGN neural space (equivalent to 1/60 the length of the V1 neural space).

5.3.2 Visualization

A weighted average of the first hidden layer's receptive fields according to the second-layer weights gives a picture of what, on average, might excite a given V1 neuron. However it ignores the non-linearity built into the process (at the LGN). Additionally, it does not quite give a clear picture as to what combinations of LGN neurons activate the cell. One extreme example of this is to suppose a neuron draws equally from two receptive fields. One such field is a left-to-right gradient, and the other one is a right-to-left gradient. If

we sum these two fields linearly according to the weights we will ultimately get a receptive field that is empty, even though the neuron responds to two distinct patterns.

Instead of trying to imagine what this neuron prefers to see, we can focus more on the properties of V1 neurons that we would expect to see if this were a good replication of the human visual system. As we have discussed earlier, neurons in this layer have a wide variety of receptive fields. Simple cells provide the most straight forward analysis, and the most applicable for the situation. We will discuss the conditions required for complex cells and global effects in further detail at the end of this section.

We can compute the V1 activation for an input image of a chosen signal (specifying the frequency and orientation of the stripes or waves). We can also combine these V1 activation maps to produce a bigger picture of overall preferences.

5.3.3 V1 Generated Receptive Fields

Receptive Fields

We can see in Figure 5.12 the V1 receptive fields generated by the ProxCD method. It is common to find other visual models produce V1 receptive fields that look like perfect localized Gabor functions [22, 18]. While Gabors are mathematically very elegant and ideal for representing oriented edges, with frequency and phase modulation, it would seem premature to assume this is the ultimate way that information is biologically represented in V1 receptive fields.

This idea that V1 receptive fields should ideally look like Gabor basis functions is not unfounded. Gabors are indeed a very nice, mathematical way of representing the information that can be captured by V1 receptive fields, and their relationships to visual responses have been investigated in papers such as [15] and [24]. Even so, some vision researchers have taken a different approach. The authors of [3] express this in the following:

> Indeed, the resilience of human recognition performance to image degradations suggests that image measurements underlying recognition can survive significant reductions in reconstruction quality. Extracting measurements that are

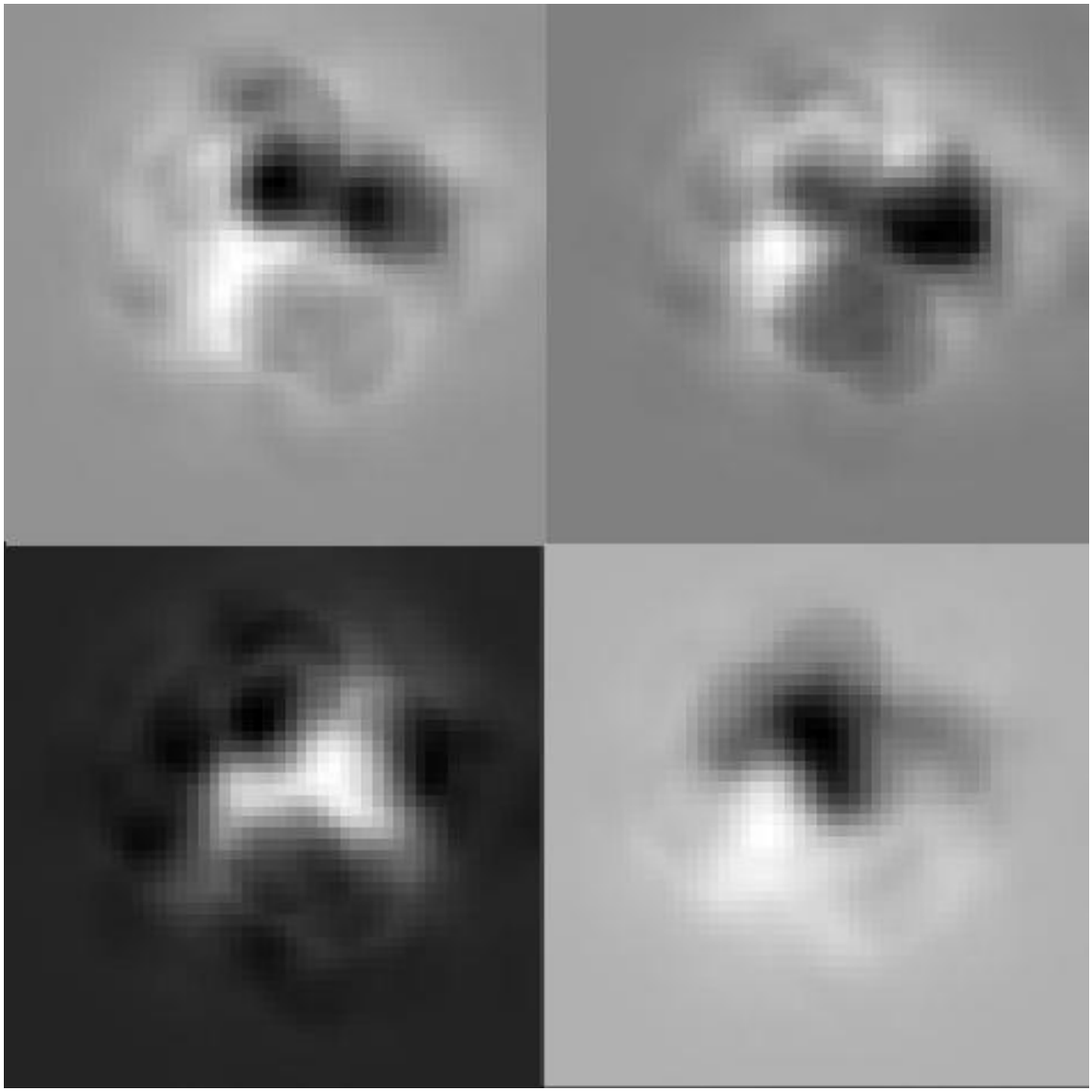


Figure 5.12: Sample of V1 receptive fields produced by ProxCD learning on natural images (each normalized so that its minimum is black and the maximum is white). Four receptive fields from the same area shown from the V1 neuron layer. Receptive fields were produced by taking a linear combination of the previous layer based on the weight to each LGN neuron. Note that regions of the receptive fields do seem to take on edge detecting properties.

stable against ecologically relevant transformations of an object (lighting and pose, for example) is a constraint that might result in qualitatively different receptive field structures from the ones that support high-fidelity reconstruction.

The visualization procedure is a slight simplification. It takes each receptive field of every LGN neuron that the V1 neuron is connected to, and combines them linearly according to the weights to the V1 layer. This ignores the non-linearity, however it still gives us a glimpse into what kind of input is more likely to activate the neuron. We will however be using a different approach to visualize V1 neural representations.

Orientation and Activation Maps

In order to bypass the difficulty in visualizing the receptive fields themselves, we instead turn to other properties of the neurons. As we know from various studies such as that done by Bonhoeffer et al. [6], as well as experiments outlined in chapter 2, the V1 cortical region largely responds to stimuli that take the form of oriented edges and lines. In order to determine which orientation a neuron prefers, we take a method used to visualize data collected from a physiological study, and modify it to work for us.

We begin by producing a series of oriented sine waves for visual input. The orientation along with frequency and phase of these sine waves are stepped at regular intervals such that each wave orientation contains a wide variation in frequency and phase data. Each orientation test image is fed into the model, and we collect the magnitude of the response for each neuron. After testing the full range of frequencies and phases, the strongest response is recorded for each orientation. The responses to the different orientation gratings are combined into a single orientation preference using a vector sum. For each orientation, the neuron's activation is used to represent the magnitude of a vector specified in the direction of the orientation grating. The angles are doubled so that orthogonal angles oppose each other, and opposite angles compliment each other. Once these vectors are added, the resulting vector determines the orientation preference of a given neuron. This process is outlined visually in Figure 5.13. This was the same procedure performed by Bonhoeffer in his study on the cat's visual system [6]. Examples of this type of visualization are seen

in Figure 5.14. The activation maps are normalized so that these maps show the relative neural activity scaled by the activity over all orientations.

The relative activation-sensitivity refers to how strongly a neuron reacts to a specific oriented stimuli. A low sensitivity means that the neuron does not respond to a specific orientation compared to other orientations. Likewise, a high sensitivity means a neuron will give a strong response when presented a certain orientation when compared to the response from other orientations. Across all orientations and all neurons, the mean relative activation-sensitivity hovered around 0.080. This is in comparison to the maximum, which varied, but generally remained in the range of 0.2 to 0.4. The minimum varied as well, but generally remained under 0.01. Figure 5.15 gives an example of one orientation's activation sensitivity (the specific 'spike' pattern is typical of most networks). A majority of the neurons did not reach such strong response levels however. In order to avoid simply displaying the few strongest neural responses, activation maps were displayed using a range of 5% of the mean. Everything above or below the range would appear at the maximum or minimum intensity respectively (note this is only for visualization purposes).

Interestingly, when we produce a combined orientation map using the above technique, we find the emerging 'pinwheel' patterns. These pinwheels are clusters of smoothly varying orientations, converging on a central point. In many earlier studies, the emergence of 'pinwheel' patterns in orientation preference was a largely cited phenomenon, however this trend is less prevalent now. We find that similar orientations have a tendency to clump together instead of dispersing randomly. Orthogonal orientations tend to activate disjoint areas, as shown in Figure 5.17.

As a note of importance, receptive fields at the LGN layer show no significant response to orientation testing. In general, nearly all the neurons in the LGN layer will have similar responses to any orientation falling within its receptive field. This makes sense when we observe the shape of the receptive field, as centre-surround patterns have no direction associated with them. Biologically, the LGN neurons do not give any special responses to edges either.

Non-proximally trained networks also fail to produce orientation maps. Figure 5.16 depicts a typical orientation map production from a CD trained network.

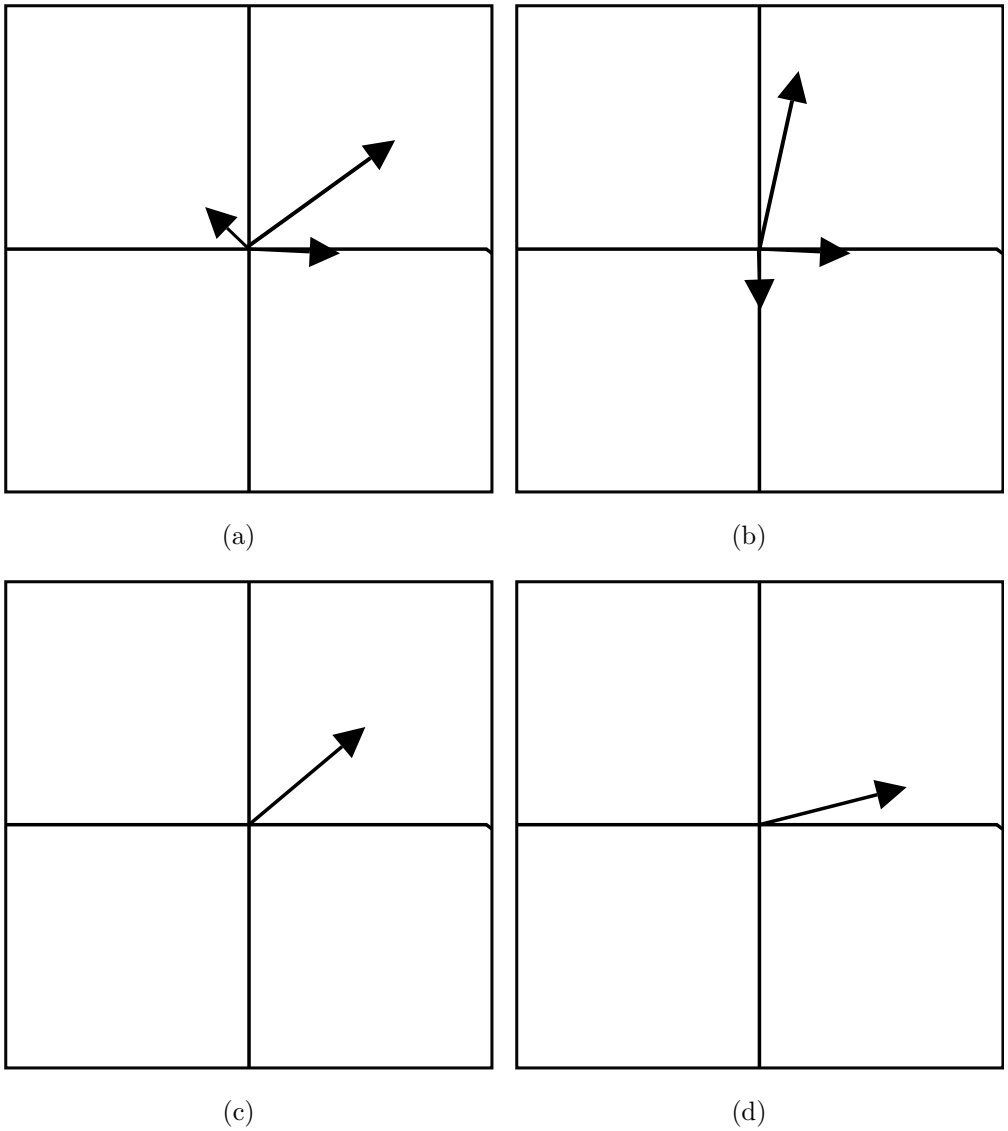


Figure 5.13: A diagram of how the activation maps are produced. (a) data on each orientation is collected as a vector. The orientation determines the direction (0 to 179 degrees) and the relative magnitude of the response determines the length. (b) angles are doubled, so that orthogonal angles oppose each other. (c) the vectors are then added together in this new space. (d) the angle of the resulting vector is halved to get the original direction.

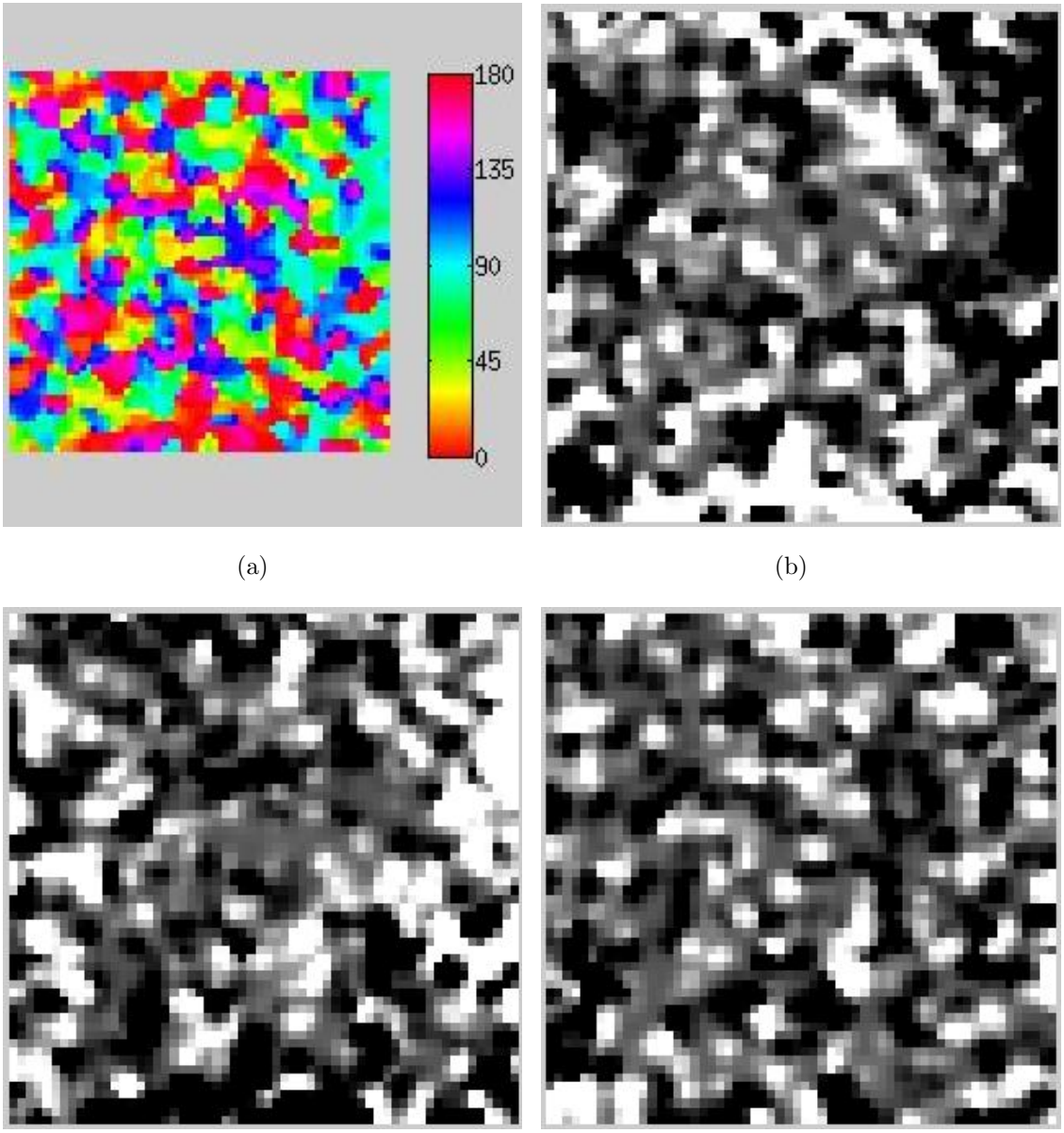


Figure 5.14: Blurred activation maps from V1 trained on 'retinal waves'. (a) shows an orientation map where each direction has been vectorially combined to show the orientation preference for each neuron (each neuron being represented by a pixel of colour). (b) (c) and (d) show individual activation maps of horizontal, vertical, and 45 degree affinity respectively.

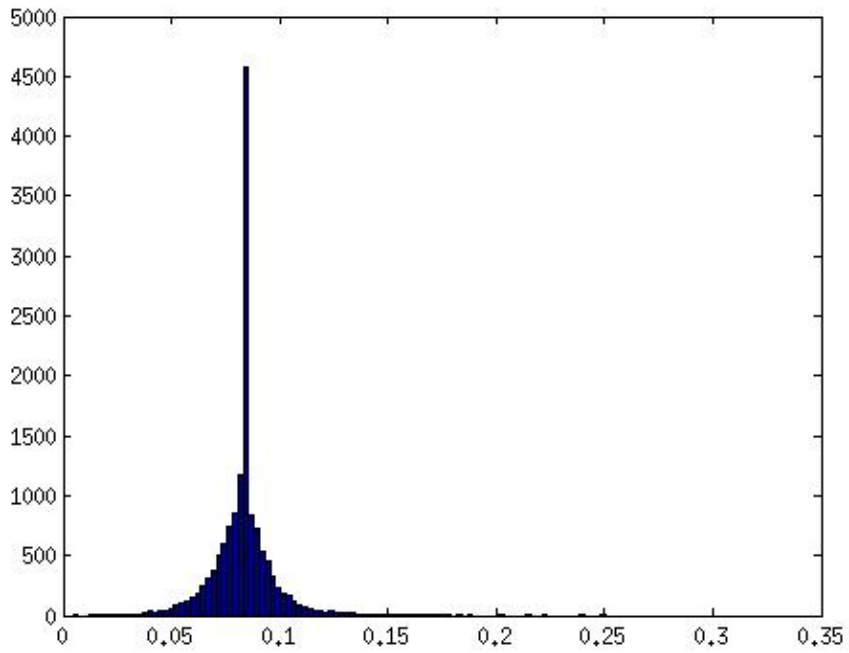


Figure 5.15: Activation sensitivity histogram of N-trained network to horizontal edge stimuli. Height of bars indicate the amount of neurons that give a certain response. Responses are normalized to the total response of all the neurons.

Orientation mapping is of course only a single measurement for V1 cells. However, it is a dominant and well-understood property. The number of neurons in the V1 layer had little bearing on the resulting orientation mapping. The smaller resolution orientation map simply appeared to be a blurrier version of the higher resolution orientation map (similar striations and clustering).

Blurring Orientation Maps

Blurring is a phenomenon that is likely to occur in studies created using optical imaging of intrinsic signals, since their optical imaging device would have an inherent point-spread function causing blurring. We blur the orientation maps slightly by filtering each individual

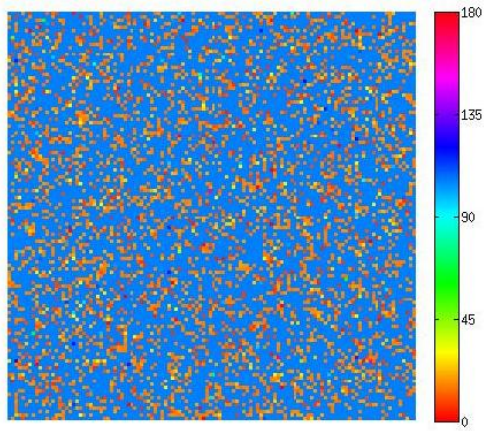


Figure 5.16: CD network orientation map is overwhelmingly tuned to a single orientation.

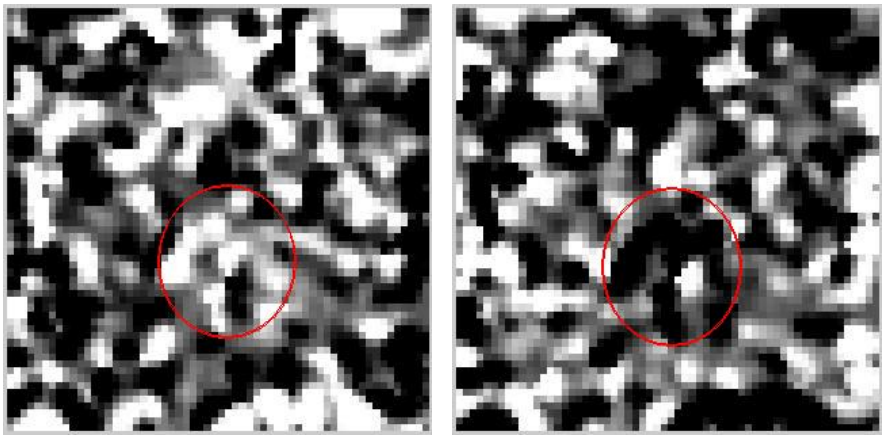
activation map with a Gaussian filter before combining them vectorially. Unless otherwise specified, the function used to blur the activation maps was a 5×5 pixel windowed Gaussian with a standard deviation of 1.5 nodes.

The blurring was applied to each orientation before combining them and taking the maximum. Thus, each orientation was allowed to 'bleed' into its neighbouring area, affecting the strength of that specified orientation, but no others. Naturally, this process smooths out random bumps or perturbations if an area of neurons favours a specific orientation.

Pinwheels also appear if one adequately blurs and combines random activation maps. However, if we observe a non-blurred orientation map (as shown in Figure 5.18) it indeed produces similar features, though not as smoothly varying. As a result, it is safe to conclude that the results are being derived from the data itself. Typically we use only small amounts of blurring and only for illustrative purposes (all results are checked against non-blurred maps to ensure that they are being accurately represented). Figure 5.19 depicts another such orientation mapping produced by imagining the V1 of young kittens [7].

Natural Image and Retinal Wave Training

Natural images and retinal waves seem to be very similar in their V1 representation. The receptive fields generated by ProxCD do exhibit qualities that could capture localized edges



(a) Vertical sensitivity (b) Horizontal sensitivity

Figure 5.17: Two orthogonal (vertical and horizontal) activation maps are compared. The main focus of this comparison is that in general, orthogonal orientations appear to occupy disjoint regions in V1. The red circle indicates one region where this occurs (the M shaped region).

(Figure 5.12).

If we look at orientation distribution for natural image training in Figure 5.20, then there appears to be a moderate bias towards the vertical and horizontal. One might expect such a bias, as many scenes from the natural and modern world have a prevalence of vertical and horizontal edges. Pictures are also generally taken facing the horizon. However if it were a case of simply being caused by the images then why do waves have vertical and horizontal preference (see Figure 5.21)?

One explanation for axis aligned bias is the 'oblique effect'. The oblique effect is such that orientation discrimination of the visual system is better around the cardinal axes compared to oblique. One theory put forth is that there are fewer neurons tuned to oblique orientations [19]. However if this is the case, our networks demonstrate that regardless of the orientation of the input, we still have a significant amount of neurons that are tuned to the cardinal axes. Another possible reason is that due to our network's construct using a neural lattice (and that everything is pixellized), vertical and horizontal edges are the

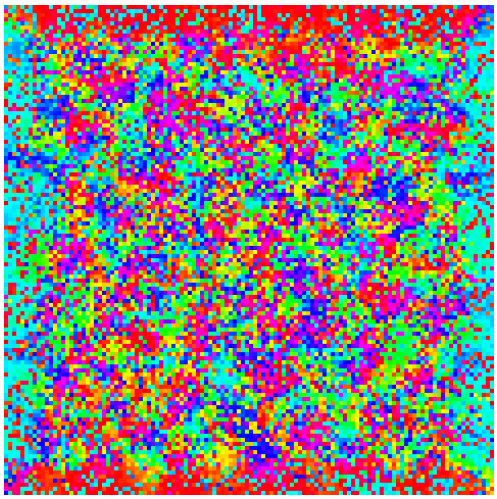


Figure 5.18: Non-blurred orientation map. Orientation clustering is still present, however regions are generally more noisy and less smoothly varying. Neurons near the sides that take on a single orientation preference are likely due to image edge effects.

least ambiguous edges to represent. Additional testing would be required to determine the exact nature of this effect.

Random, Zero, and Constant Image Training

Compared to the other training sets, random training tends to have the most 'averaged' response characteristics. Figure 5.22 demonstrates the very orientation-neutral nature of such a network. Even though we can produce an orientation mapping, most of the neural responses hover around the mean value (the mean is consistent across the various training types). Surprisingly, random and zero-trained networks have some of the 'cleaner' orientation maps (most well defined and segregated into orientation clusters) which may suggest that orientation clustering is more so an artifact of proximal training and is merely modulated by the trained images. Figure 5.23 depicts an orientation map produced by a zero-trained network. It is unclear as to what this exactly means for such a network, as it is incapable of reproducing any sort of input even though it is producing this orientation map.

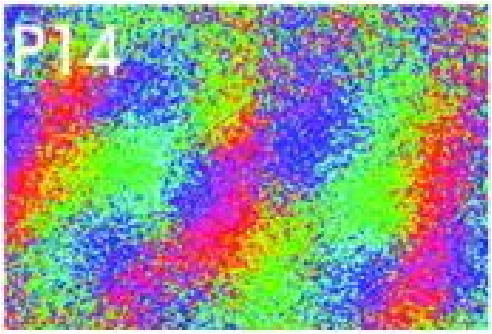


Figure 5.19: An activation map produced using intrinsic optical signals by probing a kitten at post-natal day 14.

Orientation-Constrained Image Training

What we find from our orientation-constrained image data is that our training produces a network of neurons that has a very strong affinity towards horizontal and vertical edge preferences. This is expected, as fewer neurons need to compete to encode a wide variety of angles. Additionally, by further training the network on a different set of data, such as natural images, the neurons will begin to adjust; they grow and prune connection weights, adjusting the amount of neurons that have preference towards certain orientations. Figure 5.24 shows the changes in selectivity after 30 epochs of retraining.

Orientation selectivity for angles that are not vertical or horizontal does not disappear, much like what occurs in the study done by Blakemore [5]. More so, as per Hebbian principles, neurons that encode more common stimuli are able to proliferate their connections. This is a property that is common to both the ProxCD and CD learning methods.

Global Phenomena and Complex Cells

As one would expect, constricting the learning function to a small area precludes the ability to find wildly variable sized receptive fields in the first few physical layers. A lot of papers (such as the studies listed in chapter 2 [22, 13]) tend to make the distinction that each layer of a multi-layered network represents one portion of the human visual system. In fact, for

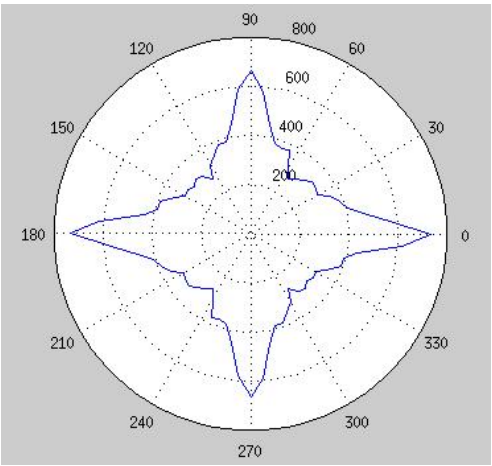


Figure 5.20: A histogram of the distribution of neural affinity for natural image training. Orientation preference is biased towards cardinal axes.

our first layers we make this distinction as well, as the retinal input, LGN, and V1 sections all have analogous distinct counterparts in our network model. However, logically, each layer is simply a set of neurons that must learn from the previous set of neurons (and also possibly have different proximity functions and/or learning rates). Since the receptive field of the input grows with each subsequent layer (as long as we choose not to further constrict the proximity window), some neurons may begin to encode more global properties of the input in subsequent layers.

As the receptive field of a cell grows, it could begin to encode many variations of the previous layer. A cell in the next layer could encode local edges independent of orientation, or local edges independent of phase, or the occurrence of multiple edges with matching orientation, depending on the size of the proximity function. The phase-independent edge detectors are typically labeled complex cells. If we wish to achieve something of this nature, it is entirely possible to give some neurons global receptive fields. These neurons would exhibit global phenomena. However, the notion of separating neurons into two classes with different weighting schemes seems to over-complicate the model in terms of biological simplicity and local interactions. Also, as iterated earlier, the idea of 'super neurons' that are connected to all other neurons does not follow any biologically plausible scheme.

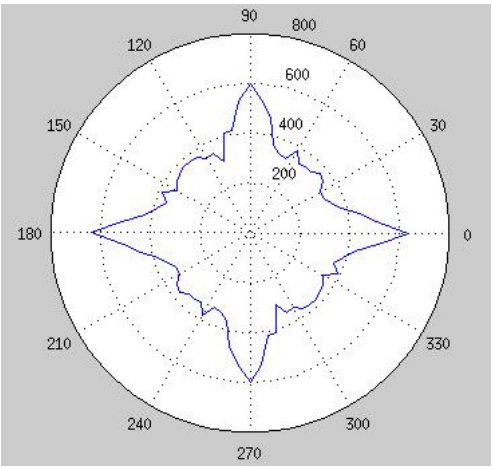


Figure 5.21: A histogram of the distribution of neural affinity for retinal-wave image training.

5.3.4 V1 Reconstructions

As with the LGN layer, we can use the recognition weights to propagate input to the V1 layer, and then use the generative weights to propagate the states back to find out what kind of input the network is encoding at the V1 layer. In general, when reconstructing using the additional V1 layer the quality of the resulting image is reduced in comparison to using just the LGN layer. By adding another proximity layer, the input data is transformed not only once, but twice. The generative pass then has to reconstruct through both layers. The result of this is a more blurry representation. Figures 5.25 and 5.26 demonstrate various reconstructions based on differing conditions.

Again image fidelity still suffers due to the bottleneck that occurs at the LGN layer. While the addition of the V1 layer can help to augment which LGN neurons fire in tandem, it does not alleviate the bottleneck during training (as each layer is trained separately).

As mentioned earlier, random noise and zero training both produce "cleaner" orientation maps. What the reconstructions show us however, is that even though the random and zero trained networks are producing this orientation map, the networks themselves are not doing anything very useful, and are actually encoding very little.

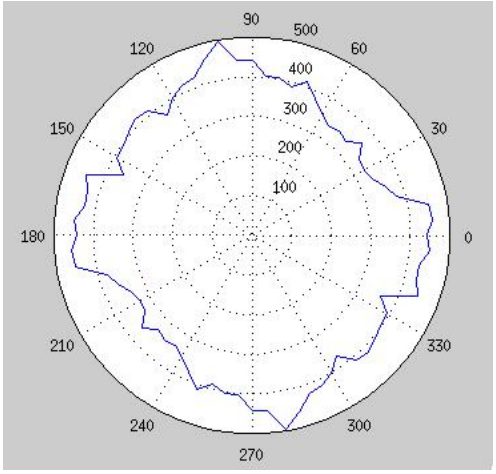


Figure 5.22: A histogram of the distribution of neural affinity for random training. Orientation preference is distributed relatively evenly for all orientations.

5.4 Non-Homogeneous Input Images

One of the interesting properties of proximal training is that by using local windows to generate the receptive fields, we can easily reconstruct spliced (images created by composing different parts of separate images) or incomplete images (see Figure 5.27). We see that ProxCD demonstrates an increase in robustness to encoding by reducing the complexity of the required training stimuli. For instance, even though the W-trained ProxCD network in Figure 5.27 has only been trained on complete waves, it is able to internally represent local pockets of separate waves that form a completely new image. The error rates for reconstruction are comparable to complete image reconstructions such as in Figure 5.26.

Standard CD learning is unable to process these reconstructions accurately as shown in Figure 5.28. As we have seen earlier, the neurons trained in these networks encode global features. Standard CD focuses on trying to adopt the image into one of the global patterns it was trained on previously. It is important to note that we have also seen sparse coding techniques that can produce localized feature detectors and thus accommodate for partial images. However, in our case, we have shown that the simplest training set of grey-scale,homogeneous images can be used by a proximal network to achieve local receptive

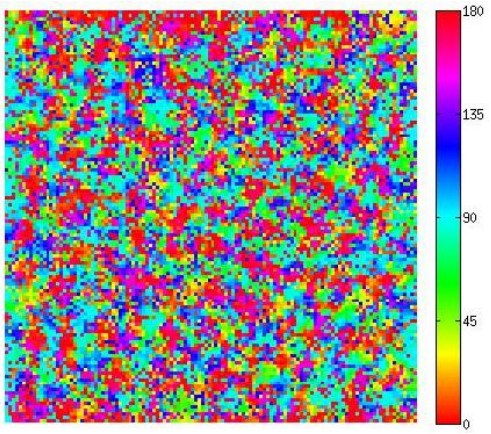
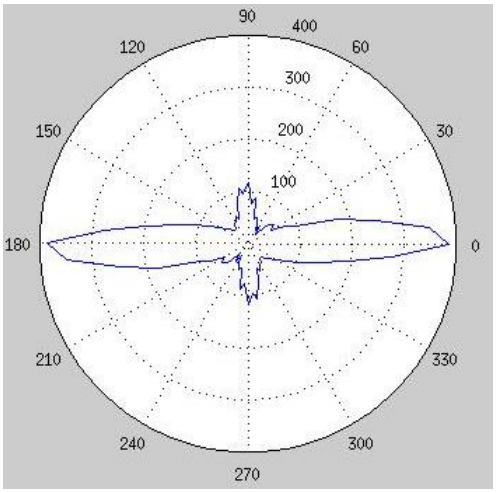


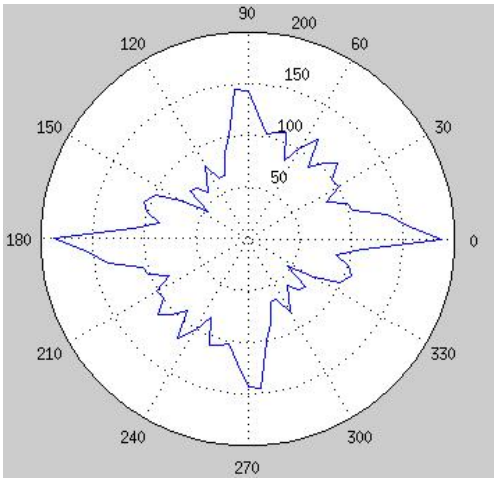
Figure 5.23: Zero-trained network orientation map appears to be well segregated and clustered.

fields capable of representing partial images (Figure 5.27). This is in contrast to other techniques that we have outlined earlier which would, at the very least, require input containing localized, detailed edge or wave information of all orientations.

Optical illusions often occur when we are fooled into visually misinterpreting something that we are seeing. How do we make this mistake? Our visual systems are trained to represent (encode) the typical images that we experience. When we view an image that does not conform to our internal priors, then we do not have an appropriate internal representation. Thus, in some cases of optical illusions, our visual system misrepresents the input. In a way, this is mimicked by the trained networks. For example, when our network trained on retinal waves observes this natural image of lines on a field, it interprets this as part of a wave and significantly darkens the area between stripes (see Figure 5.29). In this simple case we have only gone to the V1 layer. Of course, interesting optical illusions that we might be more familiar with typically involve a misinterpretation of more abstract image properties: lengths, motion, colour, etc... But this simple case illustrates how economical representation strategies can become confounded by unusual input, leading to inaccurate 'perception'.



(a) Initial training on orientation-constrained images



(b) Further training on natural images

Figure 5.24: Histogram plots for orientation selectivity in V1 layer. (a) shows the histogram for the number of neurons responding to different angles after an initial 30 epochs of training on orientation-constrained images. (b) shows the histogram after 30 epochs of further training on natural images.

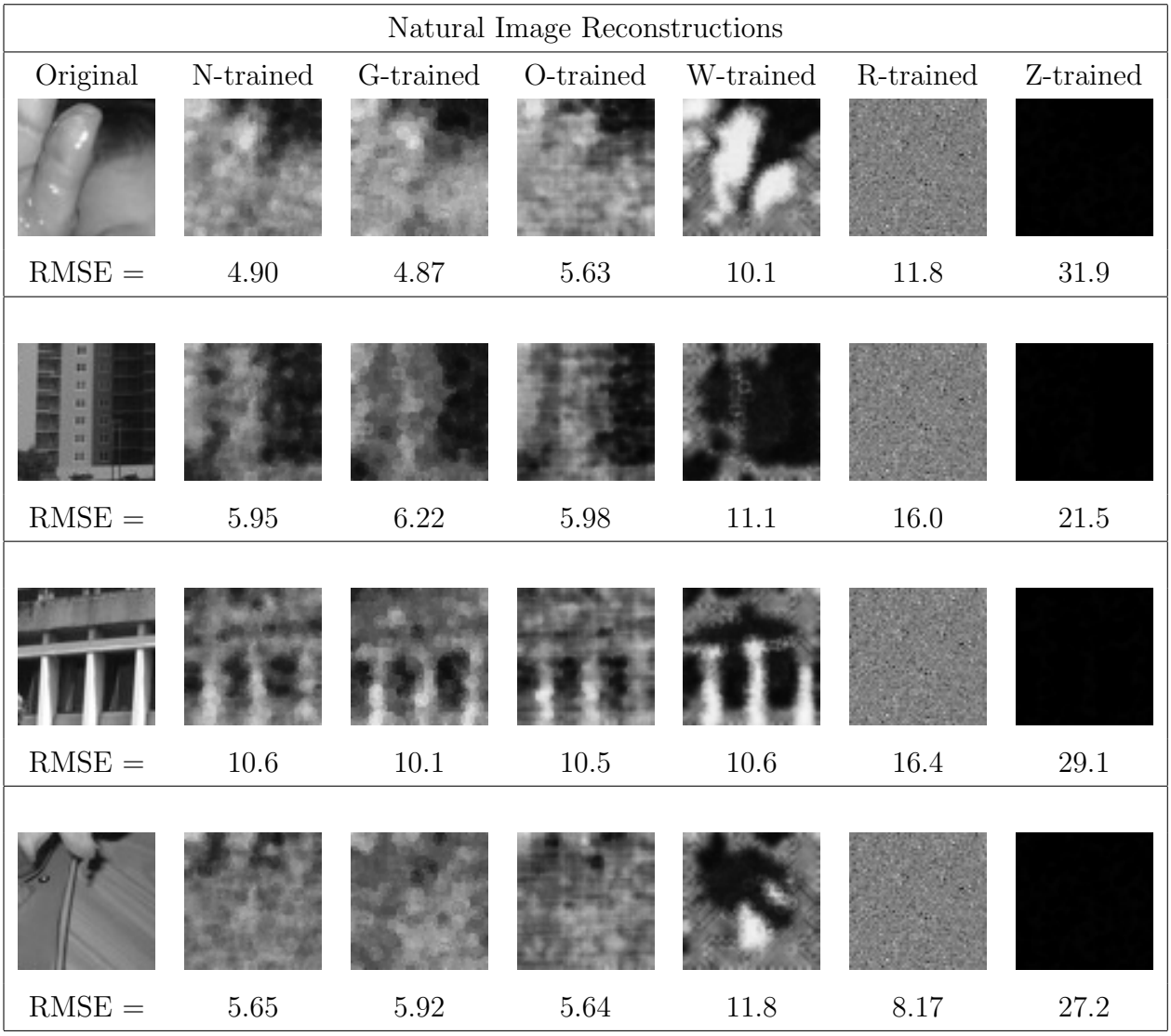


Figure 5.25: Natural image V1 reconstructions.

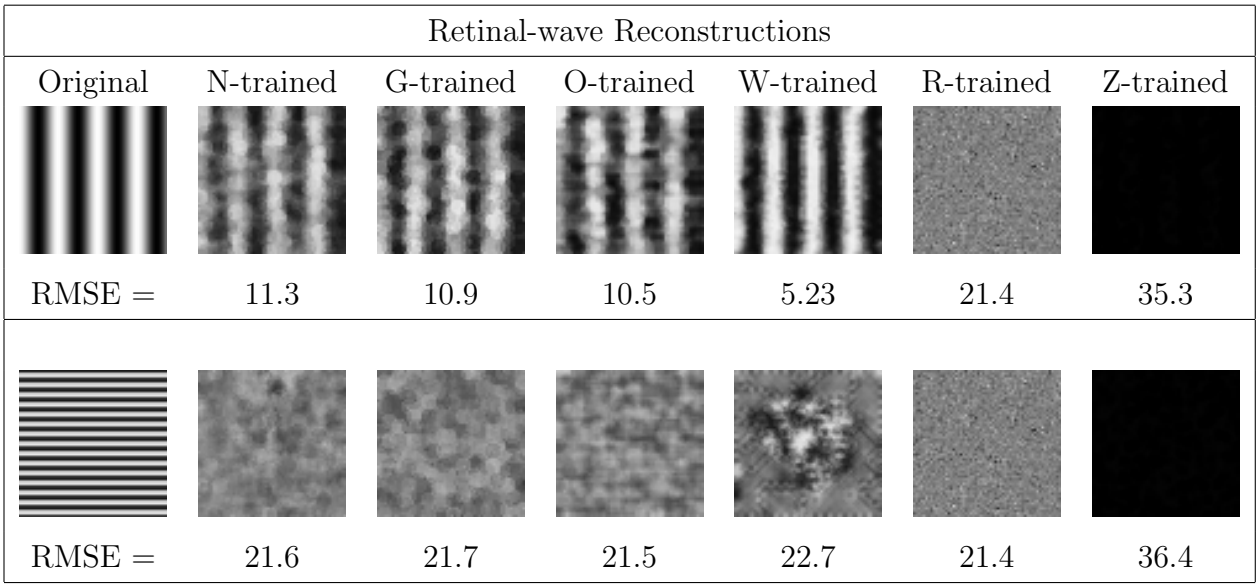


Figure 5.26: Retinal-wave image V1 reconstructions.

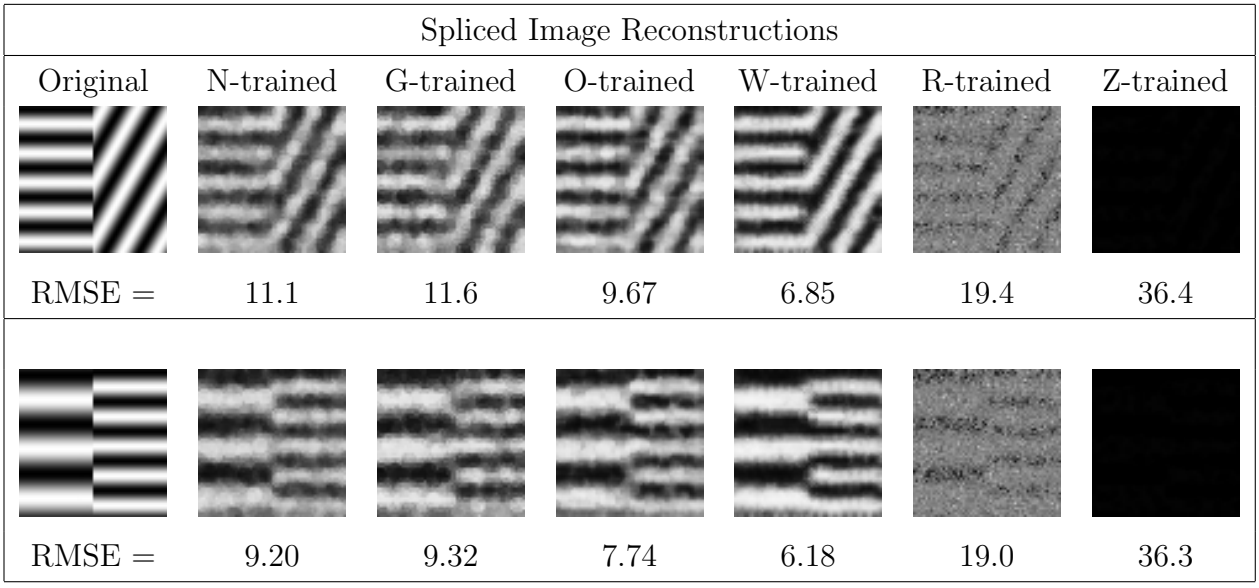
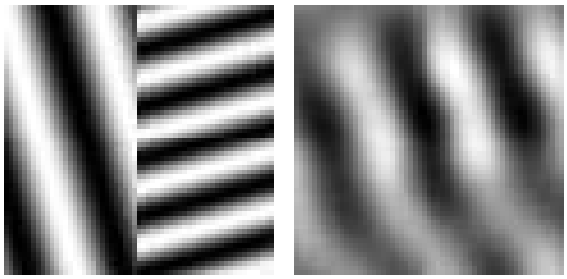
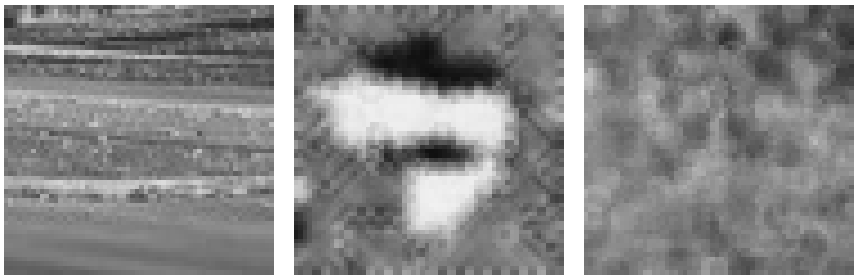


Figure 5.27: Spliced retinal wave reconstructions. Two instances of spliced images being reconstructed using various networks. The first set of images have varying orientation information, while the second set of images vary in frequency. None of the networks are trained on spliced images.



(a) (b)

Figure 5.28: Example of standard CD reconstruction of a spliced image. The network is unable to represent both halves of the image.



(a) (b) (c)

Figure 5.29: Misinterpreted wave reconstruction. Left image shows the original, middle image shows W-trained reconstruction, and right image shows N-trained reconstruction. A relatively homogenous image with a few stripes is reconstructed by the W-trained network to have a much stronger wave-like intensity pattern. In this example we see two heavily darkened stripes between two heavily brightened stripes. While the network is flexible enough to somewhat represent the input, it must do so in a way that conforms to its known priors.

Chapter 6

Conclusions

6.1 Goals

The goal of this work is to develop a biologically inspired variant of the deep belief network learning algorithm, and analyze the results of implementing such a network.

The main contribution of this work is the introduction of the proximal contrastive divergence learning algorithm, along with several variations of deep learning networks that utilize the algorithm. We modify the standard restricted Boltzmann machine and learning algorithm to include locality and topology. We demonstrate experimentally that this single learning adjustment effectively mimics several biological phenomena found in the visual system without taking an extraneous or convoluted approach.

The first part of this investigation corresponds to the discussion of the biological underpinnings of the mammalian visual system. The primary goal of this analysis is to examine aspects of retinotopology in existing models that are used to reproduce receptive fields that are characteristic of our visual system. The second part of our investigation corresponds to a comparative analysis of the results from a simple 2-level RBM model using standard contrastive divergence learning, and one using modified ProxCD learning.

6.2 Discussion

It is infeasible to assume that a hierarchical network for human vision could be genetically predetermined in detail. It is much more likely that the visual system develops through organized yet unsupervised learning mechanisms from a relatively simple structure. By introducing retinotopic learning, we provide this organized simple structure that is capable of learning visual attributes.

The gradual and parallel increase of feature complexity and receptive field size, as found in the visual system, avoids a combinatorial explosion of the number of necessary neurons in the system. Representing all possible scenes without such a system requires an exorbitant number of neurons. Although in the lower levels, many cells are required to cover the range of scales and positions, only a small set of simple features must be represented. Conversely, in higher layers, where neurons are tuned to a greater number of more complex features, neurons show a greater degree of invariance to scale and position.

Studies that attempt to model human vision using a layered deep belief model appear to assume each layer indicates a new layer in the visual cortex. In our case, we represent different areas of the visual pathway. However, there appears to be no reason to make this clear distinction, other than that one area must learn and be stable before the next area can build on it. This is analogous to if a set of neurons are simply transmitting scrambled signals, then any neurons attempting to learn from their output will not gain anything meaningful.

As we alluded to earlier, under normal conditions of human visual development, the LGN and V1 sections would be split up to receive visual information from the different hemispheres of the brain. However since both left and right sides of the visual stream operate in parallel and similarly, we make the assumption that modelling one side is an indicator of how both would operate. We are not simply assuming that introducing a second set of LGN and V1 will have no effect on how the network operates. On the contrary we should expect interesting deviations once the separate hemispherical networks were forced to coalesce. However, with respect to modelling a single side versus two sides, we should still expect the receptive fields and patterns to be similar to what we achieve

here.

Biologically speaking the retina does more complicated processing of the input than we have illustrated. This processing appears to include whitening and decorrelating the input. We did not endeavor to include these pre-processing concepts on the input, however their effects remain to be seen within a proximal network setting.

Up until now the weights have been treated as some sort of generic learning unit. What do the weights really mean though and how do we interpret them? Do we think of them as the amount of axons coming from one cluster of neurons to another? It is easy to think of the weights as a very malleable form of general neural connectivity. This can take on many forms. For photoreceptors to ganglion cells, we can think of them as the graded potentials of the photoreceptors. For ganglion cells to the V1 cortex (that instead have action potentials) we can think of them as the firing rate intensity or the number of synapses, or even the amount (and kind) of neurotransmitter released into the synapses.

Currently the PRBM network would appear to be unsuitable for a discriminatory network based purely on reconstructive ability. However, label nodes have not been attached to deeper representations to truly test this capacity. We have not endeavoured to test the discriminatory behaviour of ProxCD but have instead focused on comparisons to biological and electrophysiological results. We can say that the localization of the connectivity at least appears to be better at filtering out spatially localized noise (also supported by [30]), as well as being able to reconstruct images from various pieces.

6.3 Future Work

There are many possible directions for future research. We believe that as a follow-up to this investigation, there are a few specific avenues that may prove fruitful.

In higher level learning, motion has been cited as an important aiding aspect [4]. Utilizing motion is beyond the scope of the simple RBM and DBN that was outlined in this thesis. Incorporating such a concept requires an aspect of tied responses to previous data, however, the standard restricted Boltzmann machine can only model static frames of data.

How then, does one alter the RBM architecture to model and exploit a temporal structure? There are several approaches that have been developed to incorporate previous time frames of data into learning. Some of these approaches include the conditional RBM [31], and the temporal RBM [29]. It would be interesting to see whether these alternate RBM models would produce the same kinds of results using our proximity modulated learning rule. Utilizing motion would also allow us to further investigate how retinal waves play a role in the early development of our visual system.

Another important aspect of mammalian vision is binocular vision. This concept seems like one of the next logical aspects to model. It involves combining the input from two separate sources (using two neural networks in some manner to mimic the use of two eyes). Biologically, it more accurately represents the mammalian visual system. Also, ocular dominance columns are a known by-product produced in the V1 cortex, which are only possible to study when we introduce a second set of inputs to coalesce with the original.

Modelling coloured vision remains an aspect that we have yet to incorporate into our model. Biologically, the ganglion cells in the retina are already receiving some kind of coloured representation, as we have different photoreceptors in the retina designed to do this job. Like the RBMs that encode motion, models have been developed to incorporate different colour channels (however, lacking retinotopology) [25]. Some questions that remain though are what colour model to use, and how to incorporate multi-channel learning while retaining a retinotopic representation.

The construction of the network structure is open to different approaches that may generate a more informative, deeper picture. Currently we have only tackled an LGN layer and the V1 cortex. One of the simplest extensions to the model is a literal extension of the network into more layers. However, in this case we must also be careful of what simplifications can be made as the connectivity in the brain becomes more and more complex in these deeper layers. Interpretation of the results also becomes more challenging as these deeper layers model more abstract properties of the input.

Lastly, our choice of proximity function could be an avenue of exploration. We chose a Gaussian function based on the observation of retinotopology, and that a Gaussian is generally well behaved statistically in a wide variety of situations. Does this best represent

the biological nature of the process? From preliminary testing, there appeared to be little difference between a plateau (box-car) function and the Gaussian. However, it is unclear exactly what kind of effect other proximity functions may produce. The addition of stochastic proximity widths may elicit a wider variety of V1 cell behaviours.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169, 1985. 13, 15

[2] J. Atkinson, O. Braddick, and K. Moar. Development of contrast sensitivity over the first 3 months of life in the human infant. *Vision Res*, 17:1037–1044, 1977. 44

[3] B. Balas and P. Sinha. Receptive field structures for recognition. *Neural Computation*, 2005. 60

[4] B. Balas and P. Sinha. Observing object motion induces increased generalization and sensitivity. *Perception*, 37:1160–1174, 2008. 81

[5] C. Blakemore and G. F. Cooper. Development of the brain depends on the visual environment. *Nature*, 228:477–478, 1970. 41, 70

[6] T. Bonhoeffer and A. Grinvald. Iso-orientation domains in cat visual cortex are arranged in pinwheel-like patterns. *Nature*, 353:429–431, October 1991. 12, 62

[7] M. C. Crair, D. C. Gillespie, and M. R. Stryker. The role of visual experience in the development of columns in cat visual cortex. *Science*, 279:566–570, January 1998. 67

[8] G. Desjardins and Y. Bengio. Empirical evaluation of convolutional RMBs for vision. Technical Report 1327, Université de Montréal, 2008. 29

[9] Dowling J. E. *Neurons to Networks: An Introduction to Neuroscience*. Belknap Press, 1992. 7

[10] I. Godecke and T. Bonhoeffer. Development of identical orientation maps for two eyes without common visual experience. *Nature*, 379(6563):251–254, January 1996. 38

[11] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006. 24

[12] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *Journal of Physiology*, 148:574–591, 1959. 11

[13] A. Hyvarinen and P. O. Hoyer. A two-layer sparse coding model learns simple and complex cell receptive fields and topography from natural images. *Vision Research*, 41:2413–2423, 2001. 30, 70

[14] A. Hyvarinen, P. O. Hoyer, and M. Inki. Topographic independent component analysis. *Neural Computation*, 13:1527–1558, July 2001. 28

[15] J. P. Jones and L. A. Palmer. An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, 58:12331257, 1987. 60

[16] E. R. Kandel, T. M. Jessell, and J. R. Sanes. Central visual pathways. In Eric R. Kandel, James H. Schwartz, and Thomas M. Jessell, editors, *Principles of Neural Science: 4th edition*, chapter 27, pages 533–540. McGraw-Hill, 2000. 12

[17] E. R. Kandel and J. H. Schwartz. *Principles of Neural Science*. Elsevier/North-Holland, New York, 1981. 5, 9

[18] H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area V2. In *Advances in Neural Information Processing Systems (NIPS)*, pages 873–880, 2007. 28, 29, 60

[19] B. Li, M. R. Peterson, and R. D. Freeman. Oblique effect: A neural basis in the visual cortex. *Journal of Neurophysiology*, 90:204–217, 2003. 68

[20] M. Meister, R. Wong, D. A. Baylor, and C. J. Shatz. Synchronous bursts of action potentials in ganglion cells of the developing mammalian retina. *Science*, 252(5008):939–943, May 1991. 40

[21] J. Ng, A. A. Bharath, and L. Zhaoping. A survey of architecture and function of the primary visual cortex (V1). *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007. 11

[22] M. Norouzi. Convolutional restricted boltzman machines for feature learning. Master's thesis, Simon Fraser University, 2009. 28, 29, 60, 70

[23] V. H. Perry, R. Oehler, and A. Cowey. Retinal ganglion cells that project to the dorsal lateral geniculate nucleus in the macaque monkey. *Neuroscience*, 12:1101–1123, 1984. 8

[24] D. A. Pollen and S. F. Ronner. Phase relationships between adjacent simple cells in the visual cortex. *Science*, 212(4501):1409–1411, 1981. 60

[25] M. Ranzato and G. E. Hinton. Modeling pixel means and covariances using factored third-order boltzmann machines. *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 28, 82

[26] A. M. Sillito, K. L. Grieve, H. E. Jones, J. Cudeiro, and J. Davis. Visual cortical mechanisms detecting focal orientation discontinuities. *Nature*, 378:492–496, November 1995. 11

[27] D. Smyth, B. Willmore, G. E. Baker, I. D. Thompson, and D. J. Tolhurst. The receptive-field organization of simple cells in primary visual cortex of ferrets under natural scene stimulation. *Journal of Nauroscience*, 23(11):4746–4759, June 2003. 11

[28] I. Suner and P. Rakic. Numerical relationship between neurons in the lateral geniculate nucleus and primary visual cortex in macaque monkeys. *Visual Neuroscience*, 13(03):585–590, 1996. 37

[29] I. Sutskever and G. Hinton. Learning multilevel distributed representations for high-dimensional sequences. Technical report, 2007. 82

[30] Y. Tang and C. Eliasmith. Deep networks for robust visual recognition. *International Conference on Machine Learning*, 2010. 81

[31] G. W. Taylor, G. E. Hinton, and S. T. Roweis. Modeling human motion using binary latent variables. *Advances in Neural Information Processing Systems*, 2006. 82

[32] A. Torralba and A. Oliva. Statistics of natural image categories. *Computation in Neural Systems*, 14:391–412, May 2003. 39