Title
Analysis and applications of the Locally Competitive Algorithm

Author
Paiton, Dylan M

Publication Date
2019

Peer reviewed|Thesis/dissertation

Analysis and applications of the Locally Competitive Algorithm

by

Dylan M Paiton

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Vision Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Bruno Olshausen, Chair
Professor Jack Gallant
Professor Stanley Klein
Professor Trevor Darrell

Spring 2019

Analysis and applications of the Locally Competitive Algorithm

Abstract

Analysis and applications of the Locally Competitive Algorithm

by

Dylan M Paiton

Doctor of Philosophy in Vision Science

University of California, Berkeley

Professor Bruno Olshausen, Chair

The Locally Competitive Algorithm (LCA) is a recurrent neural network for performing sparse coding and dictionary learning of natural signals. The network itself lacks much of the anatomical structure we see in real brains, but it does include lateral connectivity and iterative, or recurrent, computation. These together produce population nonlinearities that facilitate desirable coding properties, including improved robustness, selectivity, and efficiency when compared to more traditional architectures. When trained on images of natural scenes, the network also has many overlapping response properties with those of biological neurons. This has encouraged scientists to use it as a means to conceptualize theories of neural coding and explore their consequences. Probing the network enables us to observe and test theories that account for neurophysiological phenomena. In this thesis, we will provide motivation for investigating the model, a detailed derivation of the model, an analysis of its response properties, and offer some extensions. The core computational principles in the LCA are distinct from those used in most production artificial neural networks, and we will argue that there is much to be gained from incorporating LCA-like computations in exchange for feed-forward, pointwise nonlinear model neurons.

Dedication.

I would like to dedicate this thesis to Janet Paiton, Donnie Paiton, Loni Hammer, Justin Hammer, and Christine Hopkins. Your love and support have kept my spirits high and my life on track, and I can't imagine having done any of this without you. I hope to continue to make you proud and reward the valuable life lessons you gave me. Thank you all so much!

Contents

List of Figures

List of Tables

Acknowledgments

This thesis benefited greatly from feedback and contribution from my colleagues and friends. Christine Hopkins, Wesley Chavez, Will Shainin, Sheng Lundquist, Charles Frye, Matt Brown, and Chris Warner all read portions of this text and provided valuable feedback. Thanks Vasha Dutell for helping me hunt down sources. Thanks Parker Conroy for pushing me to make continual progress. The hard-thresholded LCA analysis work was done with Will Shainin and Gar Kenyon. The weakly-supervised learning work was done under the advisement of Jack Culpepper at Yahoo, Inc. Simon Osindero, Rob Hess, Clayton Mellina, and Tobi Baumgartner also provided valuable feedback on that project. The subspace LCA work was done in collaboration with Ryan Chan. The Iso-Response contour work was done in collaboration with Charles Frye, Joel Bowen, Jasmine Collins, Sheng Lundquist, and Melanie Mitchell. The orientation selectivity experiments and rate-distortion experiments were done in collaboration with Sophia Sanborn and Spencer Kent. Also thank you Ryan Zarcone for being an invaluable soundboard and problem solver. Thank you to Vision Science faculty Austin Roorda, Stan Klein, Michael Silver, and Martin Banks for your support and friendship. Thank you to my thesis committee Bruno Olshausen, Stan Klein, Jack Gallant, and Trevor Darrell for your feedback.

I would like to thank the following for repeatedly taking their valuable time to work through challenges (both scientific and personal) with me: Charles Frye, Ryan Zarcone, Jesse Livezey, and Shariq Mobin. I am extremely grateful to have had an opportunity to work with such a collaborative group at the Redwood Center. Thank you to everyone in this list for listening, helping me sculpt my ideas, and providing feedback: Alex Anderson, Alex Terekhov, Brian Cheung, Bruno Olshausen, Charles Frye, Charles Garfinkle, Chris Hillar, Chris Warner, Connor Bybee, Eric Dodds, Eric Weiss, Friedrich Sommer, Gautam Agarwal, Guy Isley, James Arnemann, Jascha Sohl-Dickstein, Jasmine Collins, Jesse Engel, Jesse Livezey, Karl Zipser, Kris Bouchard, Louis Kang, Mayur Mudigonda, Michael Fang, Mike DeWeese, Neha Wadia, Paxon Frady, Pentti Kanerva, Pratik Sachdeva, Ryan Chan, Ryan Zarcone, Saeed Saremi, Shariq Mobin, Sophia Sanborn, Spencer Kent, Stephen Shepard, Tyler Lee, Urs Koster, Vasha Dutell, and Yubei Chen.

My foundation in theoretical neuroscience was established while working with the Garmy under the advisement of Gar Kenyon. Thank you so much Gar for continuing to provide invaluable support, advisement, and friendship. Thank you also to Pete Schultz, Steven Brumby, Amy Larson, John George, Michael Ham, Gerd Kunde, Brennan Nowers, and John Galbraith for providing project feedback and collaboration.

My sanity and happiness are *strongly* dependent on support form my Very Best Friends and my Vision Science cohort. You all know who you are. You keep me going - thank you.

Finally, all of the work in this thesis was overseen by my amazing PhD advisor, Bruno Olshausen. He was a continual source of direction, encouragement, and inspiration for my work. It was his writing that pulled me into this field in the first place, and his passion for understanding neural computation has continued to stoke my own. Thank you, Bruno.

Chapter 1

Introduction

The visual sense is powerful in that it typically dominates other sensory inputs for assessing the physical properties of objects (Ernst et al. 2002). However, our conscious perception of the world does not match the visual signal coming into our eyes. The dichotomy between the noisy light signal reaching our brain and the immediate and compelling nature of our visual percept can bias our investigations (Rodieck 1998). Vision is intuitive for humans, which makes it difficult to grasp the complicated nature of deriving our perception from light reflections in the world. Indeed, early machine vision work approached the problem from what is in hindsight a simplistic perspective (Papert 1966). For example, the seemingly elementary process of differentiating edges caused from physical boundaries of objects against those caused by lighting (i.e. shadows) was once considered introductory by the computational vision community, but it is still unsolved today (Adelson 2000; Barron et al. 2012). Much of the history of studying vision has been dominated by the notion that it is simply a passive sense facilitated by filters and feature detectors, although experimental evidence overwhelming supports an alternative viewpoint that vision is an active, dynamic process (Olshausen 2013a). Vision research is fraught with inherent complications, unsolved problems, and unintuitive answers, but it is also the source of compelling examples of fascinating phenomena in human perception and consciousness. It is motivated by our ability to perceive over 10 orders of magnitude of illuminance levels (Norton et al. 2002), untangle complicated inverse graphics problems such as shape from shading and edge detection, and rapidly recognize objects. This all happens while also demonstrating extreme robustness to input noise and distributional shift. It is also motivated by instances where we fail to perceive the stimulus as it is physically, in the form of illusions, bi-stable percepts, and metamers. Importantly, all of these phenomena are accessible and compelling to anyone. They drive the field at large and the work in this thesis.

It is well understood that there exists a dichotomy between how we perceive our world and the actual information impinging our sensory systems. For example, our eyes are constantly in motion, even when we fixate our gaze upon a stationary object, and yet the fixated object appears stable. Another example is that the retina unevenly samples visual information, yet we perceive a uniformly resolved visual world. Questions then arise about what sort of signal

processing the brain is performing to allow us to perceive the world as we do or, more narrowly, how processing in the brain results in ecologically relevant features and objects becoming salient. Early works in philosophy identified this and drew ties between objective laws of nature, our perceptual experience, and consciousness (Kant 1790; Helmholtz 1878) (and see (Westheimer 2008) for an excellent retrospective). One can make a clear connection from the ideas espoused in these and other early works to a general movement in the theoretical neuroscience community to think of perception as an "inference" problem. By this we mean that the brain uses prior knowledge of the world, gained through evolution or experience or most likely both, to resolve ambiguity in sensory signals, resulting in perception. This line of reasoning has been propagated by contemporary vision scientists and mathematicians, including Joseph Atick (1990), Horace Barlow (2009), Fred Attneave (1954), David Field (1994), David Mumford (1994), Bruno Olshausen (2013b) and others. Early philosophy suggested that perception is a *guided hallucination*, while today it is depicted as an *active process*. This is to say that our brain composes what we perceive, using information from the outside world as an anchor.

1.1 The brain is an information processor

Neuroscience is a discipline that spans many levels of understanding in order to investigate how the nervous system works. The field is broad, so before continuing it is important to narrow the scope. In David Marr's 1982 book, *Vision* (1982), he defines three levels of understanding that since have been used heavily to motivate theoretical neuroscience research. These levels are Computational Theory, Representation and Algorithm, and Hardware Implementation. Importantly, Marr asserts that all three levels must be understood before one can emulate the entity under study. In our case, we wish to understand how brains work and how to build a brain. The answers to these questions are vast and largely unsolved, but here we provide a contribution. We will investigate hypotheses for computations underlying visual perception by analyzing models of neural coding and signal representation.

Marr also supported the perspective of thinking about the brain as an information processor. This is an idea that many theoretical neuroscience academics take for granted, but was not considered seriously until relatively recently. The seminal early works of Hubel and Weisel (1959) cemented the idea for many when they showed that individual visual neurons responded selectively to relevant features in the world. From this and other works, and likely resulting from our limited ability to record from populations of neurons, many in the community have adopted a "single neuron doctrine", whereby experiments and explanations of phenomena are based in the information processing capabilities of the individual neuron. Indeed, many attempts at understanding non-linear response properties of visual neurons are focused on individual responses (Priebe et al. 2012), but this reductionist approach moved the field to a position of idolizing the computations of an individual neuron while disregarding the synergies created when computation is performed at the population level. Several modern works have advised that the community should move away from this perspective (Barlow

1972; Olshausen and Lewicki 1999; Zetzsche et al. 1999; Series et al. 2003) and towards the idea of population coding in the brain, which requires the consideration of the information processing capabilities of a family of neurons in a given region. A complete understanding of neural computation will likely require both population level and individual neuron level analysis. This principle underlies the core models, experiments, and theories studied in this thesis.

At each processing stage, the brain must form a representation that is both *efficient* and *useful*. Efficient is fairly easy to explain - there are constraints such as physical space and metabolic resources that must be adhered to. However, the idea of what might be useful is more vague. For example, consider information representation in the primary visual cortex (V1) - the utility of a representation is going to depend on the objective of downstream neurons, which is not fully known. V1 projects to a variety of brain areas, including those responsible for self motion, sensory integration, memory, and the rest of the visual processing hierarchy. Therefore, either the representation in V1 needs to be general enough to be decoded for many different tasks, or there must be numerous independent representations in V1. Indeed, to prescribe the entire visual processing stream to a single objective now seems foolish (Barlow 2009), although this has been the predominant paradigm in neural computation. The primary model we study is focused on neural coding in V1. It is not described at a biophysical level of analysis and is not meant to represent a universal computational principle for visual processing. Instead, it is a tractable implementation of a general idea that information must be represented in V1 at a population level, with an emphasis on making explicit important details in the signal for later brain regions to interpret while preserving as much information as possible. Success of the model should provide support for this idea and motivate further work into alternate levels of analysis such as a hardware implementation (i.e. how might the brain actually do this). Before going into the V1 model, we will spend the rest of this chapter outlining the inputs to V1 and important theoretical motivation for the model itself.

1.2 Inputs to V1

To investigate V1, it is important to first understand how the signal that reaches V1 is encoded and what the many processes are that transform the incoming information. Below we have focused on three primary systems that modify the visual signal before it reaches V1: ocular motion, retinal sampling and processing, and the thalamus. Our treatment of these systems is terse at best, and there are more systems involved that each constitute an important research direction. Nonetheless, it is important to highlight a few key details of these systems when considering how V1 processes information.

Light

The process of light irradiating objects in the world before entering our eyes is *highly* nonlinear. When light is incident on an object, a number of well described physical alterations occur.

They vary dramatically based on the non-uniform physical composition of the object and include light scattering, absorptance, and reflectance. On top of this, objects typically move along a continuous nonlinear trajectory in the world and often occlude each other, resulting in a dramatic change in the incoming signal across occlusion boundaries. It is the goal of the vision system to deconstruct these effects in order to identify what is in the world. Luckily, most of these properties are consistent among objects and change smoothly through time. For example, the absorptance of a particular rock can be considered effectively constant and objects in the world are do not teleport from one position to another. Thus, our brain can exploit the stability of the input to decipher one object from another without explicitly modeling physical properties like absorptance. This is accomplished via a series of anatomically localized stages that all share recurrence as a common processing motif. The first stage in the visual processing hierarchy is the retina.

The retina

In order to see objects, our eyes focus reflected light onto our retinas. Each retina has over 100 million photoreceptors, which convert incident photons into chemical signals to be processed by the rest of the retinal network. Photoreceptors vary in the type of information they convey. About 120 million of them are rods, which primarily function in dim lighting conditions and do not convey color information. The other 6 to 7 million of them are cones, which supply most of our visual experience. In most of the retina, multiple photoreceptors will converge onto a single output neuron, a ganglion cell. This convergence is not direct, however. The signal first passes through a complicated network of horizontal, bipolar, and amacrine cells that all do their part to transform and condition the signal. The ratio of photoreceptors to output channels is nearly 1 to 1 in the fovea, which, in humans, is a circular patch of retina with the highest acuity. The ratio decreases to over 100 inputs to 1 output at the far periphery. As a result of this varying convergence, visual acuity is highest in the fovea, and decreases radially in the periphery. In all, information from the roughly 130 million photoreceptors is conveyed to the brain by approximately 1.5 million ganglion cells. Mammalian ganglion cells themselves can be divided into at least 15 different types, most of which evenly tile the input space of the retina. These different output cell types convey different information to the brain about the same area in visual space. Finally, information is sent from the retina down the optic tract to the brain. The optic tract is a narrow communication channel with important constraints that are relevant to information coding: one is that the diameter of the nerve bundle creates a "blindspot" where your retina cannot process light information, and another is that the eye needs to be able to move quickly, so the bundle must be flexible enough to allow this movement. Both of these constraints encourage the retina to prioritize compressing information to be passed down a narrow channel to the brain.

Although it is largely agreed upon that preprocessing is done by the retina before the signal is sent to the brain, the amount of processing and the method by which it is communicated is still highly debated. Work from Joseph Atick and colleagues proposes that the center-surround receptive field of the retinal ganglion cell performs a "whitening" of the input signal, which

is a statistical transform that produces a signal with equal power at all spatial frequencies (Atick and Redlich 1990; Atick and Redlich 1992). Evidence for the retina performing spatial whitening of input signals has only been indirectly found through psychophysics experiments (Atick and Redlich 1992), although evidence for temporal whitening has been directly shown (Dong et al. 1995) by recording from thalamic inputs. Whitening would be a desirable preprocessing step for the brain because it performs decorrelation on the structured input signal, which allows for a more efficient use of the optic tract.

Another important property of the retina is the division of labor among different cell types (Van Essen et al. 1995). Although there are many different ganglion cells, two primary types are the midget and parasol cells. They divide the visual input to encode low temporal, high spatial frequency (midget) and high temporal, low spatial frequency (parasol) signals. This division has been proposed as an optimal division of the input given the restrictions of the retina and the desire to compress the signal (McIntosh et al. 2016; Van Essen et al. 1995). It is also important to consider how this division of labor impacts processing downstream, where the magno (with parasol cell inputs) and parvo (with midget cell inputs) processing streams remain largely divergent through V1.

Eye movements

In photography it is critically important to stabilize the camera, especially in dim lighting conditions. When the camera shutter is open, the light sensors continuously integrate light information such that any movement is going to result in blurring of spatial details. Many modern imaging systems have tools to compensate for this, but anyone who has tried to take a concert or city lights photo can tell you that it is far from a solved problem. Any organism with eyes also has to solve this problem, but we know little about how it is done (Olshausen and C. H. Anderson 2010; Burak et al. 2010).

Given the eccentrically nonuniform sampling being performed by the retina, a logical conclusion for the purpose of eye movements is that they are simply to "foveate", or direct the fovea to, whatever object in the scene we wish to see with the most detail. This class of movements is called a saccade, and can range in scale from 1 degree visual angle to the full extent of your eyes ability to move (>100 degrees). To put this into perspective, you can fit about 120 foveal cones ($\sim$2 microns in diameter each) into 1 degree visual angle on the retina (about 0.3mm). Another large scale eye movement is smooth pursuit, which has a unique (near constant velocity) time signature and involves actively and continuously adjusting fixation on an object that is moving in the world. Saccade and smooth pursuit eye movements are typically purposeful (or conscious) and are classified as non-fixational. Eye movements that subtend less than 1 degree visual angle are classified as fixational eye movements, are typically not conscious, and are subdivided into microsaccades, drift, or tremor. During free viewing, microsaccades happen 1 to 2 times per second at a peak velocity of as much as 100 degrees per second. They have been shown to have much overlap with the physiological and psychophysical characteristics of larger-scale saccades. Drift, on the other hand, can be thought of as a low frequency (<40Hz), slower (25-60 arcmin/sec), and constant-velocity

fixational movement. As a first approximation, drift can be roughly characterized as following a Brownian path around a fixation point, in that the variance of the spatial distribution increases approximately linearly with time (Rucci and Victor 2015). However, this description of drift is merely to orient one to the nature of motion, for there is not sufficient evidence that the motion itself is generated by a random process. Finally, tremor is typically characterized as a high-frequency (∼70Hz), small amplitude random movement and is most often explained as noise caused by the muscles innervated on the eye.

Fixational eye movements were once all classified as noise, but now have been implicated in improving visual acuity (Ratnam et al. 2017; Rucci, Iovin, et al. 2007) and performing signal processing functions, such as normalization and whitening (Aytekin et al. 2014). Each class of eye movements, whether voluntary or involuntary, could potentially be constructively contributing to the signal that is interpreted by the retina. Considering that visual information translates across roughly 60 cones at a peak rate of about 6,000 cone diameters per second during a microsaccade, it is surprising that our perception is stable at all, much less improved. Recent efforts have established theories and demonstrations by experiment of how we might achieve stable perception (Arathorn et al. 2013; Bridgeman 2010; Murakami et al. 1998; Burak et al. 2010) and how these fixational eye movements might be performing enhancing operations on the incoming visual information (Ahissar et al. 2012; Mostofi et al. 2016; Kenyon et al. 2004). However, many still maintain that these eye movements are in fact a "bug" that must be compensated for (Packer et al. 1992; Kowler et al. 1979; Engbert et al. 2011).

In our work we assume the retinal input is fixed, although there is active research underway that includes eye motion (Ratnam et al. 2017; A. Anderson et al. 2019). Incorporating eye motion into models of V1 computation will involve extending them to explicitly code for time-varying stimulus, an example of which can be found in (Olshausen 2003a). Understanding how eye motion influences the signal coming to the eye will undoubtedly be extremely important for understanding neural computation in V1.

The Thalamus

After the incoming visual signal is transformed via eye movements and retinal processing, it is communicated along the optic tract to the lateral geniculate nucleus (LGN) of the thalamus and then to the primary visual cortex, V1. It has become increasingly clear that the thalamus performs considerable computations on the visual data. Like their retinal inputs, LGN neurons are selective to onset and offset of light, and have structured spatiotemporal receptive fields. Unlike the retina, however, the LGN has binocular and color-opponent cells (Schiller et al. 1978; Schmielau et al. 1977). It is known that the division of labor identified as different processing streams (most notably the magno- and parvo-cellular streams) is maintained in the LGN and there are notable differences between cells in these streams. For example, conductance velocity (which likely impacts information transfer rate and temporal sensitivity) for magnocellular inputs is faster than parvocellular and although both streams exhibit center-surround organization, the magnocellular neurons appear to be invariant to

wavelength while the parvocellular neurons have specific color opponency (Schiller et al. 1978). Although many regard the LGN as a relay station for information traveling from retina to cortex, the retina accounts for only a small fraction of the inputs to the LGN (Weyand 2016). Other inputs include branched afferents from axons that ultimately connect to motor centers as well as inputs from the brainstem, suggesting that information coming from the thalamus includes motor instructions (Guillery et al. 2002). Additional feedback connectivity from the cortex likely provides context information that allows for modulation of the feedforward signal (Weyand 2016; Ghodrati et al. 2017). The pulvinar nucleus of the thalamus has been proposed as a driver for higher order cortical areas as well, where information from cortical areas is routed through the thalamus before driving other cortical areas. Indeed, anatomical and physiological evidence suggests that layer 5 (output) cortical neurons drive thalamus, instead of just modulating it, and that thalamic outputs provide powerful inputs to higher-order visual cortical regions (Guillery et al. 2002). The thalamus can perform significant modification to the signal coming into the cortex: it can be used to adjust binocular salience (Schmielau et al. 1977), dynamically route or gate information (Olshausen, C. H. Anderson, and Van Essen 1993; Weyand 2016), and compensate for motor commands (Guillery et al. 2011).

Accounting for these stages in our visual models

The focus of this thesis is in understanding a particular model for information processing in V1, namely sparse coding. Our analysis of the model provides important explanations of shared response properties between the model and biological systems. We also generate hypotheses about coding properties of the model. However, the burden of motivating this model as a candidate for brain computation is largely left to previous and future work. As such, much less emphasis is put into our models having high correspondence to the biological input (pre-V1) pathway of animals in the real world. In the following chapters, we will reduce all of the processing in early vision areas to simple linear transforms. Instead, our focus is on understanding the model dynamics themselves. We will use static images of natural scenes, as well as small contrived datasets as stimulus for our model. Extending the algorithms and ideas from this thesis to account for the complexity of preprocessing in the retina and thalamus, as well as inputs from other important visual and non-visual cortical regions would be an important contribution to the field. We do believe that including these important facets of biological processing will not detract from the core computational motivations or principles in the sparse coding model.

For natural scene stimulus, we will perform a preprocessing step of whitening data (1997) to simulate the proposed whitening being performed by the eye (Atick and Redlich 1992; Rucci and Victor 2015). Beyond that, we assume that our model neurons are receiving input from a uniform sampling of identical phototransduction cells. Little work has been done to try to model the entire early vision pipeline, although (Shan and Cottrell 2013; Doi et al. 2007; Lindsey et al. 2019) all provide promising directions. With this in mind, it is important to also note that the core goal of the sparse coding model is not to prescribe a

specific computational algorithm to V1 cells, but instead to determine what is possible by applying an efficiency-constrained generative model to natural data. This broad concept is proposed to be an underlying computational principle in V1, which has been well supported in the literature.

1.3 Theoretical foundation for understanding neural computation in V1

We are interested in understanding how the early vision system transforms light signals into a neural code to provide information about objects in the world. Contrary to what occurs in modern neural network models, we believe that the goal of our vision system is not to search a scene and label every object in it. The goal of the vision system is more ecological. It aims to build a representation of our world that integrates our many sensor modalities. It allows us to identify the causes of the incoming light signal, which includes labels, locations, poses, etc. of important objects in the scene. Our internal model must account for ill-posed problems to extract 3D scene information, integrate and store the information across time and modality, and guide motor actions. This is the groundwork upon which we seek to understand the computational principles of V1.

Our visual world is a highly structured environment, with correlations and redundancies present at all spatial scales. Given restrictions in space, metabolism, and neural wiring length, it is in the best interest of an ideal visual observer to model these redundancies using an efficient code. The initial objective of the vision system is to convey as much light information as possible from the eye to the brain using the limited bandwidth provided by the optic tract. Then, after this bottleneck, the brain has more freedom to adjust, modify, expand, or collapse the data signal, although the aforementioned biophysical constraints must still be considered. It appeals to our intuition to think that the brain should represent the world in a meaningful way, such that the causes of the scenes we see are encapsulated. Here we will discuss theories on how the brain efficiently models, or accounts for, redundancies in our world using a set of distributed, statistically independent representations.

Fred Attneave (1954) was the first to apply newly minted ideas in information theory to understanding visual coding. He suggested that efficiency was a primary goal of the visual system. The efficient coding hypothesis was later posed in terms of redundancy reduction (Barlow 1961), with reasoning that the brain may seek an efficient code of the input in order to minimize the number of neurons required to represent the signal. This theory was supported by (Laughlin 1981), who compared the responses of fly eyes with contrast levels in natural scenes and suggested that the coding mechanism use is optimal in terms of efficiency and the neuron's information capacity. Joseph Atick (1992) suggested that information theory could provide a basis for understanding sensory processing, which was also supported from other findings around the same time (Hateren 1992; Field 1994).

Anatomical evidence tells us that V1 expands the image representation coming from

LGN by having many more outputs than inputs (Olshausen 2003b). Thus, redundancies are probably *created* in the perceptual process. The goal of cortical processing, then, cannot be said to be strictly redundancy reduction or compression. As an alternative to redundancy reduction, several researchers have argued that the goal of perception should not be discussed in isolation from action; an organism forms perceptual representations for the purpose of directing its behavior towards the achievement of desirable outcomes and away from undesirable ones (Barlow 2009; Simoncelli and Olshausen 2001). From this perspective, the brain aims to extract the statistical structure of the input in order to form a"meaningful" representation that recovers the environmental causes of the sensory data, which it can use to guide action. Along these lines, the efficient coding hypothesis has been revised to emphasize redundancy *representation* rather than reduction (Barlow 2009). Redundancies in the input signal indicate structure in the environment. An encoding that makes these redundancies explicit encodes the causal and statistical structure of the environment, which the organism can exploit to plan and direct behavior. The first-order method for doing this is to disregard all details that are not pertinent to ecologically relevant objects in the world, which happens to be precisely what is done in current object recognition deep networks (Tishby et al. 2015). However, a better solution is to learn the probabilistic structure of the world, such that one can *generate* the fine details when needed. In the words of Barlow: "Instead of thinking of neural representations as transformations of stimulus energies, we should regard them as approximate estimates of the probable truths of hypotheses about the current environment, for these are the quantities required by a probabilistic brain working on Bayesian principles" (Barlow 2009). This notion of a probabilistic brain is shared by many (Kersten et al. 2004; Lee et al. 2003; Lewicki and Sejnowski 1997; Olshausen 2013b), and is a core principle underlying the research in this thesis.

To apply the ideas outlined above to vision, we need to understand the statistics of natural images. David Field used Fourier analysis to demonstrate that the power spectra of natural images typically falls off as $\frac{1}{f^2}$, where f is spatial frequency and the computed power is circularly averaged across orientations (Field 1987). Wherever there is signal, as measured by Fourier power, then there is structure in the scene. Therefore, a $\frac{1}{f^2}$ power spectrum tells us that we should expect to find structure at all (reasonable) spatial frequencies in natural scenes, which is hopefully intuitive if you consider the high spatial frequencies present in tree bark or sand and the low spatial frequencies found from faint shadows on a wall or clouds in the sky. However, this characteristic "pink" power spectrum is not sufficient for generating natural scenes. Natural scenes also exhibit a large amount of higher-order structure, such as elongated edges. A dictionary of wavelets can be used to efficiently encode such structure (Field 1999). They produce more independent codes than other dictionaries, which results in a sparse code with high kurtosis[1]. Perhaps unsurprisingly, the linear approximation of receptive fields for individual V1 neurons are well-fit with a localized wavelet dictionary

[1]Kurtosis is the fourth-order moment of a data population, which indicates "peakiness" by measuring how much mass lies in the tails of the distribution. We would want our code to be peaked at zero, indicating that most of the neurons are silent.

composed of Gabor functions. The notion of efficiency as measured by sparsity will is a core component of the sparse coding model, which is our primary subject of interest.

It has been demonstrated that a typical redundancy reducing code leads to large errors in estimates of the frequency of a particular input, since many neurons are active in response to both the input of interest as well as other stimuli (Gardner-Medwin et al. 2001). A sparse code, on the other hand, learns dictionary elements that occur independently in the environment and produces a factorial code. Any deviations from a factorial distribution signal indicate a previously unknown statistical dependency to be learned. Later, we will discuss how hierarchical extensions of sparse coding can model these deviations and learn higher order statistical regularities. An ultimate goal of these extensions is to develop a model that extracts all of the statistical structure to form a *meaningful* representation that recovers the environmental causes of the sensory data to guide action.

1.4 Thesis outline

This thesis investigates how lateral connectivity and recurrent inference affects image coding properties of model neurons. The primary network of interest is rooted in probabilistic inference and builds an internal model of its sensory world through experience. Although the algorithm is fairly simple, its lineage goes back to a centuries old idea that our *perception* of the world is achieved via an active, conscious process rooted in exploiting the unique statistical structure of natural signals. This first chapter was devoted to outlining important biological systems that lead up to the brain region of interest - V1. Additionally, we introduced relevant theories from others that motivate the later chapters. Next, We will describe the sparse coding model and Locally Competitive Algorithm (LCA) in detail, with a complete derivation and exposition of the features learned and coding properties. We then extend the model for a semi-supervised learning application as well as a hierarchical variant that learns cells that are analogous to "complex" cells in V1. Finally, we improve upon recent methods for understanding model neurons, which allows us to increase a network's degree of nonlinearity without significantly decreasing our understanding of the computations performed. We use this method to argue for increasing the complexity of model neurons in neuroscience and machine learning research and to provide a perspective on the robustness, efficiency, and selectivity/invariance properties of LCA.

Chapter 2

Models for Biological Encoding of Natural Images

2.1 Introduction

Sparse coding is a generative model for explaining natural signals. In other words, it models the joint probability distribution between natural signals and a latent code such that it is able to generate new naturalistic signals from the code. Although the model and motivating theory have been explored in other modalities, here we will focus on coding visual stimulus as a proxy for computations performed by layer 4 simple cells in V1. We will begin this chapter with a probabilistic derivation for the sparse coding model. Importantly, this will make clear the various assumptions built into the model, which will be relevant for our interpretations of results in later chapters. The first section will get us to an energy function that a sparse coding algorithm would seek to minimize. In the second section, we recapitulate the sparse coding model from an optimization perspective. Next, we derive the Locally Competitive Algorithm (LCA), which is a neural network model designed to perform sparse inference. This network is the subject of the rest of the thesis. Finally, we compare the LCA to other popular neurally-inspired image coding models, namely Predictive Coding (Rao et al. 1999) and ICA (Bell et al. 1997). Depending on the context, we may refer to the connections between pixels and neurons as a matrix of weight vectors, a dictionary of elements, or a set of basis functions - these should be considered synonymous in this work.

2.2 A probabilistic approach for image coding

As was explained in chapter 1, the signal coming from the two-dimensional retina is incomplete and the problem of inferring the exact three-dimensional structure of the world from that signal is impossible. However, some solutions are more likely than others and ecological pressure encourages brains to perceive highly likely solutions. If population activity in the brain represents possible hypotheses about the world, then we want to model the brain in

a probabilistic, Bayesian framework. In this section, we derive such a framework and show that sparse coding approximately solves it.

We aim to encode an incoming signal efficiently under the assumption that the signal is composed of structured components and unstructured additive noise. We do this with sparse coding, which assumes a linear generative model:

$$s = \Phi a + \varepsilon, \tag{2.1}$$

where s is our input signal, Φ is an overcomplete (in that the number of columns in Φ is greater than the number of rows) dictionary of functions, a is a row vector of activation coefficients, and ε is Gaussian noise (which we justify below). For a given input signal, we must solve the ill-posed task of inferring an optimal set of coefficients. The problem is difficult in part because the noise is unknown and therefore cannot be completely accounted for. Another reason is that the dictionary is overcomplete and thus non-orthogonal, resulting in an infinite number of possible combinations that are each equally valid representations. Following the works of Olshausen and colleagues (Olshausen 1996; Olshausen 2003b; Lewicki and Olshausen 1999), we will derive a probabilistic generative model for computing image codes. Additionally, we can follow a similar derivation to learn a dictionary that maximizes coding efficiency. This gives us a principled and extendable foundation from which we can derive sparse coding and similar models.

Inference

Our model assumes a prior probability over the activity coefficients, $p(a)$. We also assume that the coefficient prior does not depend on the dictionary itself; that is $p(a|\Phi) = p(a)$. Although in practice the coefficients and dictionary will be jointly optimized, we can adhere to this last assumption by fixing one variable while optimizing the other in an expectation-maximization-like procedure. We specify the likelihood of an image under the model for a given state of coefficients as $p(s|a, \Phi)$. Conversely, given evidence in the form of an image sample, we can define the posterior as the conditional probability distribution of our coefficients given the image and model: $p(a|s, \Phi)$. Our goal is to maximize this posterior distribution, which gives us the most probable estimate of the coefficients. As is done in (Lewicki and Olshausen 1999), we can relate the likelihood and posterior using Bayes' rule:

$$p(a|s, \Phi) \propto p(s|a, \Phi)p(a), \tag{2.2}$$

where we use $\propto$ to indicate that we are ignoring the image probability because it does not affect inference or learning. We model the noise distribution as Gaussian, which we justify by assuming that the generative model can account for everything except for random, uncorrelated (i.e. independent) causes that combine in the limit to follow a Gaussian distribution. Under this assumption, we can specify the log probability density function (up to additive constants) for our image likelihood:

$$\log p(s|a,\Phi) = -\frac{1}{2\sigma_{\mathrm{n}}^2}\|s-\phi a\|^2 + c, \tag{2.3}$$

where σ_{n} is the standard deviation of the additive noise. We will later drop the additive constants, c, as they do not effect inference or learning. Our goal is to encode the images as efficiently as possible. To do this, we impose a factorial (i.e. independent) prior on the coefficients that is highly peaked at 0, which encourages most of the neurons to be silent for any given input signal. We assume that our activations, a, are independent and positive-only with finite mean. The maximum entropy distribution for such a random variable is a one-sided Laplacian, with log probability density

$$\log p(a) = -\lambda \sum_i |a_i| + c, \tag{2.4}$$

where λ is the inverse of the mean. Now that we have densities for our log probabilities, we can write down the log-likelihood of our model up to additive constants, L, which we will later define as the negative of the sparse coding energy function:

$$\begin{aligned} L &= \log\left(p(s|a,\Phi)p(a)\right) \\ &= \log p(s|a,\Phi) + \log p(a) \\ &= -\frac{1}{2\sigma_{\mathrm{n}}^2}\|s-\Phi a\|^2 - \lambda\sum_i |a_i|. \end{aligned} \tag{2.5}$$

As we said earlier, we wish to maximize the posterior distribution to infer a sparse code. That is, we wish to do a maximum a-posteriori (MAP) estimate. To do this we can find a minimum of the negative log-likelihood:

$$\begin{aligned} \hat{a} = \max_a p(a|s,\Phi) &= \max_a \left[\log p(s|a,\Phi) + \log p(a)\right] \\ &= \min_a \frac{1}{2\sigma_{\mathrm{n}}^2}\|s-\Phi a\|^2 + \lambda\sum_i |a_i|. \end{aligned} \tag{2.6}$$

Learning

The probability of the image under the model, $p(s|\Phi)$, is obtained by marginalizing over the activations:

$$p(s|\Phi) = \int p(s|a,\Phi)p(a)da. \tag{2.7}$$

To find the optimal dictionary, we have to maximize the model posterior. This is equal to the image likelihood times the weight prior and tells us how good our model is given some data samples:

$$p(\Phi|s) \propto p(s|\Phi)p(\Phi). \tag{2.8}$$

We impose an l_2 prior on the pixel dimension of the dictionary, which we write down as a log density:

$$\log p(\Phi_i) = -\frac{1}{2\sigma_\Phi^2}\sum_i \|\Phi_i\|^2 + c, \tag{2.9}$$

where σ_Φ^2 is proportional the expected pixel variance of the individual weight vectors and the sum penalizes each weight individually. We've now defined our entire log posterior:

$$\log p(\Phi|s) = \log p(\Phi) + \log \int p(s|a,\Phi)p(a)da, \tag{2.10}$$

where the integral tells us that we want to do inference. Ideally, we would like to maintain a distribution of probable coefficients for a given image and model. However, to do this exactly is intractable and so is sampling from the coefficients to approximate the integral. Instead, we will again use a MAP estimate, which is a good approximation if the model is sure of the posterior. That is to say that $p(a|s,\Phi)$ truly follows a highly peaked distribution with heavy tails (i.e. the distribution has high kurtosis). This gives us

$$\begin{aligned}
\log p(\Phi|s) &= \log p(\Phi) + \log \int p(s|a,\Phi)p(a)da \\
&\approx \log p(s|\hat{a},\Phi) + \log p(\hat{a}) + \log p(\Phi) \\
&= -\frac{1}{2\sigma_\text{n}^2}\|s-\Phi\hat{a}\|^2 - \lambda\sum_i |\hat{a}_i| - \frac{1}{2\sigma_\Phi^2}\sum_i \|\Phi_i\|^2,
\end{aligned} \tag{2.11}$$

where $\hat{a}$, as above, is a MAP estimate of the posterior. Now we can take the gradient with respect to our model to get the weight update:

$$\nabla_\Phi \log p(\Phi|s) \approx \frac{1}{\sigma_\text{n}^2}[s-\Phi\hat{a}]\hat{a}^\top - \frac{1}{\sigma_\Phi^2}\|\Phi\| \tag{2.12}$$

As we will see in the following section, the learning and inference rules can be derived directly from the same energy function, which is the negative of equation (2.5). In practice, instead of imposing a prior on the weights, we will force them to have unit l_2 norm by normalizing them after each weight update. This ends up being equivalent to doing projected gradient descent on the energy function. From this point we can make a simple change to the inference rule to define the LCA. Finally, we will discuss some important variants to the LCA, analyze the network trained on natural images, and derive comparisons to similar models.

2.3 Sparse coding

In practice, sparse coding on images involves the tasks of learning a dictionary from data and, given a dictionary, finding an optimal sparse code for an input signal. We define an optimal sparse code as one that gives the most faithful representation of the data using the fewest dictionary elements. The primary objective of sparse coding is to encode input data (e.g. an image of a natural scene) in a generative framework, such that the data can be reconstructed from the code (Olshausen and Field 1997). We want the encoding to be of a higher dimensionality than the input image; that is to say we want our dictionary to be overcomplete.

For an N-dimensional input signal (e.g. an N pixel image), $s \in \mathbb{R}^N$, we want a dictionary of $M > N$ vectors, where each vector is of length N. In other words, the dictionary is a matrix Φ with dimensions $N \times M$. Each of the M columns are dictionary elements, $\Phi_i, i \in \{1, ..., M\}$, and each dictionary element has a corresponding coefficient, $a_i, i \in \{1, ..., M\}$, that we can assemble into a vector of activations. Our goal is to minimize an energy function that requires both a faithful and efficient representation of the data, in that it reconstructs the data well with the fewest number of active elements. We can express this mathematically as

$$\underset{a,\Phi}{\operatorname{argmin}} \left(E = \overbrace{\tfrac{1}{2}\|s - \hat{s}\|_2^2}^{\text{Preserve Information}} + \overbrace{\lambda \sum_{i=1}^{M} C(a_i)}^{\text{Limit Activations}} \right), \tag{2.13}$$

where $\hat{s} = \sum_{i=1}^{M} \Phi_i a_i$ is the image reconstruction, $\underset{a,\Phi}{\operatorname{argmin}}$ tells us that we want to minimize the energy with respect to both the activations and the dictionary, and $|| \cdot ||_2^2$ indicates the squared l_2 norm. In this energy function, we use λ as a trade-off parameter between the reconstruction quality and the sparsity constraint. Note that we have performed a change of variables from the previous section, to include the noise variance, σ_{n} into a new λ parameter that now modulates the l_1 cost as a Lagrange multiplier. In practice we tune it to maximize network sparsity for a minimum accepted reconstruction quality. One popular choice for the cost function, $C(\cdot)$, is the l_1 cost penalty, which is the sum of the absolute value of all activations:

$$\sum_{i=1}^{M} C(a_i) = \sum_{i=1}^{M} |a_i|. \tag{2.14}$$

Depending on the choice of cost function, solving equation (2.13) can be convex or non-convex. For the l_1 cost, it is convex. Previous work has solved this via alternating projected gradient descent (Olshausen and Field 1997) or basis pursuit denoising (S. S. Chen et al. 2001). The basis pursuit denoising variant is often accomplished using the iterative shrinkage and thresholding algorithm (ISTA) (Daubechies et al. 2004; Beck et al. 2009). Rozell et al.

(2008) proposed an alternative to ISTA using a neural network following dynamics governed by the energy gradient called the Locally Competitive Algorithm (LCA). The following section explains this alternative in detail. It is also possible to consider an l_0 like cost, which measures the total number of active neurons and is non-convex. This can be approximately solved by matching pursuit (Davis et al. 1997; Rehn et al. 2007; Rebollo-Neira et al. 2002), or with a particular "hard thresholded" variant of the LCA (see section 2.5).

2.4 The Locally Competitive Algorithm

The Locally Competitive Algorithm (LCA) is a method for computing a sparse code from a given input signal and dictionary and was original described by Christopher J. Rozell and colleagues (2008). The model describes an activation coefficient, a_k, as the thresholded output of some model neuron's internal state, u_k, which is analogous to the neuron's membrane potential. This internal state follows similar dynamics to a leaky integrator neuron model. The LCA can be described as a continuous system that is implementable in hardware, or as a discrete system with an internal state that advances as a function of a time step. If implemented in hardware, you can follow current dynamics (Kirchhoff's law for an equivalent RC circuit) to perform inference extremely quickly (Rozell et al. 2008). As the model state advances, the system relaxes to a minimum solution of the energy function described in equation (2.13). Figure 2.1 gives a schematic drawing of the LCA network. In this section we will derive the dynamical equation for computing the state transitions from the sparse coding energy function.

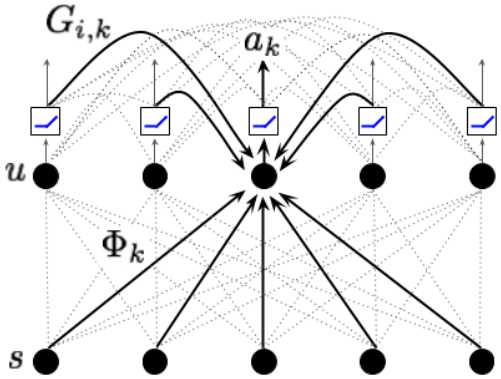


Figure 2.1: **LCA architecture.** The LCA network includes feedforward driving connections, Φ as well as a lateral connectivity matrix, G. The input, s, is a vector of pixels, which is the lower row of black dots. The neurons are the upper row of black dots and have internal states, u. The dark arrows are connections for neuron k. All other connections are denoted by dotted lines.

We wish for our neuron activations to minimize the energy function described in equation (2.13). This will be accomplished via gradient descent. To illustrate the derivative, we first

express the energy function in subscript notation:

$$E(t) = \frac{1}{2}\sum_{i=1}^{N}\left[s_i - \sum_{j=1}^{M} a_j(t)\Phi_{i,j}\right]^2 + \lambda\sum_{j=1}^{M} C(a_j(t)) \tag{2.15}$$

and then we take the derivative with respect to an individual neuron's activity, $a_k(t)$. Since we ultimately want to do gradient *descent*, we will write the negative derivative:

$$\begin{aligned} -\frac{\partial E(t)}{\partial a_k(t)} &= \sum_{i=1}^{N} S_i\Phi_{i,k} - \Phi_{i,k}\sum_{j=1}^{M} a_j(t)\Phi_{i,j} - \lambda\sum_{j=1}^{M}\frac{\partial C(a_j(t))}{\partial a_k(t)} \\ &= \sum_{i=1}^{N}\left[s_i\Phi_{i,k} - \sum_{j\neq k}^{M}\Phi_{i,k}\Phi_{i,j}a_j(t) - \Phi_{i,k}\Phi_{i,k}a_k(t)\right] - \lambda\frac{\partial C(a_k(t))}{\partial a_k(t)} \end{aligned} \tag{2.16}$$

We constrain the model to have unit norm dictionary elements in the pixel dimension, $||\phi_i||_2^2 = 1$. This means that the $\sum_{i=1}^{N}\Phi_{i,k}\Phi_{i,k}a_k(t)$ term in equation (2.16) reduces to $a_k(t)$:

$$-\frac{\partial E(t)}{\partial a_k(t)} = \sum_{i=1}^{N}\left[s_i\Phi_{i,k} - \sum_{j\neq k}^{M}\Phi_{i,k}\Phi_{i,j}a_j(t)\right] - a_k(t) - \lambda\frac{\partial C(a_k(t))}{\partial a_k(t)} \tag{2.17}$$

$E(t)$ and $a(t)$ both vary in time, but in future sections we will drop the time variable to improve the equation readability. In this scenario, we can imagine a constant input signal, s, and a constant set of dictionary elements, Φ. Given these constants, we want the system to evolve over time to produce an optimal set of activations, a. Equation (2.15) gives us the energy at each time point during this inference process. Each neuron has a corresponding dictionary element, ϕ_k, which is a vector that indicates the connection strength to each pixel in the input. In the literature this is also referred to as the strength of the synaptic connections between the neuron and each input pixel, or the neuron's feed-forward receptive field. In this model, we are going to find a sparse code for each individual image patch. Additionally, all M neurons will be connected to the same image patch, s (i.e. the model is "fully-connected"). The model neuron has a driving excitatory input, which we will denote $b_k = S\phi_k = \sum_{i=1}^{N} s_i\Phi_{i,k}$. This is the projection of the input image onto the neuron's corresponding dictionary element. The stronger the similarity between the input, s, and the dictionary element, ϕ_k, the stronger the driving excitatory input. Each neuron also receives inhibition from every other active neuron. The magnitude of inhibition is partially set by an $M \times M$ Gramian matrix, G. The matrix evaluates to the inner product of the each neuron's dictionary element with all other neurons' dictionary elements, $G = \Phi^T\Phi$, such that each element in G gives the overlap in pixel space between two dictionary elements, $G_{i,j} = \sum_{n=1}^{N}\Phi_{n,i}\Phi_{n,j}$. The total inhibition onto a neuron is also scaled by how active the inhibiting neuron is.

Next, we define a new function that maps the activity for a neuron and the sparsity cost function to a scalar. This function can be thought of as the self inhibition that increases as the value of a_k increases:

$$f_\lambda(a_k(t)) = a_k(t) + \lambda \frac{\partial C(a_k(t))}{\partial a_k(t)} \tag{2.18}$$

Self inhibition imposes sparsity by penalizing the neurons own output, which is different from the sparsity inducing input from other laterally connected neurons. Now we can write the partial derivative of our energy function (equation (2.15)) in terms of our new variables:

$$-\frac{\partial E(t)}{\partial a_k(t)} = b_k - \sum_{j \neq k}^{M} G_{k,j} a_j(t) - f_\lambda(a_k(t)). \tag{2.19}$$

At this point we could update $a(t)$ using gradient descent following equation (2.19) to produce a sparse code from an input signal. However, a more biologically consistent solution would be to have the model neuron maintain an internal state, analogous to a biological neuron's membrane potential. The model neuron could then only produce output when its membrane potential exceeded some threshold. This has better correspondence to biology and it gives the neurons sub-threshold dynamics that influences their activity (see section 2.5) when compared to directly using equation (2.19). Following this logic, Rozell et al. (2008) define an internal state variable, $u_k(t)$ that represents the membrane potential for neuron k at time t. When a neuron's potential climbs above some threshold, it communicates in the form of an activation, $a_k(t)$, which is analogous to a spike rate[1]. Only these excited neurons can contribute to the image code and reconstruction. Ultimately, the network of neurons should still descend the energy function from equation (2.15), so we define the neuron's state dynamics to be governed by the following equation:

$$\begin{aligned} \dot{u_k}(t) &\propto -\frac{\partial E(t)}{\partial a_k(t)} \\ \dot{u_k}(t) &= \frac{1}{\tau}\left[b_k - \sum_{m \neq k}^{M} G_{k,m} a_m(t) - f_\lambda(a_k(t))\right], \end{aligned} \tag{2.20}$$

where τ is a proportionality constant that represents the time constant of the dynamics.

In order to have a complete description of the model, we need to describe a relationship between u and a. Earlier the $f_\lambda(a_k(t))$ term was described as a self inhibition term that

[1]Neuroscientists often use the spike rate, or number of spikes per fixed unit of time, as a measure of the overall activity of a neuron. The LCA is not a spiking network, but when comparing to biological neurons (e.g. in the work by Zhu et al. 2013 and in section 4.3) we make a direct analogy between the activity of the LCA neuron and the biological neuron's spike rate. Spiking versions of sparse coding have been explored by (Zylberberg, Murphy, et al. 2011; Olshausen 2003a).

encourages sparsity. Another way to enforce self inhibition is to introduce a membrane leak term. If we assign the internal state, $u_k(t)$, to this function:

$$u_k(t) = f_\lambda(a_k(t)), \tag{2.21}$$

then we can think of our neuron as a leaky integrator. We can also invert the function to get our neuron's output activity:

$$a_k(t) = f_\lambda^{-1}(u_k(t)) := T_\lambda(u_k(t)).$$

This gives us the LCA neuron update equation:

$$\dot{u_k}(t) = \frac{1}{\tau}\left[\overbrace{b_k}^{\text{Driving excitation}} - \overbrace{\sum_{m\neq k}^{M} G_{k,m}a_m(t)}^{\text{Lateral Inhibition}} - \overbrace{u_k(t)}^{\text{Leak}}\right], \tag{2.22}$$

or equivalently

$$\tau\dot{u_k}(t) + u_k(t) = b_k - \sum_{m\neq k}^{M} G_{k,m}a_m(t).$$

When the neuron's membrane potential passes over a threshold, defined by $T_\lambda(u_k(t)) = f_\lambda^{-1}(u_k)$, it becomes active:

$$a_k(t) = T_\lambda(u_k(t)). \tag{2.23}$$

For this expression to perform gradient descent on the energy function, a must be a monotonically increasing function of u. Rozell et al. (2008) describe the relationship between the sparseness cost penalty, the neuron activity, and the internal membrane potential via a thresholding function. The thresholding function can take various forms that determine the exact nature of the sparseness penalty. For the l_1 penalty, Rozell et al. (2008) use a soft thresholding function:

$$T_\lambda(u_k(t)) = \begin{cases} u_k(t) + \lambda, & u_k(t) < -\lambda \\ 0, & -\lambda < u_k(t) < \lambda \\ u_k(t) - \lambda, & u_k(t) > \lambda \end{cases} \tag{2.24}$$

Here λ indicates the sparseness penalty trade-off as well as the threshold that the membrane potential must surpass for the neuron to become active. An illustration of how one gets to the thresholding function from the l_1 penalty is given in figure 2.2. With the internal state dynamics from equation (2.22) and the thresholding function in equation (2.24), we have defined a network that settles to a sparse code, a, which represents the input signal. For all of the work in this thesis, we will diverge from the implementation in (Rozell et al. 2008)

and implement a rectified version of the thresholding function. That is, from equation (2.24), the first term for when $u_k(t) < -\lambda$ is set to $T_\lambda(u_k(t)) = 0$. This adds a higher degree of nonlinearity, and is important for implementing hierarchical sparse coding models.

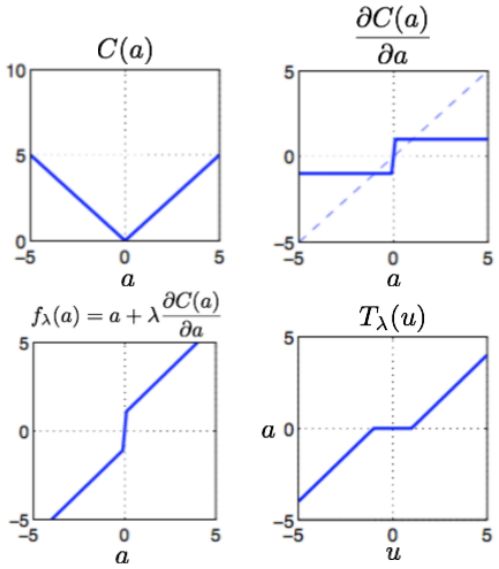


Figure 2.2: **Deriving the LCA thresholding function.** Starting from a pictorial view of the l_1 cost function, one can derive the self inhibition term from equation (2.21) and invert it to see the thresholding function described in equation (2.24). The value of λ dictates the range of allowable sub-threshold membrane potentials, u_k, before the neuron becomes active.

In addition to finding a sparse code, we are also interested in learning a set of dictionary elements, Φ. This can be done by performing gradient descent on equation (2.13) using the coefficient values obtained with the LCA. This yields

$$\Delta\phi_k = \eta(s - \hat{s})a_k, \tag{2.25}$$

where η is the learning rate and $\hat{s}$ is the image reconstruction. To recap, for a given image sample we first find our image code, a, for a fixed dictionary, Φ, and then using that code to update the dictionary with equation (2.25).

The Convolutional LCA

In this thesis, unless otherwise noted, it should be assumed that we are using the fully-connected variant of the LCA, in which each neuron is connected to every pixel in the input. However, the LCA can also be easily extended to be convolutional to scale it up to larger images. There are benefits to using convolutional over fully-connected LCA: it is more amenable popular GPU computing architectures, it converges to lower minima, and it is more stable when encoding video. However, some drawbacks are that it no longer has a straight

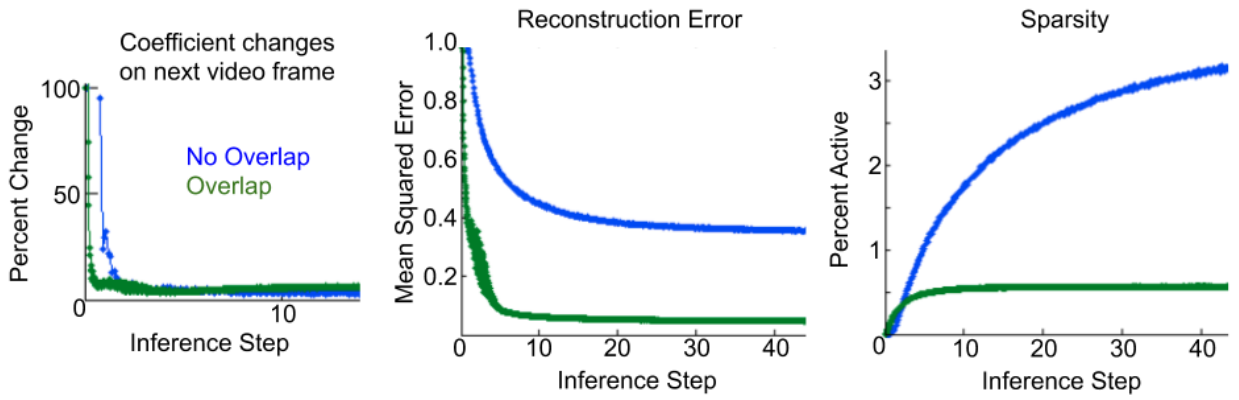


Figure 2.3: **The convolutional LCA has unique coding properties.** The convolutional LCA (overlap, green lines) has better encoding properties than the patch-based LCA (no overlap, blue lines) in terms of stability and energy. We hypothesize that this is an effect of global competition across the image that is afforded by the convolutional architecture. Note that the patch-based LCA is still sharing weights between patches; it is acting as if each patch was encoded independently with the same model. For video input, we recorded the percent change in coefficients from one frame to the next as a function of the number of inference steps performed on each frame. The left plot shows that convolutional variant is more stable earlier in inference, and approximately equally stable after both variants have converged. Additionally, the convolutional LCA settles to a lower mean squared error between the input and reconstruction (middle plot), and is more sparse (right plot) than the patch-based model. This figure was reproduced from unpublished work done with the Kenyon lab (Paiton, Brumby, et al. 2013).

forward analog hardware implementation and it diverges from the type of connectivity we see in the brain. A possible better alternative to convolution would be to use local (i.e. tiled) connectivity, as was done in (Le, Ranzato, et al. 2011) and (Ngiam et al. 2010). At the time of writing we are unaware of any attempt to use this type of connectivity with the LCA.

We will occasionally use the convolutional LCA to scale up experiments to larger inputs. For a neural-network oriented introduction to convolution, see (I. Goodfellow et al. 2016). The convolution will be parameterized by the kernel size, number of kernels, and stride. If the stride is equal to the patch size (i.e. no overlap), then this model implements the regular patch-based LCA that operates on an entire image worth of patches independently, but simultaneously.

For the convolutional LCA, we write our energy function in terms of a "deconvolution", or transposed convolution (Zeiler et al. 2010):

$$E = \frac{1}{2}||s - a \circledast \Phi||_2^2 + \lambda||a||_1, \tag{2.26}$$

where $\circledast$ is the transposed convolution operator. We also rewrite the membrane update equation to be in terms of a residual error convolved with the weight vector:

$$\tau \dot{u}_k + u_k = e * \Phi_k + a_k, \tag{2.27}$$

where $*$ is convolution and $e = s - \hat{s}$ is the residual error. In this variant, the driving force to the neurons is the error, instead of the b_k term in equation (2.19). This means that every neuron that is competing for the image reconstruction will contribute to driving neuron k, even if it is in a different convolutional position. The effects of this global competition across the image appear to give the model an advantage in terms of finding an energy minima and stability, which we demonstrate in figure 2.3.

A note on convolutional overcompleteness

In patch-based sparse coding, we compute the network overcompleteness by dividing the number of neurons by the number of pixels. However, with convolutional sparse coding there is a caveat that the neighboring inputs have shared weights. Therefore, directly expressing the overcompleteness as number of outputs / number of inputs misrepresents the number of unique weights in the model. Even more nuanced is that the LCA typically learns weights that tile all possible positions in the image patch, which is not as necessary with a convolutional model since the convolution operation translates weights across the image. As a result, all weights learned with convolution tend to be centered in their patch. With all of this in mind, we will denote convolutional overcompleteness as the number of kernels divided by the product of the horizontal (x) and vertical (y) strides: $o = \frac{f}{s_x * s_y}$. Therefore, within reasonable parameter ranges, you can effectively increase the overcompleteness by increasing the number of kernels or by decreasing the stride, which is important when considering computational constraints (Schultz et al. 2014). From this we can also conclude that overcompleteness is not dependent on the patch size, which has been exploited for learning disparity selective neurons that require large receptive fields (Lundquist et al. 2016).

Postdictions from the LCA

In their landmark paper, Olshausen and Field (1996) show that the sparse coding model learns oriented, band-pass receptive fields. These are well matched to fits of the linear approximations of receptive fields of mammalian V1 simple cells obtained via spike-triggered averaging (Hateren and Schaaf 1998). They argue that this supports the hypothesis that the V1 neural population encodes visual stimulus using an efficient wavelet-like dictionary. This result has been replicated a number of times with various types of sparse coding models (Zylberberg, Murphy, et al. 2011; Zylberberg and DeWeese 2013; Rehn et al. 2007). As a followup to this study, (Zhu et al. 2013) show that the LCA network also exhibits a variety of non-linear receptive field effects, including end-stopping, cross-orientation inhibition, and surround suppression. This is an important finding, as it shows that a single LCA network trained on natural scenes matches many linear and non-linear empirical effects found with biological neurons. Most studies of these non-linear effects employ unique models that are extended from the classic linear/non-linear neuron model for each effect found, while Zhu

and Rozell are able to demonstrate many effects with a single model. In chapter 4 we will argue that these effects result from the explaining away process during sparse inference. In (Vinje et al. 2000), the authors find that stimulating the area outside the classic receptive field (non-classical receptive field, nCRF) with naturalistic stimuli causes pairs of V1 neurons to become more decorrelated in their response. They observed "dramatically increased sparseness" when stimulating nCRFs, and suggest that a consequence of this "is increased independence of the responses across cells".

As we will expand on in chapter 4, the LCA makes explicit predictions about the existence (and necessity) of population non-linearities among V1 neurons, which facilitate unique responses to natural image stimuli. The model gives important attribution for inhibitory lateral connectivity in V1 (Zhu et al. 2015). In addition to the obvious energy saving motivation for sparse activity, it has been argued that limiting the number of active neurons can result in a more explicit code that is easier to read out by downstream neurons (Olshausen 2003b). We will explore this in more detail in later chapters.

2.5 Properties of the LCA trained on natural images

Features learned

It has been shown previously that sparse coding will learn a dictionary of features that tile spatial position, frequency, and orientation (Olshausen and Field 1996; Olshausen and Field 1997). We reproduce this finding while also varying the overcompleteness of the model in figure 2.4. We find that as overcompleteness increases, the density of the tiling also increases. Additionally, the model learns higher spatial frequency features and a more even distribution of orientations when more overcomplete. It is also interesting that the models tend to learn significantly more features that are aligned with the vertical and horizontal axis of the input images, which is well matched to image statistics (Switkes et al. 1978; Torralba et al. 2003). The spatial frequencies also have anisotropy across the vertical axis, which is maintained as you increase overcompleteness. We suspect this is an artifact of the dataset tested, as all three models were trained from the same image patch samples, although we did not test this hypothesis. We also show in figure 2.5 that when we increase overcompleteness, the neurons have more similar receptive fields as indicated by a smaller angle between weight vectors.

Inferring sparse codes of natural images with the LCA

Inference is a critical component of the sparse coding model. It gives the model the ability to "explain away" causes of the input signals (Olshausen and Field 1997) and gives neurons a higher degree of selectivity (see chapter 4). Figure 2.6 shows the values of the loss function in equation (2.13) through inference, demonstrating that the total loss is reduced. Notice that the sparse loss starts at zero, indicating that none of the neurons are contributing to the reconstruction. At the beginning of inference, all of the neurons with weights that have

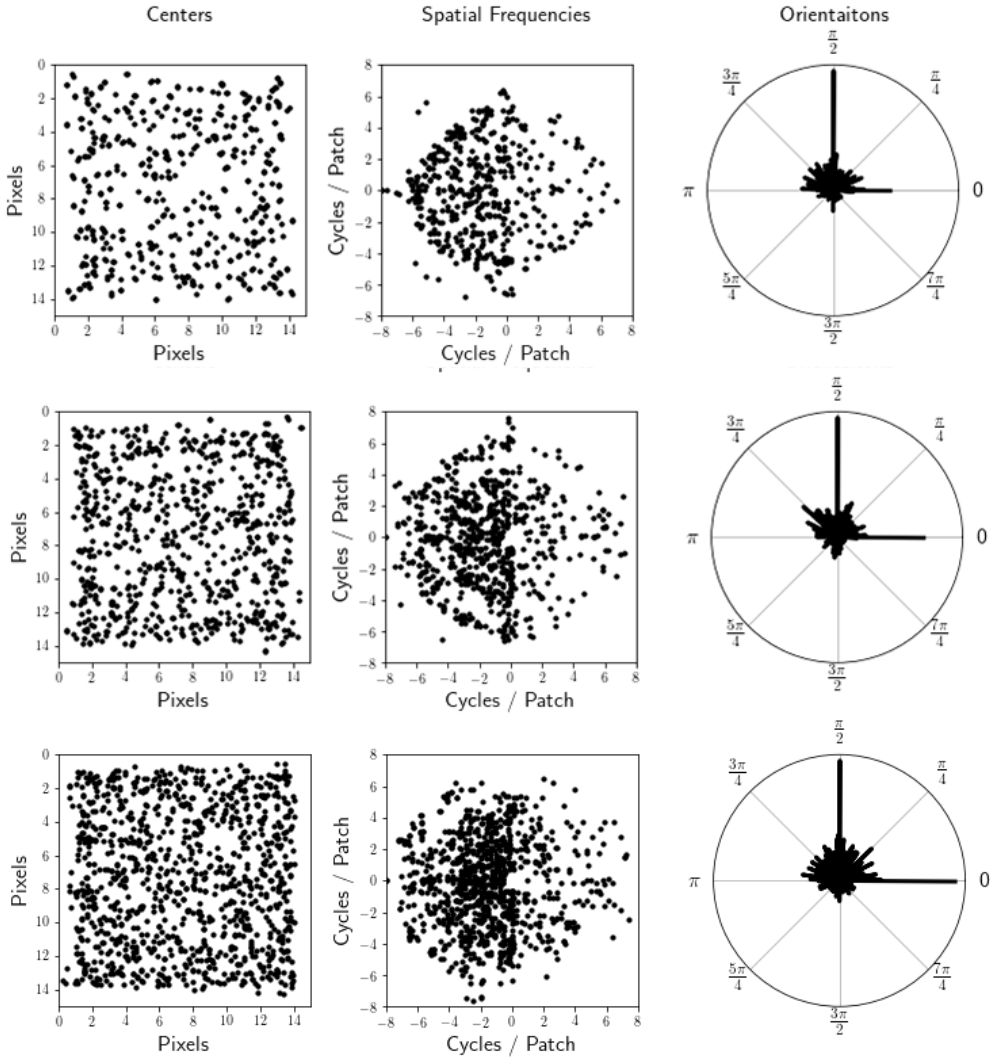


Figure 2.4: **Properties of features learned with LCA.** The LCA learns a diverse set of features that tile important signal characteristics when trained on patches from natural scenes. From top to bottom, the rows represent the LCA with 2, 3, and 4 times more neurons than pixels, respectively. From left to right, the columns are the spatial positions, spatial frequencies, and orientations of each function.

a non-zero inner product with the input will increase their membrane potential without producing any output and without receiving any inhibition from other neurons. It is only after a neuron passes threshold that it starts producing outputs and inhibiting its neighbors. These sub-threshold dynamics can result in a better solution than the standard ISTA framework, which turns on all units that have a non-zero inner product with the input at the first inference step (Rozell et al. 2008).

Figure 2.11 gives us an idea of the inference dynamics for a subset of neurons in the LCA

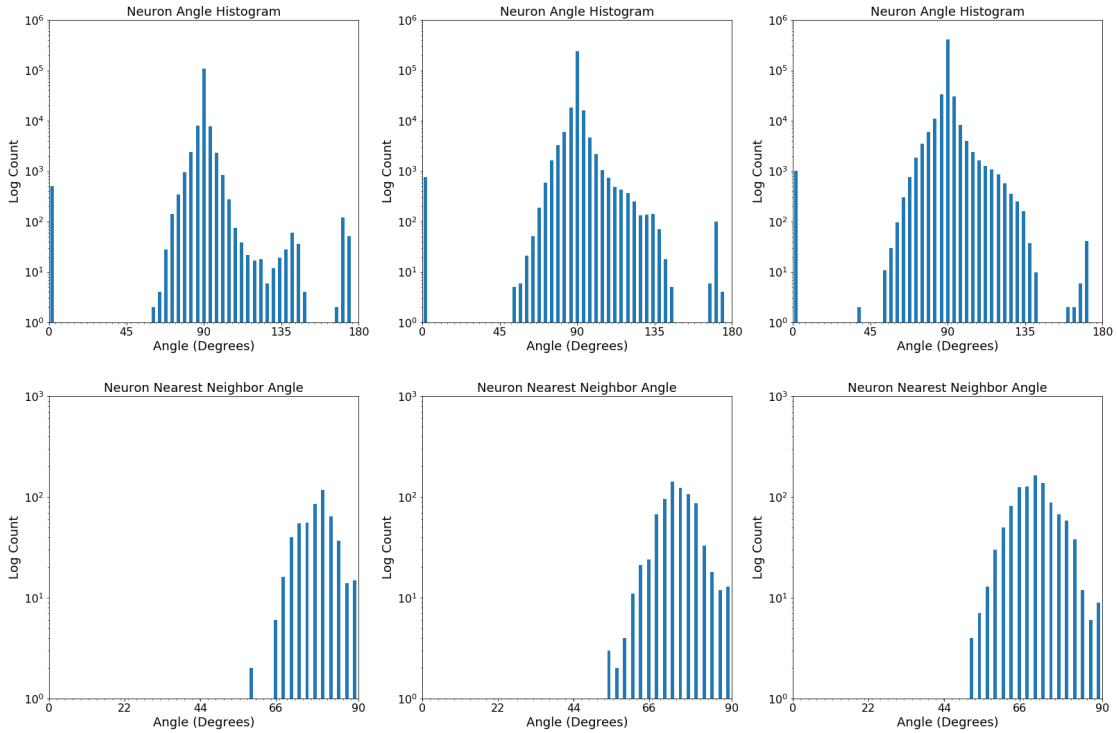


Figure 2.5: **The LCA neuron weights are closer as overcompleteness increases.** The top row are histograms of the angles between all pairs of basis functions. The bottom row are histograms of the angles to the nearest neighbor for each neuron. From left to right, we increase the network's overcompleteness such that there are 2, 3, and 4 times more neurons than pixels, respectively. The histograms shift left as we increase overcompleteness, indicating that the weight vectors are closer in angle. This will increase competition among neurons because the Gramian matrix term in equation (2.22) will have higher values.

network. Each line in the trace plots represents a different term from equation (2.19). Nearly all of the traces follow nonlinear trajectories and have interesting sub-threshold dynamics. Figure 2.10 shows the corresponding weights for each of the neuron traces in figure 2.11.

The hard thresholded LCA has many energy minima

It is possible to define a variety of valid threshold functions for converging LCA dynamics. The two primary functions considered in (Rozell et al. 2008) are termed "soft" and "hard" thresholds, which approximate solutions to the l_1 and l_0 cost functions, respectively. Much of the work herein is focused on the soft threshold function (equation 2.24), primarily for model simplicity. However, the hard thresholded variant is intriguing as it results in a higher degree of non-linearity and solves a harder (non-convex) optimization problem with many equally valid local minima. The rectified hard thresholded LCA model behaves exactly as is

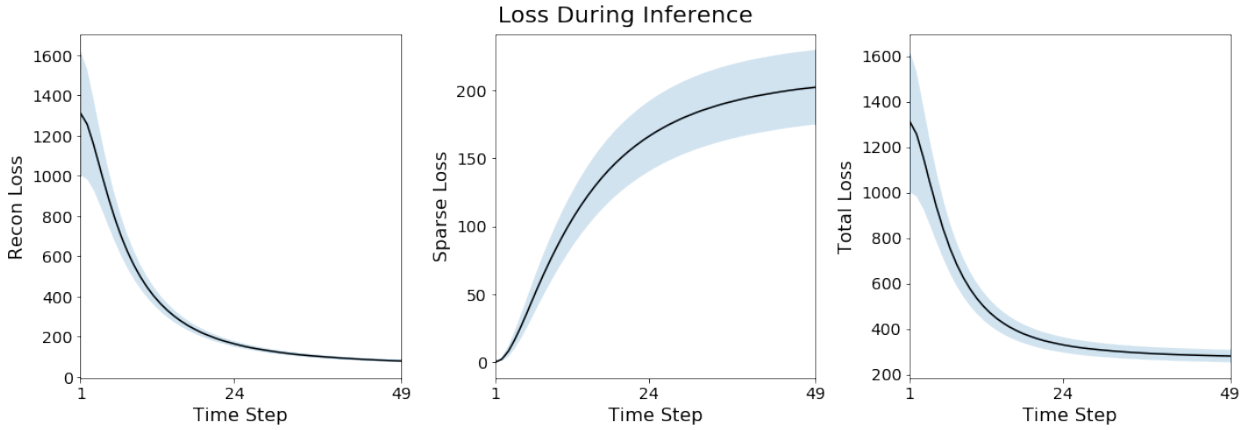


Figure 2.6: **LCA inference minimizes the sparse coding energy function.** The black lines represent the mean loss when encoding 50 natural scenes and the blue background represents the standard error of the mean.

described in section 2.4, except that the thresholding function has been changed to:

$$T_\lambda(u_k(t)) = \begin{cases} 0, & u_k(t) \leq \lambda \\ u_k(t), & u_k(t) > \lambda \end{cases} \tag{2.28}$$

As is described in (Rozell et al. 2008), this variant approximates the l_0 cost function for $C(\cdot)$ in equation (2.13). The hard thresholded LCA performs similarly to the soft thresholded variant, except that there is no longer a convex energy landscape with a single global minimum. After some number of inference steps, the LCA settles to a stable minima, but in (Shainin et al. 2016) we show that the minima reached by a convolutional, rectified, hard thresholded LCA is not a global minima. It is not clear how the amount of information differs between these minima. However, it has been proposed that the brain could represent likely events in a probabilistic framework (Lee et al. 2003) and that sampling could be a mechanism for decoding signals in higher cortical areas (Hoyer et al. 2003).

To better understand the energy landscape, we sampled a number of approximately equivalent minima (in terms of energy) for sparse codes produced from weights trained on the CIFAR10 dataset (Krizhevsky and Hinton 2009). We performed this sampling by first allowing the network to settle to a minima for a given image, then perturbing the neuron activations with Gaussian noise, then allowing it to settle again, and repeating for some fixed number of perturbations. We measured the distances between fixed points by counting the number of neurons that changed from active to inactive, which is equivalent to the Hamming distance between binarized (changed threshold or did not) vectors. Figure 2.7 shows that the Hamming distance from the original fixed point to the perturbed fixed point grows as we continue to perturb the network until it reaches a plateau. The figure also shows that the network *always* settles on a different minima.

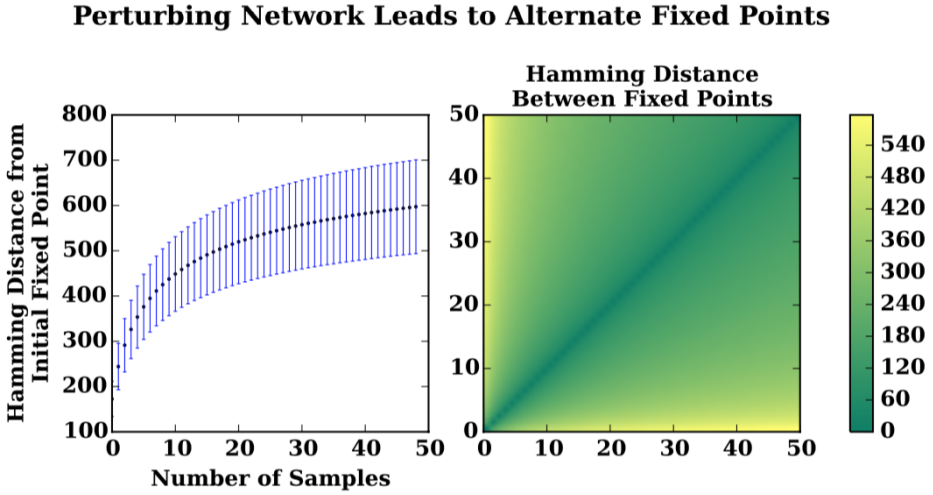


Figure 2.7: **Fixed points of the hard thresholded LCA.** The hard thresholded LCA has a number of valid fixed points in terms of the energy defined in equation (2.13). If you allow the model to settle to a minima, and then perturb the latent code with Gaussian noise, it settles to a different minima. On the left we plot the Hamming distance between codes from the original fixed point to the perturbed fixed point after many perturbations. The Hamming distance is measured by assessing how many neurons crossed threshold from active to inactive or vice versa. On the right we plot the Hamming distances between all pairs of fixed points for all perturbations. All of the off-diagonal values are above 0, which indicates that the model always settled to a unique minima after perturbation. This figure was reproduced from unpublished work with permission from (Shainin et al. 2016).

In figure 2.8, we show that the l_0 sparsity of the activations improves as we continue to perturb the input while the reconstruction error remains the same, suggesting that it is finding better minima. To understand the information content of these minima, we trained two layer classifiers on the sparse codes produced after convergence. The first experiment was to average the fixed point vectors together. We show that the classification accuracy increases as we add more fixed points to the average, which implies that there is different information content per fixed point. The second experiment was to search the independent fixed points for whichever produced the highest confidence (in terms of minimum entropy of the classifier's output distribution) and use this single fixed point as input to the network. The right panel in figure 2.8 shows that the classifier accuracy improves as we increase the number of candidates in the search. Additionally, we found that the number of perturbations to achieve a fixed point with highest confidence was highly variable (not shown).

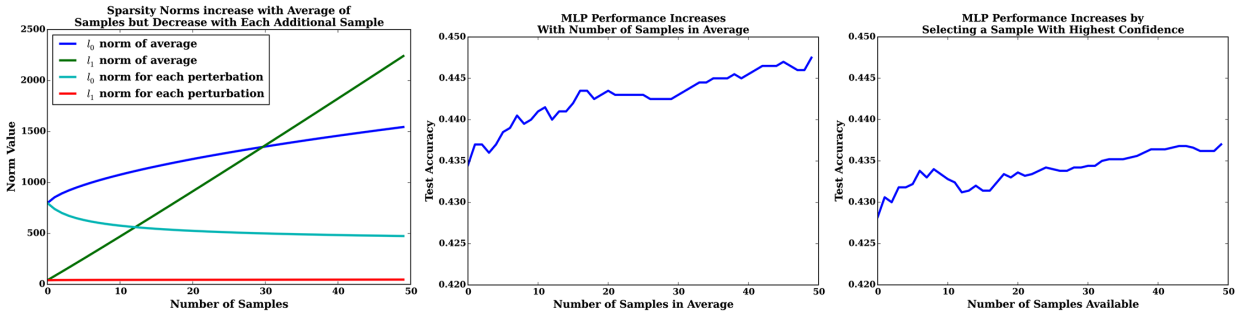


Figure 2.8: **Fixed points have different sparsity and contribute unique information for a classification task.** The plot on the left indicates that new fixed points after perturbing the hard thresholded LCA network tend to have fewer active elements. The blue and green lines show that norms increase when we average together the codes from each perturbation, which is to be expected. The cyan and red curves show that as we continue to perturb the activations, the network settles on a minima with approximately equal l_1 norm and an improved l_0 "norm", or active neuron count. For the middle plot we constructed an averaged vector from each fixed point and used that as the input to a classifier network. The classifier accuracy improves as the number of vectors in the average grows, indicating that there is additional information in the alternate fixed points. As a followup experiment, instead of averaging the fixed points, we searched them to determine which one produces a minimum entropy output from the classifier. This fixed point was then used to determine the class label. The right most figure shows that as you increase the number of samples in the search, the classifier accuracy improves. This figure was reproduced from unpublished work with permission from (Shainin et al. 2016).

2.6 Alternative image coding models

Predictive coding

Here we will draw comparisons between the LCA and the predictive coding model (Rao et al. 1999). The LCA encoder is a single layer predictive coding encoder with a linear synthesis function. This is made intuitive by comparing the LCA against the predictive coding model.

The objective of this investigation is to explore the encoding processes for sparse coding and predictive coding assuming that we are given a trained dictionary. In our sparse coding energy function (equation (2.13)), we use λ as a trade-off penalty between the reconstruction quality and the sparsity constraint. In practice we tune it to maximize network sparsity for a minimum accepted reconstruction quality. In Olshausen and Field (1997), their choice of cost function, $C(.)$, specified a Cauchy distribution for the prior on a:

$$C(a_i) = \log(1 + a_i^2) \tag{2.29}$$

For a learned dictionary, to encode an input image we compute a coefficient vector, a.

Our coefficients (neuron activations) adapt to minimize the energy function described in equation (2.13). The standard way to do this is to perform direct gradient descent on the energy function, following equation (2.16), which we will rewrite here with a small algebraic change:

$$-\frac{\partial E(t)}{\partial a_k(t)} = \sum_i^N \Phi_{i,k}\left(S_i - \sum_j^M a_j(t)\Phi_{i,j}\right) - \lambda \sum_j^M \frac{\partial C(a_j(t))}{\partial a_k(t)} \tag{2.30}$$

A block diagram illustration of this encoding process is given in figure 2.9.

Much like the sparse coding model, the predictive coding model is a neural network that aims to implement efficient coding for natural visual stimuli (Rao et al. 1997; Rao et al. 1999). Although the predictive coding model is described in a more general fashion, the actual implementation used for experiments ends up being strikingly similar to the sparse coding model. Assuming a learned dictionary, the energy equation for the predictive coding model is as follows:

$$E = \tfrac{1}{\sigma_S^2}\|S - \hat{S}\|_2^2 + \tfrac{1}{\sigma_{td}^2}\|a - a^{td}\|_2^2 + \lambda \sum_i^M C(a_i), \tag{2.31}$$

where $a^{td} = f\left(\Phi^{1T} a^1\right)$ represents the top-down feedback and σ_{td}^2 is the expected Gaussian variance of the a^0 estimate. As they describe it, this is a combination of the "sum of squared prediction errors for level 1 and level 2, each term being weighted by the respective inverse variances." The energy function also includes a cost on the activations, which for this example will still be derived from the Cauchy prior. A key difference here between this model and the sparse coding model is how the image approximation, $\hat{S}$ is computed. Rao and Ballard describe a non-linear synthesis function, $f(.)$, which is used in the reconstruction:

$$\hat{S} = \sum_i^M f(\phi_i a_i) \tag{2.32}$$

If $f(.)$ is the identity function (i.e. $f(a) = a$), then we get the sparse coding reconstruction energy term. Rao and Ballard describe this as a "lateral inhibition" model, although the form of synthesis does not actually differentiate between a feedback or lateral inhibition model. In their study, they experiment with both an identity synthesis function, $f(a) = a$, and a hyperbolic tangent synthesis function, $f(a) = \tanh(a)$.

Again we can take the derivative of equation (2.31) to get our update rule. In (Rao et al. 1999) they include a time constant for the iterative update procedure, which we will leave out for clarity.

$$-\frac{1}{2}\frac{\partial E}{\partial a_k} = \frac{1}{\sigma_S^2}\sum_i^N \Phi_{i,k}\left(S_i - \sum_j^M a_j \Phi_{i,j}\right) - \frac{1}{\sigma_{td}^2}\sum_i^M (a_i - a_i^{td}) - \lambda \sum_j^M \frac{\partial C(a_j)}{\partial a_k} \tag{2.33}$$

Comparing equations (2.33) and (2.30), one can conclude that the two differences between the predictive coding model and sparse coding model are the (potential) use of a non-linear synthesis function and a second layer of processing. This is illustrated pictorially in figure 2.9.

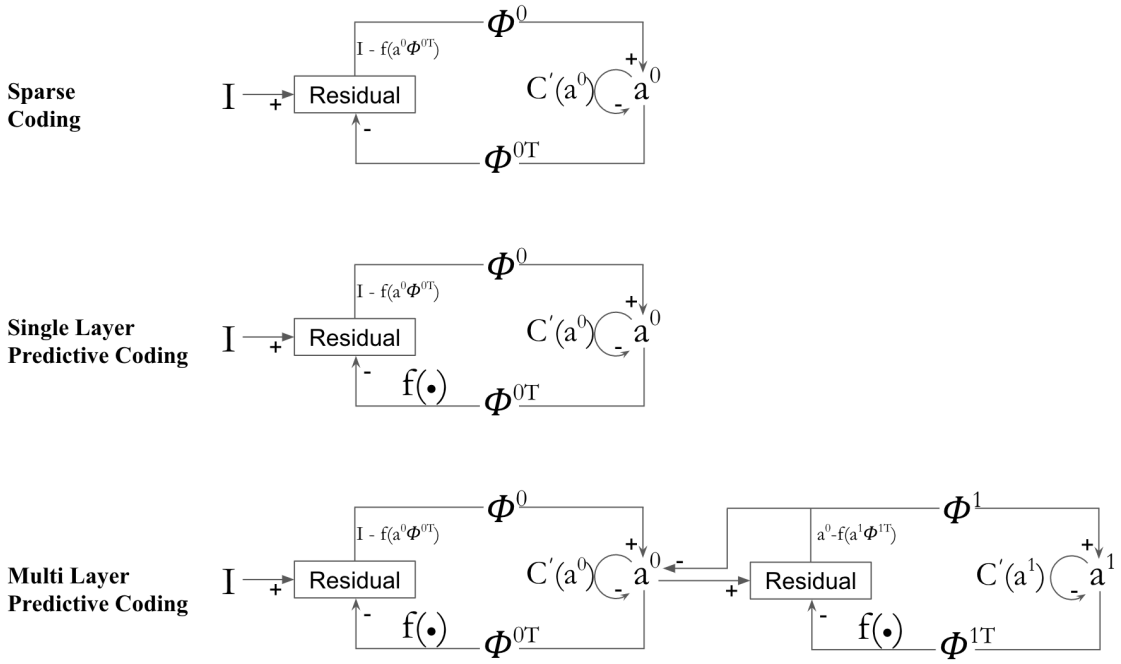


Figure 2.9: **Comparing the LCA to Predictive Coding.** Sparse approximation can be performed via direct gradient descent on the energy function defined in equation (2.13), which we depict as a dynamic process. Adding a non-linearity to the LCA synthesis function results in a network that is equivalent to a single layer of the predictive coding model. The full model combines two such layers (Rao et al. 1999)

ICA

Sparse coding and independent components analysis (ICA) are related algorithms that employ similar linear generative models, but differ critically in their encoding processes. The image code produced by ICA is computed linearly, whereas sparse coding computes the code with a nonlinear inference process. In section 4.3, we show that this nonlinear encoding process confers a considerable advantage in coding efficiency and orientation selectivity.

ICA assumes the generative model

$$s = \Phi a, \tag{2.34}$$

where s is an image, Φ is a matrix of filters, and a is a vector of activations that represents the neural code (Bell et al. 1997). The goal of ICA is to learn a set of statistically independent

filters such that the input data can be reconstructed with minimal error. In ICA, the filter matrix is square and full rank and thus invertible. This allows the ICA activations for a given input to be computed directly as:

$$\hat{a} = \Phi^{-1} s \tag{2.35}$$

During inference, activations are computed with a single, linear, feed-forward operation. The filter matrix is learned via a non-linear, iterative optimization process to accurately reconstruct the input while maximizing the statistical independence of the filters as measured by the joint entropy of the activations. This process results in filters that are optimized for the higher-order statistical structure in natural scenes.

Sparse coding, however, has a highly non-linear encoding process. The overall optimization procedure involves a fast inner loop in which the coefficients are computed for each data vector and a slower outer loop in which the basis functions are adapted to the statistics of the entire dataset. In the inner loop, coefficients are computed considering the prior on their expected distribution. In ICA, however, the prior plays no role in determining the coefficients, but it does still play an important role in the learning the basis functions.

The ICA learning algorithm is simpler and faster than the sparse coding algorithm because the encoding can be computed from the data in a single feedforward pass. The ICA algorithm of Bell and Sejnowski (1997) is formally equivalent to maximum likelihood estimation in the case of no noise and a square system (the dimensionality of the output is equal to the dimensionality of the input). It is easy to generalize this to the case when the number of outputs is less than the number of inputs, but harder the other way around. When the number of outputs is greater than the effective dimensionality of the input the extra dimensions of the output will simply drop out (Livezey et al. 2016; Le, Karpenko, et al. 2011). While this is not as important for blind separation problems where the number of independent sources is less than or equal to the number of mixed signals, it will become a concern in the representation of images, where overcompleteness is a desirable property (Simoncelli, Freeman, et al. 1991). The main difference between the Olshausen and Field (1996) sparse coding model and the ICA algorithm of Bell and Sejnowski (1997) is in the simplifying assumptions they make in order to deal with the intractable integration problem posed by finding the set of basis functions that maximize the likelihood of the inputs given the model defined by those basis functions. Olshausen and Fields algorithm assumes low-noise and a peaky, unimodal distribution on the joint probability between the image and the sparse code given the model in order to justify evaluating it at the maximum. Bell and Sejnowski limit the dimensionality of the code to equal the dimensionality of the input, assume no noise, and enforce an orthogonal set of basis functions so that the integral becomes tractable and can be analytically computed.

2.7 Conclusion

In this chapter we derived the LCA from a probabilistic perspective. We then showed the features learned by the LCA when trained on natural scenes and highlighted some interesting

properties of the inference process. Finally, we compared the LCA to alternative image coding models. In the next chapter we will extend the LCA model to a semi-supervised learning framework and a hierarchical framework. In the following chapter we demonstrate how a geometric analysis technique can be used to explain the LCA's high degree of robustness to noise and selectivity when compared to alternative models.

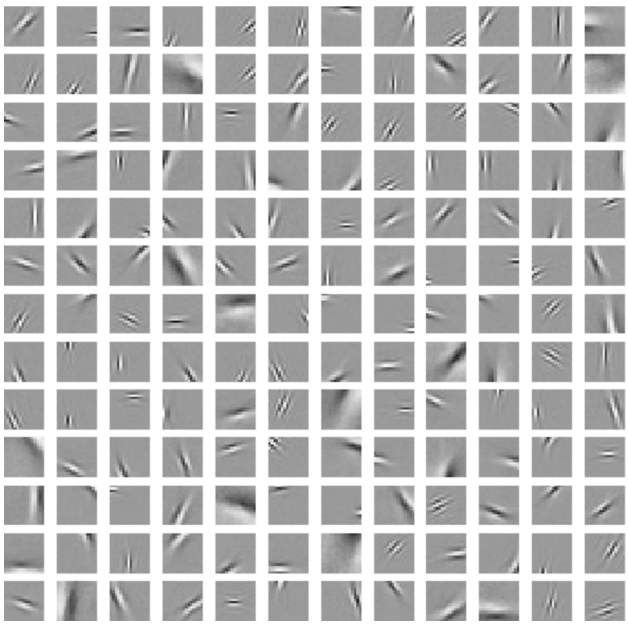


Figure 2.10: **Weights learned when the LCA is trained on natural scenes.** Like other sparse coding variants, dictionary learning with LCA inference produces weights that have a high degree of correspondence to those recorded from mammalian V1 (Zylberberg, Murphy, et al. 2011; Rehn et al. 2007; Olshausen and Field 1997). The weights in this plot correspond to the same neurons as were used to generate the traces in figure 2.11.

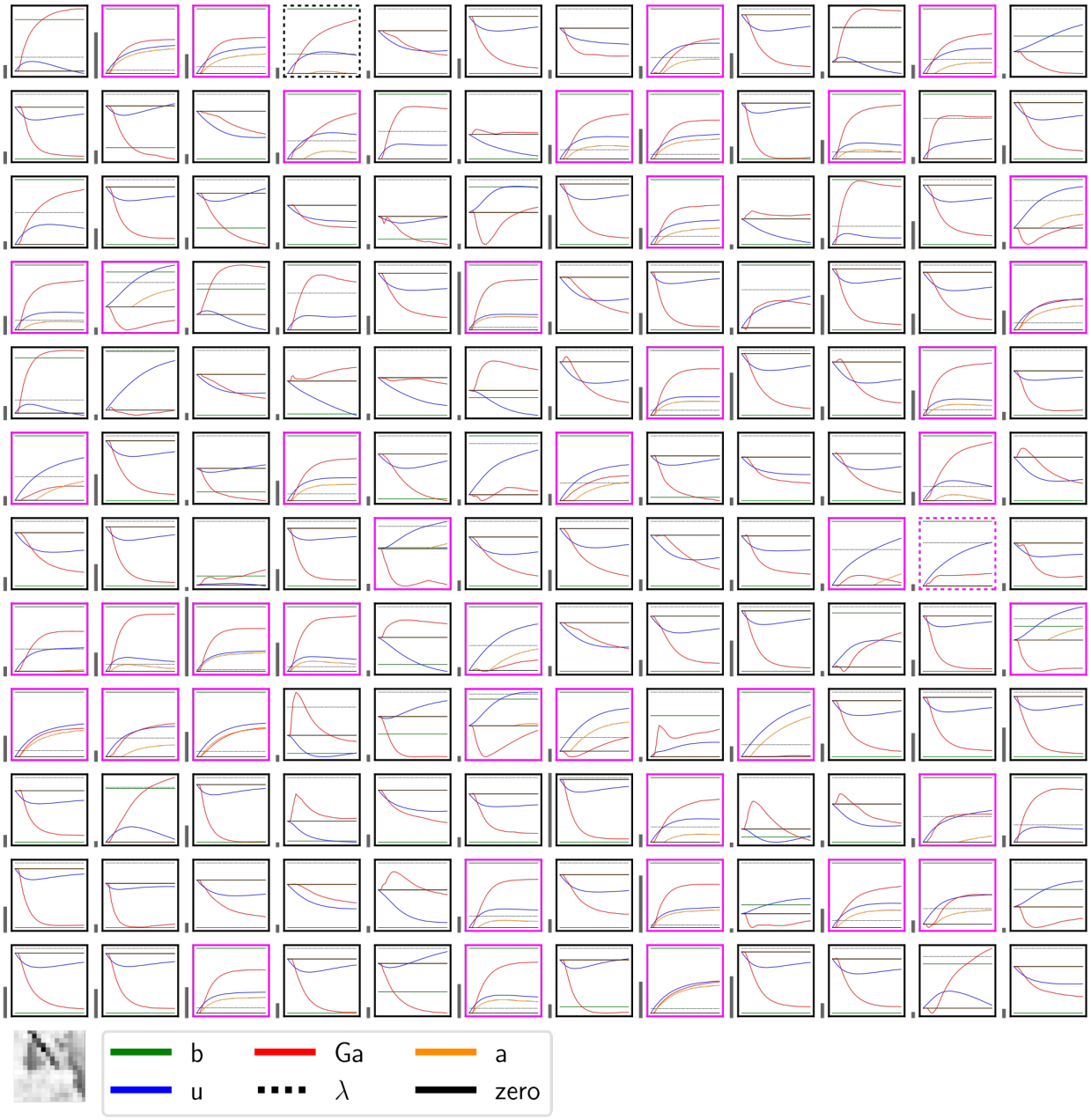


Figure 2.11: **Individual trace dynamics for LCA neurons.** Shown are input and output traces for a subset out of 768 LCA neurons during inference. The input image is in the bottom left. Each trace plot has a corresponding weight, shown in figure 2.10. The lines in the trace plot represent terms from equation (2.19), as indicated in the legend. The grey bars on the left of each trace plot indicate relative scale of the vertical axis. A magenta boundary indicates that the neuron was active at the end of the inference period, and a black boundary indicates otherwise. A dotted boundary indicates that it became active or inactive in the last 20 percent of the inference process.

Chapter 3

Hierarchical extensions of the LCA

3.1 Introduction

We propose that a hierarchical model of natural scenes should produce a general representation of input data that spans all layers. The network should be able to model the causal structure of time-varying natural scenes. Representations at the bottom of the hierarchy should contain information about the details of the scene, and as one ascends the hierarchy, one should see a more general, abstract representation. Information should move up the hierarchy in the form of inference and down the hierarchy as expectations or priors. Resolving ambiguities at the top should be easier as there are more regularities and there is more context. This can then help inform inference in lower layers to resolve more difficult ambiguities about details of the scene. For this to occur, the network should perform causal inference in all layers simultaneously. The objective for the network should be to efficiently represent the independent underlying causes of incoming signal. While we will not be providing a single implementation of all of these ideas, we will review several important contributions and present novel architectures that put us closer to this goal.

3.2 Hierarchical sparse coding models

Broadly speaking, there have been many hierarchical unsupervised learning architectures in the literature. Here we will highlight work that is particularly relevant considering the aforementioned goals. Lee and Mumford (2003) propose a model that partially inspired what was described in the previous section. They simplify the model by forcing each layer to only receive input from the layers below and above, as in a Markov chain. Expectations are propagated down to alter priors for lower layers in a dynamic, context-sensitive manner. Feedback connections influence the inference process, causing a layer to converge on a solution that fits the expectations above and they use dynamic sampling algorithms to represent Bayesian inference. We believe a version that is true to their theory and uses sparse coding as a core computational framework is a promising direction for future research. Karklin and

Lewicki (2003) propose a two-layer probabilistic model that is an extension of the ICA model. They start with learning a complete orthogonal basis with ICA, and then learn "variance bases" that model higher-order structure in images. Their network produces a hierarchical sparse distributed code of natural scenes, but instead of performing inference, they directly compute first layer activations using a feed-forward architecture. They are able to learn to represent higher-order structure such as elongated edges with a nonlinear (in the second layer) encoding process because the joint coefficient distribution is replaced with a hierarchical prior. As proposed, their model is not extendable to include additional layers, which we identify as a direction for improvement. Additionally, we are interested in extending the LCA network to include their probabilistic second layer. The variances learned in the second layer can then be used to guide LCA inference. Honghao Shan and colleagues propose a hierarchical model that alternates PCA-like and ICA transforms (Shan and Cottrell 2013) and a second hierarchical model that recursively applies a pointwise nonlinear function of the ICA transform (Shan, Zhang, et al. 2007). These models learn visual features that resemble those found in the early stages of the primate vision system. In the first model they use PCA to reduce the input dimensionality and then overcomplete ICA (Le, Karpenko, et al. 2011) to learn higher order features from the PCA output. They show that when this process is repeated, their network continues to learn more interesting features. This model is a promising proof of concept for a framework that alternates dimensionality reduction and expansion to learn higher-order structure from natural images. In the second model they iteratively apply ICA with a pointwise nonlinear transform applied in between operations. The network learns features that qualitatively resemble higher-order features learned by other models. Cadieu and Olshausen (2008) propose a two-layer sparse coding network that explicitly disentangles form and motion in the second layer when trained on video input. The first layer is a sparse coding model that is trained with complex valued dictionary elements. They then factorize the first layer output into two sets of second layer inputs that encode the time-derivative of the amplitude and phase components to explicitly separate form and motion, respectively. The second layer units perform sparse coding on the logarithm of factorized quantities from the first layer. They demonstrate that form selective neurons develop invariance properties from the time-varying signal. We believe the use of time information from natural videos is a key component of learning a hierarchical representation of natural scenes that has high correspondence to what we see in biology. A convolutional LCA variant called the Deconvolutional Competitive Algorithm (DCA) is proposed in (Paiton, Lundquist, et al. 2015). This model deconstructs images into a three-layer hierarchical code. The network performs inference in all three layers simultaneously, and all three layers compete to reconstruct the input. By configuring the strides and patch sizes appropriately, the network was designed such that each layer contributes a different degree of spatial detail to the reconstruction. The lowest layer reconstructed high spatial frequency, grayscale information while the highest layer reconstructed low spatial frequency and color information. However, the generative process is linear and therefore could be represented with a single dictionary. Ideally, we would like the generative process to be able to account for the non-linear structure in images. Additionally, the network does not learn to represent the higher-order *structure* found in natural scenes,

such as elongated edges. The sparse manifold transform, proposed by (Y. Chen et al. 2018), is an extendable hierarchical sparse coding network that models the underlying manifold structure of time-varying natural scenes (i.e. videos). The model combines dimensionality expansion using sparse coding and reduction using manifold embedding. The transform is also invertible via a non-linear generative process, such that linear translations in the embedded space result in non-linear transformations in the image space. The learned dimensionality reduction step can be thought of as an alternative to pooling operations commonly found in deep learning architectures.

In the following sections, we will propose two novel methods for extending the LCA to a hierarchical framework. The first is framed as a weakly-supervised machine learning network, while the second is an unsupervised model for learning invariant representations of natural scenes.

3.3 Weakly-Supervised Feature Learning

Introduction

We evolved a visual sense to allow us to understand the external causes of incident light and our visual system is intimately connected with the statistics of light as it propagates through our natural world. These statistics have been analyzed extensively by scientists exploring images and videos of natural scenes (see chapter 1). Our visual processing pipeline is not designed to construct a veridical representation of the world (Gollisch et al. 2010). Instead, it is designed to allow us to identify objects of significance and act upon them. The sparse coding model represents an attempt to build on the knowledge gained from studying the statistics of natural scenes to better understand our visual system and produce a useful representation of input signals. In the field of visual unsupervised machine learning, a cornucopia of models have been proposed to learn the statistics of natural scenes without human labels, or "supervision" (see (Baldi 2012; Bengio et al. 2012; I. Goodfellow et al. 2016) for reviews). The unsupervised objective function of most of these models asks for a information-preserving latent representation of the inputs, often with an additional goal of compressing the signal through a bottleneck architecture. However, when a reconstruction loss is combined with a family of constraints that include minimum entropy, maximum compression, and minimal energy expenditure, autoencoder models can exhibit interesting properties that are also found in biological vision systems. In this chapter, we consider the LCA as an autoencoder. Like an autoencoder, the LCA receives inputs, transforms them into a latent code, and produces reconstructions. The LCA loss includes a term for information preservation, such that the input can be reconstructed from the encoding, as well as a constraint to minimize energy expenditure in the form of active neurons. We are interested in understanding how useful the LCA can be for the machine learning field. One way to assess this is to look at weakly-supervised (or semi-supervised) learning. Here the objective is the same as for supervised learning, where we want to associate images with

some predetermined category label. However, the catch is that many of the training images do not have ground truth labels assigned to them. A fully supervised model would not be able to use these, and would suffer from limited training examples. Here we show that the LCA can be used to improve weakly-supervised learning results. We also demonstrate how an alternate objective, such as labeling objects in the world, can be used in the LCA dynamics to modify inference and dictionary learning.

Weakly-supervised learning

Human generated labels are extremely expensive to produce and often biased. A learning paradigm that avoids this process is unsupervised learning, but it is not directly applicable to the machine learning task of assigning labels to data. Weakly-supervised learning is a sub-field that aims to combine the benefits of unsupervised and supervised learning. An ideal model should learn to categorize (e.g. cluster) data without ground-truth labels. As we will demonstrate in chapter 4, sparse coding produces a code that is both descriptive and faithful to the image content. Here, we wish utilize the descriptive nature of the LCA by modifying it to utilize limited label information about an input scene.

Features learned in unsupervised frameworks match those learned in supervised frameworks

The Discriminative Recurrent Sparse Auto-Encoder (DrSAE, Rolfe et al. 2013) behaves similarly to sparse coding and has been shown to be successful at semi-supervised learning. Unlike LCA, the encoder, decoder, and lateral connectivity weights of DrSAE are unconstrained and trained independently. However, when trained on the MNIST dataset (LeCun 1998), the DrSAE learns weights that closely match what we impose for sparse coding. That is, the decoder weights have high correspondence to the transpose of the encoder weights and the lateral connectivity weights have high correspondence with the Gramian of the encoder weights. Additionally, when semi-supervised training is performed, the DrSAE learns an emergent hierarchical architecture. Figure 3.1 shows that the features learned by the unsupervised LCA and DrSAE models have a high degree of structural similarity to those learned by a supervised model (LeCun et al. 1998). In general, supervised architectures require different types of regularization, such as dropout and weight decay, to learn structured weights.

Given the success of the DrSAE model, we constructed a similar framework with the LCA. The LCA dynamics are derived from a principled energy function, so we were able to easily extend the framework by adding semi-supervised loss terms. In sparse coding the sparsity enforcing term is typically applied uniformly, penalizing all nodes. Instead of the prior limiting the total activation, we want the prior to encourage some nodes to be active based on expectations propagated down from higher layers. In our network, these higher layers are focused on grouping inputs into similar categories.

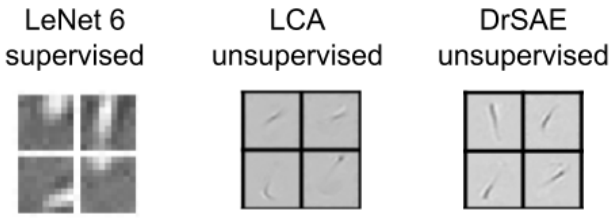


Figure 3.1: **Supervised and unsupervised learning produce similar weights.** The features learned from unsupervised training on MNIST using LCA (Rozell et al. 2008) and DrSAE (Rolfe et al. 2013) have similar structure to those learned using the supervised LeNet model (LeCun et al. 1998).

Our proposed network is capable of weakly-supervised learning, where only a small percentage of data examples have corresponding labels. We trained the models to categorize novel inputs using a heavily reduced set of labeled examples. Typical solutions to this problem utilize a combination of supervised and unsupervised learning objectives. The supervised objective aims to build an association between a given input and label, such that similar inputs receive the same label. The unsupervised learning objective aims to preserve a faithful representation of the input, such that the input data can be reconstructed directly from the network activations. In a typical scheme, the supervised objective is used when labels are available and the unsupervised objective is used when they are absent. In addition to these two classic objectives, we have added an additional unsupervised objective that encourages the network to confidently group the inputs into categories. This additional objective improves the networks ability to categorize inputs, which is typically absent from unsupervised learning. The loss function minimizes the output entropy per image and maximizes it per batch. The intuition is that minimizing entropy per image will force the network to confidently place the image into a category, since the number of output nodes is small (e.g. 10 for MNIST). Maximizing the entropy across batches is intended to prevent the network from placing all images into a single category. It assumes that there is an approximately even distribution of classes in a given batch. We implemented this loss by adding a second layer on top of the LCA network that produced proposed categorical outputs. The network is trained using cross-entropy when there are labels or the combined entropy terms described earlier when there are not. Taking the derivative of this new cost with respect to a neuron's activation will give us a new update rule for sparse inference with the LCA.

LCA with feedback

A traditional deep network layer produces its output by filtering input data through a linear weight matrix and a nonlinear thresholding (i.e. activation) function. The thresholded output is then passed to the next layer in the hierarchy. Dimension-reducing nonlinearities, such as max-pooling, are often included between layers to increase network invariance for label-preserving variations. They also prevent combinatorially increasing layer size with depth.

This process continues until, ultimately, a probability distribution over possible categories is produced as the final layers output. For static data classification, such as image labeling, most deployed state-of-the-art networks are feedforward in that information strictly flows in one direction through the network. Consequently, the layers themselves do not produce a dynamical response to the input. In our alternative approach, the first layer of our network performs a dynamical non-linear computation on the input, which modifies the second layer outputs at each iteration. The first layer is an LCA layer that incorporates lateral connectivity between neurons to enforce competition. This results in a descriptive, distributed sparse code of the input data (see chapter 2 for details). The code is produced in a recurrent fashion, where the network dynamics evolve through time to a converged representation of the input. Additionally, each LCA neuron receives input from the layer above that alters the dynamics in a context-dependent way. The resulting network representation is hierarchical and faithful to the input, such that the data can be directly reconstructed from the first layer neuron activation values. Within-layer competition and top-down feedback encourages the LCA to produce a descriptive code that is useful for image categorization. The semi-supervised nature of the model will allow us to leverage raw data without the need for expensive human labeling.

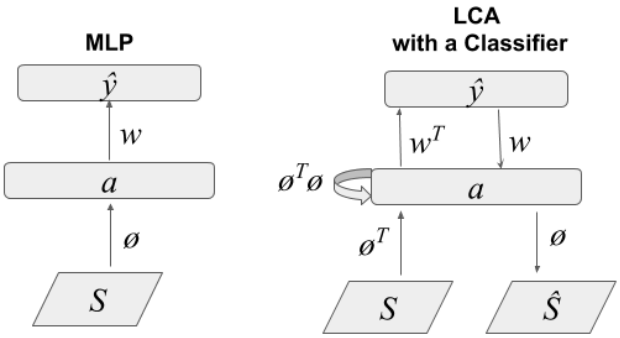


Figure 3.2: **Training a classifier on LCA outputs.** We propose a two-layer LCA architecture that uses a semi-supervised objective in the second layer and we compare it against a standard 2-layer MLP architecture. In our experiments, we use different values for Φ in the MLP model: random, pre-trained to match that used for LCA, and trained using the traditional supervised learning method.

For the LCA with feedback (LCAF) model, we will add an additional classification layer on top of the LCA model, as illustrated in figure 3.2. The network minimizes one of two different energy functions, depending on whether the input image has an associated label. In the case that there is a label, the energy function is the same as was used in equation (2.13), with the addition of a cross-entropy term:

$$E = \overbrace{\tfrac{1}{2}\|s - \hat{s}\|_2^2}^{\text{Preserve Information}} + \overbrace{\lambda \sum_{i=1}^{M} C(a_i)}^{\text{Limit Activations}} - \overbrace{\alpha \sum_{j=1}^{K} y_j \log(\hat{y_j})}^{\text{Cross-Entropy Cost}}, \tag{3.1}$$

where K is the number of label categories, α is a tunable trade-off parameter, y_j is a ground-truth one-hot label, and $\hat{y_j} = \frac{e^{-Wa_j}}{\sum_n e^{-Wa_n}}$ is the softmax output of the classification layer. When a label is not present, we swap out the cross-entropy cost with an entropy cost. We want our model to have high confidence (low entropy) per image and high entropy across batch (because we are assuming the categories are evenly distributed). We do this by defining two entropy terms. The first computes the entropy per image by summing across the neuron indices:

$$\begin{aligned} Q_{b,k} &= \frac{e^{\hat{y}_{b,k}}}{\sum_{j=1}^{K} e^{\hat{y}_{b,j}}} \\ H_b^{\text{neuron}} &= -\sum_{k}^{K} Q_{b,k} \log Q_{b,k}. \end{aligned} \tag{3.2}$$

The second term computes the batch entropy by summing across the batch dimension:

$$\begin{aligned} P_{b,k} &= \frac{e^{\hat{y}_{b,k}}}{\sum_{j=1}^{B} e^{\hat{y}_{j,k}}} \\ H_k^{\text{batch}} &= -\sum_{b=1}^{B} P_{b,k} \log P_{b,k}, \end{aligned} \tag{3.3}$$

where B is the batch size and K is the number of softmax outputs. Now we can combine these terms for our unsupervised entropy loss:

$$E = \overbrace{\tfrac{1}{2}\|s - \hat{s}\|_2^2}^{\text{Preserve Information}} + \overbrace{\lambda \sum_{i=1}^{M} C(a_i)}^{\text{Limit Activations}} + \overbrace{\alpha_1 \sum_{b=1}^{B} H_b^{\text{neuron}} - \alpha_2 \sum_{k=1}^{K} H_k^{\text{batch}}}^{\text{Entropy Cost}}, \tag{3.4}$$

where α_1 and α_2 are tunable loss trade-off parameters. Our LCA inference equation follows the same derivation from equation (2.19), with an added term computed from the derivative of the entropy or cross-entropy costs with respect to the activity vector, a.

	MLP	MLP with random Φ	MLP with LCA Φ	LCA with classifier
Test Error	411	2979	1723	331
Test Accuracy	95.89%	70.21%	82.77%	96.69%

Table 3.1: **LCA helps with MNIST classification.** This table shows that a single layer classifier trained on LCA encodings of the MNIST digit dataset outperforms a two layer classifier trained directly on MNIST pixels. The first column is the results for a two-layer classically trained MLP. The second is the same, except that the first layer weights were frozen with random initialization. The third had the first layer weights frozen to those that were trained with LCA, but did not utilize sparse inference. The final column is the results for a single-layer classifier trained on the LCA activations.

Number of labeled training examples	LCA	LCAF sup only	LCAF sup & unsup
50,000	96%	96%	96%
100	68%	67%	65%
20	33%	33%	38%

Table 3.2: **Feedback helps with MNIST classification.** We compare the LCAF model against two variants: One with strictly supervised feedback (middle column) and another with supervised and unsupervised feedback (right column). Although the feedback does not appear to help when there are a large number of labeled examples, it does show a positive effect when the number of labeled examples is restricted.

Experiments on MNIST dataset

Our fist experiment is to verify that the classifier is able to train using sparse codes as input. Table 3.1 gives the MNIST test accuracy using a fully supervised label set. The LCA model was pre-trained on MNIST without labels and then a single layer perceptron (SLP) classifier was trained on the activity vector, a. Our results indicate that the LCA+SLP network out performs the MLP network.

Next we modified the MNIST dataset such that a varying percentage of the labels are removed to test our networks ability to generalize and label unseen digit images. In table 3.2, we show that adding supervised cross-entropy feedback to sparse inference had little effect in the limited label regime. However, combining the supervised cross-entropy feedback with the unsupervised entropy feedback resulted in an improved score with very few labeled examples. These scores were not cross-validated, and therefore we cannot assign confidence intervals to the accuracy reported. The error rates found from previous studies were 0.1%, 1%, and 5% for 50000, 100, and 20 labeled examples respectively (Rasmus et al. 2015). We expect that our error rates would be close to those previously reported, as they largely stem from variation in the training and validation sampling.

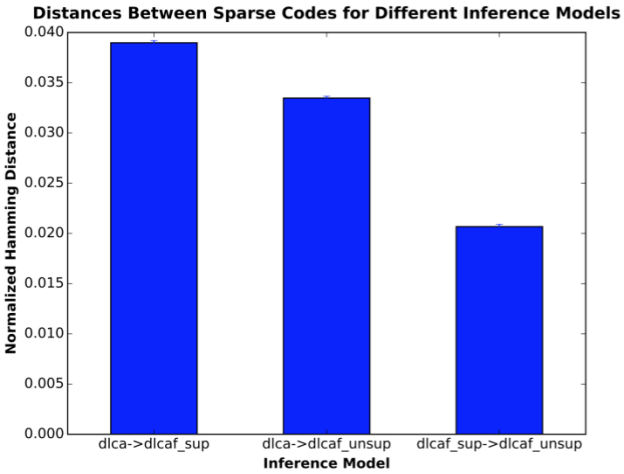
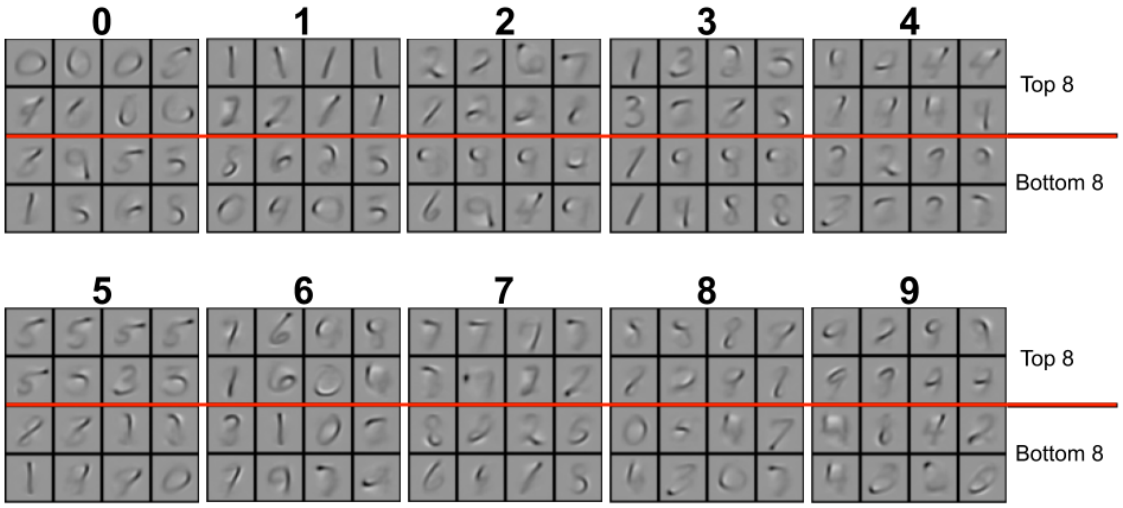


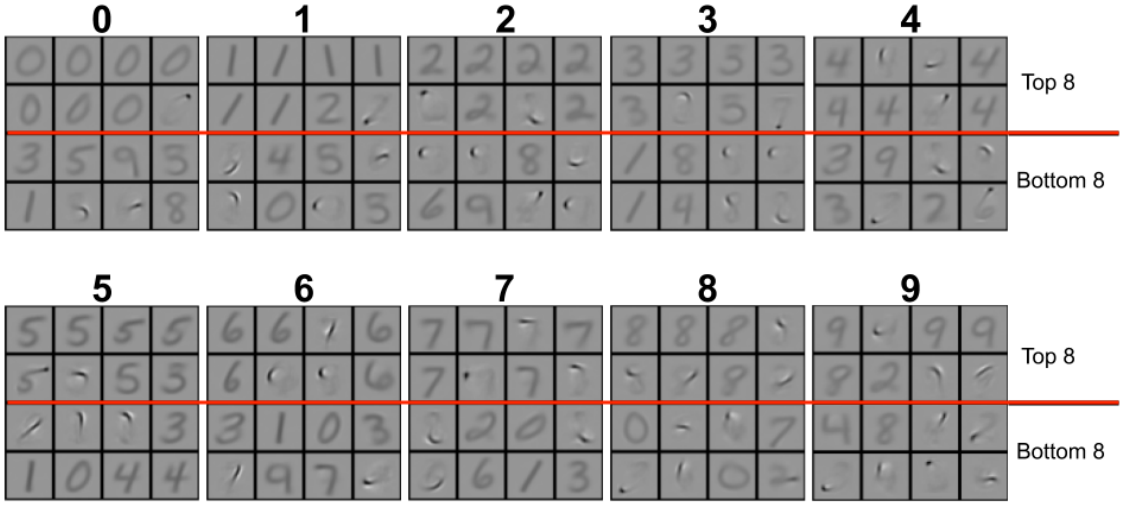
Figure 3.3: **Unsupervised feedback during LCA inference produces similar codes to supervised feedback.** We measure the number of neurons that crossed their activation threshold (from active to inactive or vice versa) in terms of a Hamming distance. The distances between the codes produced without feedback and with either form of feedback (left two bars) are larger than the distance between the codes produced with supervised and unsupervised feedback (right bar). This tells us that the unsupervised entropy feedback produces a meaningful signal that is similar to that produced by supervised cross-entropy feedback. The bars height indicates mean hamming distance for 100 images and error bars indicate standard error of the mean.

To test whether the unsupervised feedback term produces a meaningful signal for inference, we measured the distances among sparse codes produced without feedback, with supervised feedback, and with unsupervised feedback. Figure 3.3 shows that the Hamming distance between codes produced with either form of feedback is smaller than the distance from a code produced with feedback to one produced without feedback. Here we measure Hamming distance as the number of flipped bits between binarized vectors where a 1 indicates that a neuron's membrane potential, u, crossed threshold (i.e. went from active to inactive or vice versa) and a 0 indicates otherwise. This tells us that the feedback itself changes the code produced, and also that the unsupervised feedback produces a code that is similar to the supervised feedback code. This suggests that the unsupervised feedback is a good proxy for the supervised signal.

We also tested the weights learned with and without feedback during inference. In this experiment, the dictionary update rule is the same as was described in equation (2.25), but the sparse inference process was modified. Figure 3.4 shows that the feedback process influences the first layer weights, which learned to produce more prototypical digits. Additionally, the first layer neurons that are most strongly connected to a specific second layer classification neuron have a higher degree of correspondence to the classification category when feedback was used during inference.



(a) Without feedback



(b) With supervised feedback

Figure 3.4: **Feedback influences weight gradients for the LCAF network.** The subfigures show the basis functions, Φ, for the top and bottom strongest connected first layer neurons to each classification output neuron. (a) Inference was performed with supervised feedback. (b) Inference was performed without feedback. Each 4x4 grid corresponds to a particular classification label. Images above the red line are the basis functions for the 8 neurons that are most strongly connected to the given classification neuron and below the red line are the bottom 8. The basis functions themselves change with feedback, and the structure of the top connected basis functions is better matched to the digit label when feedback is used.

Conclusion

We proposed a two-layer LCA network extension that uses semi-supervised loss signals to direct sparse inference. We showed that the feedback signal influences the weights learned and improves classifications accuracy when there is a restricted set of labels. In the following section, we propose an alternative two-layer LCA network that is completely unsupervised. Combining the ideas from these two sections is a current area of research.

3.4 Subspace LCA

Introduction

We describe an extension to the LCA model that produces invariant representations of its inputs. The model is similar to Independent Subspace Analysis (Hyvärinen et al. 2000), but differs in several key ways. First, the LCA model allows for the first layer representation to be overcomplete, which improves efficiency (Lewicki and Sejnowski 2000). Second, the first layer encoding process is non-linear, which further improves efficiency as well as selectivity (see section 4.3).

Model description

The subspace LCA follows a similar derivation to the LCA (section 2.4). Like the LCA, it learns a set of weights to efficiently describe natural signals, although in this variant we constrain the weights to be grouped. We set the number of neurons in a group as a hyper-parameter. We start with the same generative framework as in sparse coding (equation 2.1). We will constrain our activations (and therefore weights) to non-overlapping groups that have an amplitude, σ:

$$\sigma_i = ||a_i||_2 = \sqrt{\sum_{j \in L} a_{ij}^2}, \tag{3.5}$$

where i indexes the group and $j \in L$ indexes the neuron within the group. The group amplitude is equivalent for many combinations of $a_{j \in L}$. Each of these equal combinations can be thought of as a direction of a vector in the activity space. We define this as a unit-length direction vector, z, that has the same number of elements as our activity vector:

$$z_{ij} = \frac{a_{ij}}{\sigma_i}. \tag{3.6}$$

We can now define a new energy function in these terms. We also add a regularization term that pressures the within-group weights to be orthogonal, which prevents the pathological

solution of within-group neurons learning to have identical weights. This regularization term will also reduce competition between neurons within groups, which is scaled by the inner product between neighboring neurons' weight vectors. The energy is

$$E = \frac{1}{2}\sum_p \left[s_p - \sum_{ij} \sigma_i z_{ij} \Phi_{ijp} \right]^2 + \lambda \sum_i \sigma_i + \alpha \sum_{ij} \left| \sum_p \Phi_{ijp}\Phi_{ijp} - \mathbf{I}_L \right|, \tag{3.7}$$

where α is a trade-off multiplier, $\mathbf{I}_L$ is the $L \times L$ identity matrix, and L is the number of neurons in a group. Following the LCA derivation in section 2.4, we will next take the derivative of the energy with respect to neuron k's activity:

$$-\frac{\partial E}{\partial a_{ik}} = \sum_p s_p \phi_{ikp} - \sum_{lm} G_{iklm} a_{lm} - \lambda \frac{\partial ||a_i||^2}{\partial a_{ik}}. \tag{3.8}$$

We can rewrite the last term in the above equation as our direction vector:

$$\begin{aligned} \frac{\partial ||a_i||^2}{\partial a_{ik}} &= \frac{1}{2}\left(\sigma_i^2\right)^{-\frac{1}{2}} 2a_{ik} \\ &= \frac{a_{ik}}{\sigma_i} \\ &= z_{ik}. \end{aligned} \tag{3.9}$$

In parody with the LCA derivation, we group the self inhibition terms:

$$\begin{aligned} f_\lambda(a_{ik}) &= a_{ik} + \lambda z_{ik} \\ &= (\sigma_i + \lambda)\, z_{ik} \\ &= a_{ik}(1 + \frac{\lambda}{\sigma_i}). \end{aligned} \tag{3.10}$$

We again assign our membrane potential, u as a function of a:

$$u_{ik} = f_\lambda(a_{ik}) = (\sigma_i + \lambda) z_{ik}, \tag{3.11}$$

which also gives us an alternative definition for the neuron angle, z:

$$z_{ik} = \frac{u_{ik}}{\sigma_i + \lambda} = \frac{u_{ik}}{||a_i||_2 + \lambda}. \tag{3.12}$$

The resulting membrane update rule is nearly identical to equation 2.22, except that the lateral competition term includes group assignments:

$$\tau \dot{u_{ik}} - u_{ik} = \sum_p s_p \phi_{ikp} \sum_{lm \neq ik} G_{iklm} a_{lm}. \tag{3.13}$$

Finally, we define the output amplitude in terms of the group threshold:

$$a_{ik} = T_\lambda(\sigma_i) = \begin{cases} 0, & \sigma_i \leq \lambda \\ (\sigma_i - \lambda) z_{ik}, & \sigma_i > \lambda. \end{cases} \tag{3.14}$$

This tells us that all within-group neurons become active when the group amplitude surpasses the threshold, λ. Figure 3.5 shows a diagram of the group model.

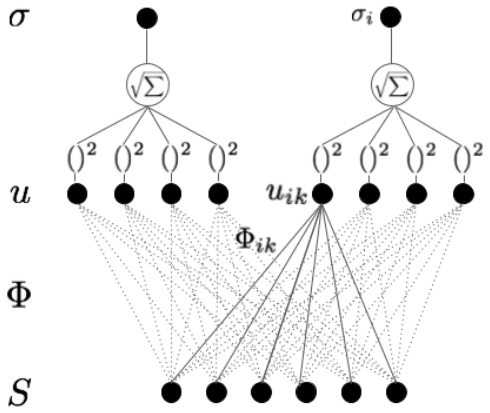


Figure 3.5: **Subspace LCA**. Neurons are grouped such that they learn subspaces of co-active units. Once a group amplitude passes the threshold, all neurons in the group produce an output activity that is modulated by the direction vector, z.

Features learned

When trained on natural images, the weights learn to tile orientations, spatial frequencies, and positions just like regular LCA. They also learn to have within-group similarities, such as equal orientation or position.

Interestingly, we show in figure 3.7 that the group structure naturally separates digit classes when the model is trained without supervised labels on the MNIST dataset. We believe this shows promise to use this model in a semi-supervised framework similar to that described in section 3.3.

3.5 Conclusion

Hierarchical structure is found throughout cortex, but the underlying computation being performed by this structure is largely unknown. We propose that Bayesian efficient coding models can be extended into a hierarchical framework and in section 3.2 we provided several supporting examples from the literature. In section 3.3, we proposed a hierarchical extension whereby a classifier is trained on the outputs of the LCA. Our semi-supervised network learns

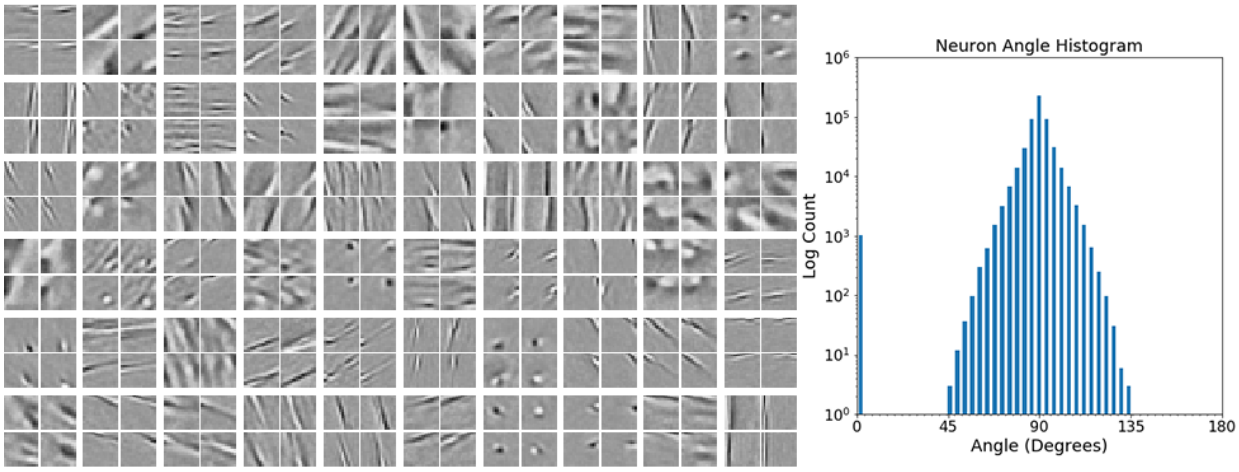


Figure 3.6: **Natural image features learned with subspace LCA**. The left plot shows that subspace LCA learns features that have similar properties within group, but are different across groups. The right plot shows that the basis function angle histogram is very similar to that learned with the LCA.

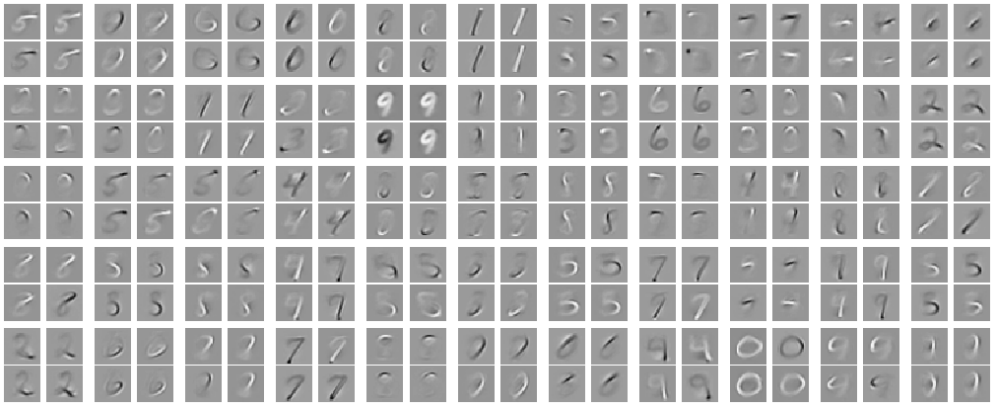


Figure 3.7: **MNIST features learned with subspace LCA**. The features learned with this model natural separate into digit categories, indicating that it might be a simple operation to assign images labels from the group amplitude representations.

a hierarchical representation of visual data with or without corresponding labels. Instead of a narrowly specified task, the primary objective of our model is to construct a general, hierarchical and efficient code of the data that can be applied to a myriad of different tasks. We demonstrated a promising research direction of using supervised and unsupervised entropy signals to direct LCA inference. Future work includes investigating further how feedback influences inference and learning, how the depth of the classifier influences inference and learning, as well as making comparisons to alternative methods.

In section 3.4, we proposed a novel extension of the LCA that learns statistical dependencies

of sparse codes by grouping co-active elements into subspaces. Future work includes drawing comparisons to the Independent Subspace Analysis (ISA - Hyvärinen et al. 2000) and extending the model to a topographic version. Also, we believe this formulation can be extended to include ideas from dynamic routing (Olshausen, C. H. Anderson, and Van Essen 1993), where the z variables "steer" the output representation. Finally, we wish to study the invariance gained by this type of grouping and make comparisons to alternative complex cell models.

Chapter 4

Selectivity, Efficiency, and Robustness of the LCA

4.1 Introduction

Work from David Field's lab (Golden et al. 2016; Vilankar et al. 2017) proposes a single-neuron analysis technique that allows us to increase the complexity of our neuron models without significantly decreasing our understanding by characterizing the neurons response geometry in the form of iso-response contours. They show that contours can be straight or bent, and if bent they can bend towards the origin (endo-origin) or away from it (exo-origin). We build on this work and show that the LCA exhibits curved iso-response contours when trained on natural images as well as the MNIST dataset. This curvature is a result of population nonlinear computations provided by the LCA's lateral inhibitory connectivity and indicates improved selectivity against perturbations that are orthogonal to the neuron's preferred stimulus. First we directly measure the network's selectivity to full-field oriented grating stimulus and show improved performance over linear and pointwise-nonlinear networks. Next we provide evidence supporting the hypothesis that improved selectivity results in a more efficient code of natural images. Finally, we propose that this selectivity forces adversarial perturbation vectors to have angles that are more in line with data dimensions, producing more semantically meaningful perturbations. We develop an analytic argument to support this claim for single-neuron adversarial attacks. Furthermore, we demonstrate by experiment that our method successfully improves robustness to adversarial attacks for a classifier trained on our model's outputs. We contribute a novel perspective for understanding adversarial attacks and population nonlinear networks.

4.2 Measuring iso-response contours

Inferring causes from natural signals is a challenging problem faced by biological vision systems. In biological neural networks we see a preponderance of evidence that individual

neurons are highly non-linear, recurrent, and laterally connected. The LCA is a generative model with lateral connectivity and recurrent inference that exhibits a variety of linear (Olshausen and Field 1996) and non-linear (Zhu et al. 2013) receptive field effects that are well matched to biological neurons. Many of these effects can be explained using the model neurons' iso-response contours - an experimental method for determining stimulus combinations that result in equal activation from recorded neurons (Golden et al. 2016). In the brain, the atomic unit of computation is generally considered to be the neuron, but most deep learning research has eschewed the individual neuron, possibly obfuscating the connections between biological and artificial neural networks. We compare the iso-response contours of feed-forward, pointwise nonlinear discriminative and generative network architectures against a more biologically-consistent population nonlinear generative network architecture. Pointwise nonlinearities are the more traditional form of nonlinearities and are seen in many deep neural network architectures and computational neuroscience models. They are defined as nonlinearities that are a function of only a single neuron in a layer and include rectification, sigmoid, and hyperbolic tangent. Population nonlinearities represent the alternative class, where the nonlinearity output is a function of multiple neurons in a set. These include divisive normalization (Carandini et al. 2012; Ballé et al. 2016) and the network nonlinearity present in sparse coding (Rozell et al. 2008; Olshausen and Field 1997).

Consider that a P-pixel image can be thought of as a single point in $\mathbb{R}^P$. To measure a target neuron's iso-response contours, we first choose a two-dimensional cross section (or plane) in the P-dimensional space that is defined by two orthogonal vectors. For most of the work here, we will use the target neuron's feed-forward weights as one of the two vectors. To determine the second axis of our image plane, we start by finding another neuron's weight vector that has a nonzero inner-product with our target neuron's weight vector (and thus they are not orthogonal). Then, to get an orthonormal basis, we use the Gram-Schmidt process to find a vector orthogonal to our target neuron, but coplanar with our second neuron. For competitive networks like the LCA, this method increases the likelihood of competition between neurons and thus increases the curvature. Each point in the 2-D plane can be injected back into the image space and will have a high degree of correspondence to relevant data features. We use each point as input to the neuron model and then bin the points according to the neuron's output amplitude. The bin boundaries reveal the iso-response contours of the target neuron. The iso-response contours of linear neurons are straight: any input perturbation that is orthogonal to the weight vector will result in equal activation. For pointwise nonlinearities, this remains true: because the nonlinearity is performed after a linear projection, the output must also produce straight iso-response contours. By contrast, for a population nonlinearity the gradient of the activation function with respect to a small perturbation in the input is a function of all other neurons in the layer. Consider: for a perturbation that is orthogonal to a target neuron's weight vector, it is generically the case that some other neuron will have a non-orthogonal weight vector, which can result in a net change in all neuron outputs. Therefore, iso-response contours for population nonlinear neurons can be bent. In figure 4.1 we demonstrate contours for linear and nonlinear neurons.

It is important to determine an individual neuron's response contours for a large number of

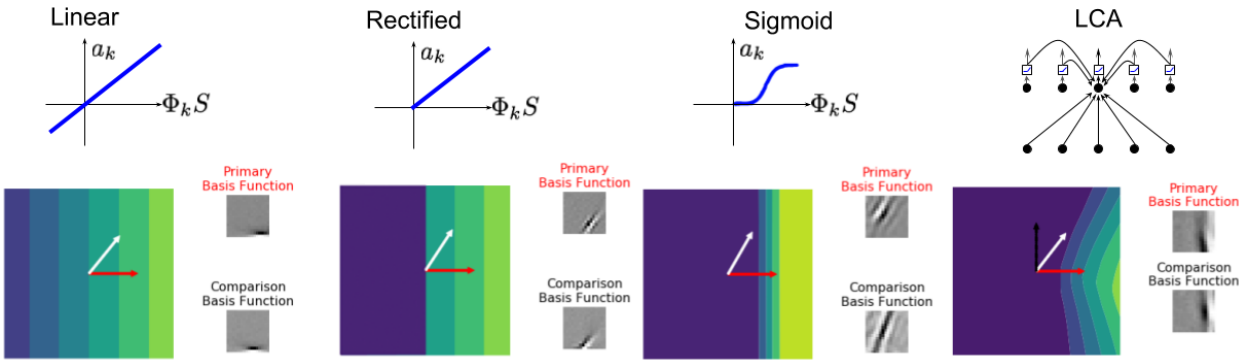


Figure 4.1: **Empirically measured iso-response contours.** A fine sampling of points in a 2D plane are injected into the high dimensional image space and used as inputs to a target model neuron, a_k. The color value indicates a binned normalized response magnitude of the k^{th} neuron, where purple indicates zero activity and yellow indicates the maximum. The red arrow is the weight vector for the neuron, Φ_k. The white arrow is an alternate neuron's weight vector.

planes to gain insight into its high-dimensional response geometry. We propose two different methods for choosing image planes. For both methods, we define one axis as the target neuron's weight vector. The first method to find the orthogonal axis is to iteratively apply the process we described above for each other neuron in the layer. For the second method, we compute a set of random vectors that are orthogonal to the target neurons weight vector. We then estimate the curvature in each plane found via these two methods, which allows us to better understand the neurons high-dimensional iso-response geometry. Figure 4.2 demonstrates that LCA neurons have negative curvature in nearly all planes tested. Thus, most orthogonal perturbations from a neuron's weight vector will result in a decrease in its activation. This is an important quality that we desire from our model neurons. In visual neuroscience, we often use the neuron's linear receptive field (in our model that is represented by its weight vector) to represent the stimulus that the neuron is selective for. With a pointwise nonlinear neuron model, it is possible to deviate away from its weight vector in many different directions without changing the neuron's response. LCA neurons, on the other hand, have a higher degree of selectivity to perturbations away from their receptive field (Vilankar et al. 2017). In the following section, we provide supporting evidence by comparing the orientation selectivity of the LCA against linear and pointwise nonlinear neuron models. We also show that the LCA is more efficient than linear encoding models with oriented receptive fields, which we argue is a result of the improved selectivity.

4.3 Orientation selectivity and efficient coding

A longstanding hypothesis in sensory neuroscience proposes that a primary function of early sensory processing is to form an efficient, redundancy-reduced code of the input that

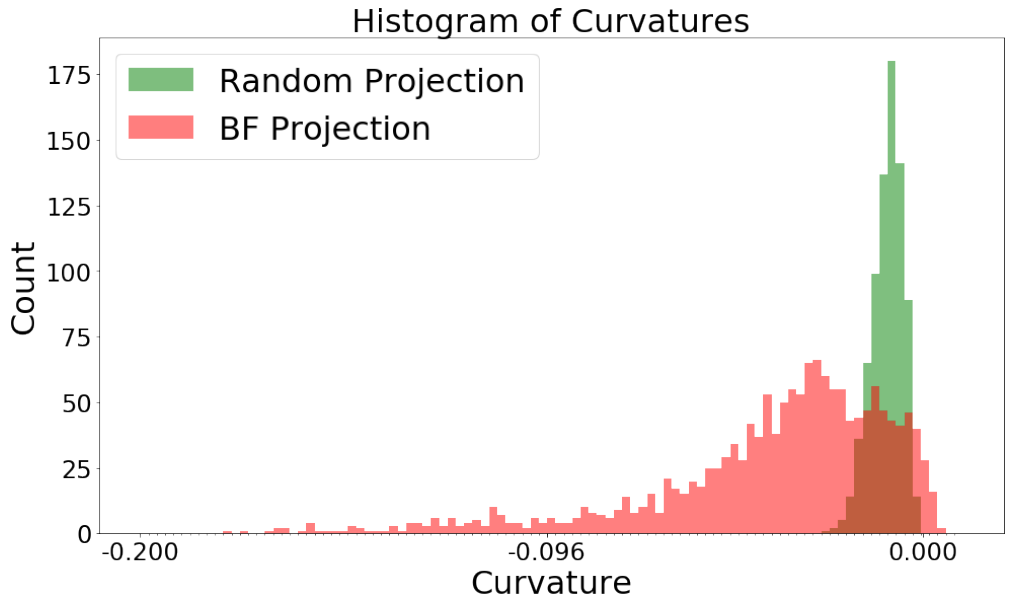


Figure 4.2: **LCA neurons have high-dimensional exo-origin curvature.** The histogram shows second order coefficients for polynomial fits to the lines in the middle plot. Negative coefficients indicate exo-origin curvature, which tells us that LCA neurons exhibit exo-origin curvature in almost all orthogonal directions tested. The target neuron was randomly selected from a 2x overcomplete LCA model trained on the MNIST dataset with z-score (zero mean, unit standard deviation) preprocessing. We replicated these results for many random neurons. See text for details about random versus targeted directions.

maximizes the brain's limited computational and biological resources while making explicit the statistical structure of the input (Barlow 2009). This hypothesis predicts that the response properties of sensory neurons should be adapted to the statistical structure of their input. In support of this hypothesis, a number of the response properties of visual neurons have been reproduced by optimizing redundancy-reducing linear transformations on natural images. For example, a symmetric decorrelation transformation of natural images yields center-surround receptive fields (Atick and Redlich 1990), and Principal Component Analysis (PCA) applied to color images yields the luminance, red-green, and blue-yellow channel observed in the opponent color coding of retinal ganglion cells (Ruderman et al. 1998; Buchsbaum et al. 1983). When higher-order correlations are additionally reduced, localized, oriented, and band-pass filters that resemble the orientation-selective receptive fields in V1 simple cells emerge (Bell et al. 1997; Olshausen and Field 1997). It has thus been proposed that the oriented filters in V1 function to remove higher-order correlations.

Orientation selectivity is a striking feature of the response properties of simple cells in V1. Since the discovery of orientation selectivity in Hubel and Wiesel's Nobel Prize-winning work, the mechanism for the computation has remained unclear. A common point of confusion in the field has been the assumption that a neuron with a locally oriented receptive field will

exhibit orientation selectivity. Here, we will argue that narrow orientation selectivity requires a non-linear encoding process in addition to an oriented receptive field. For experiments in this section all models were trained on natural image patches from the van Hateren dataset, which were transformed to log intensity, whitened, and normalized to have zero mean and unit variance (Hateren and Schaaf 1998).

LCA neurons have improved orientation selectivity

Bent iso-response contours have important implications regarding the coding capabilities of neurons. Work from David Field's lab (Golden et al. 2016; Vilankar et al. 2017) suggests that the degree of exo-origin curvature for neurons indicates how selective they are. Here, we demonstrate this by showing that LCA neurons are more selective to oriented gratings than ICA neurons or neurons with pointwise sigmoid nonlinearities. As we illustrate in figure 4.3, linear or pointwise nonlinear neurons will respond equally to perturbations that are orthogonal to their weight vector. However, if the target neuron has exo-origin bent contours, then any orthogonal perturbation from the neuron's weight vector will result in the activation decreasing.

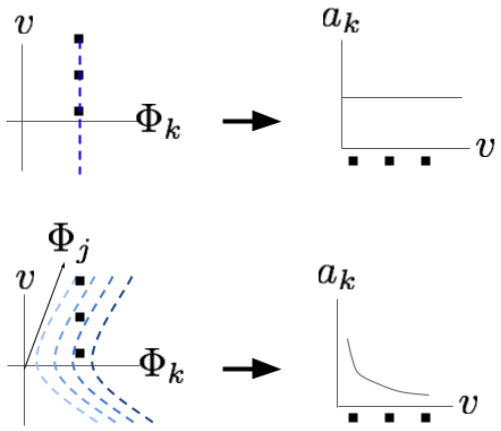


Figure 4.3: **Bent contours indicate increased selectivity.** This diagram shows qualitatively what expected responses would be for perturbations that are orthogonal to a neuron's weight vector. Each square represents an example input and the plots on the left indicate the inputs' positions in an image plane defined by the neuron's weight vector and an orthogonal vector. The plots on the right show what a model neuron's output would be for each type of nonlinearity. The bent contours indicate that a neuron's response will go down as one moves in an orthogonal direction from the weight vector in the image plane. This has important implications for neural coding - the neuron with bent contours is going to produce an activation value that is more indicative of the input's proximity to its weight vector.

We acknowledge that any degree of preference towards one orientation could be considered "orientation selectivity". In the works of (Golden et al. 2016; Vilankar et al. 2017), they

alleviate this confusion by defining *hyperselectivity* as a measure of the falloff in sensitivity as one moves away from a neuron's optimal stimulus, which is consistent with what we explained previously. Instead of adhering to their mathematical definition of hyperselectivity, going forward we will refer to it qualitatively as narrow-band sensitivity to a type of stimulus. Here we will only measure selectivity to changing orientation; we reserve measuring selectivity to other stimulus variations for future work.

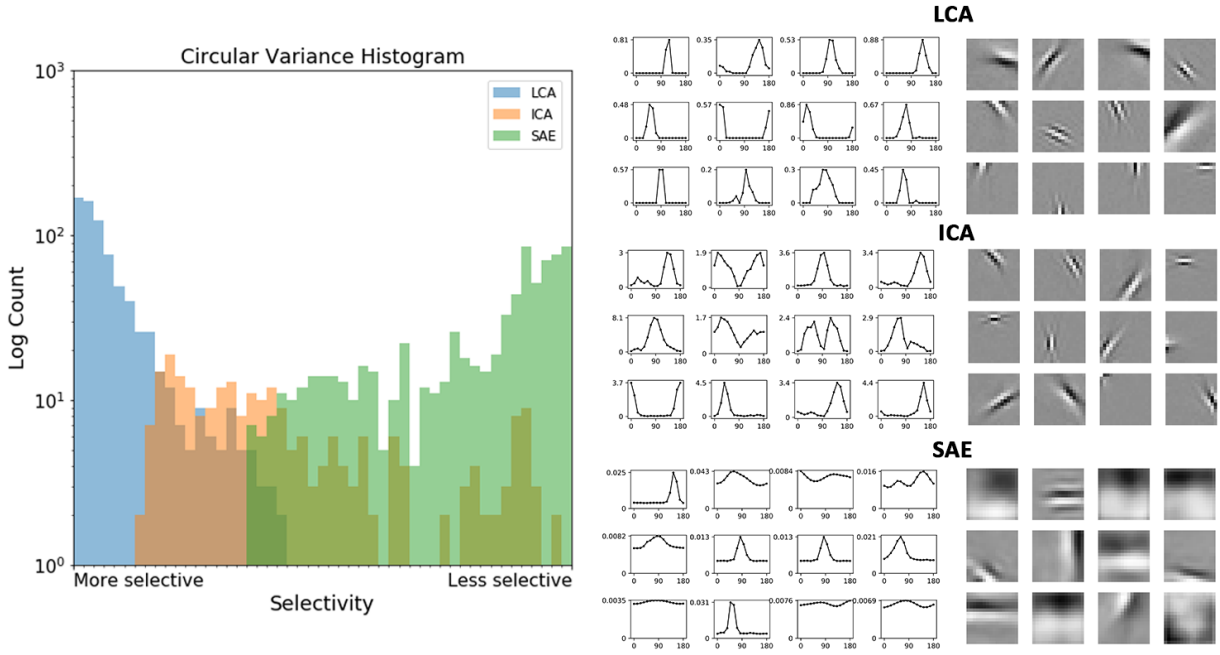


Figure 4.4: **The LCA is more selective to oriented gratings than ICA or neurons with a sigmoid nonlinearity.** On the left is a histogram of the circular variance (Ringach et al. 2002), a measure of orientation selectivity, for all of the neurons in each model. In the middle we show example orientation selectivity diagrams for neurons with different nonlinearities. On the right we show the corresponding weights for each of the orientation plots. The LCA and SAE networks had 768 latent units and the ICA network had 256. We also found that the shift in circular variance histograms as we increased overcompleteness (we tested 2x, 3x, and 4x overcompleteness levels) of the LCA was negligible compared to the difference between nonlinearities (data not shown).

In figure 4.4, we compare the orientation selectivity of the LCA against ICA and a sparse autoencoder (SAE, A. Ng et al. 2011). We show that although all of the models can be trained to learn oriented filters, none of them are as selective to oriented grating stimuli as the LCA. To produce the grating stimulus, we first computed the optimal spatial frequency for each neuron's weight vector using a Fourier transform. Next, we constructed a set of full-field grating stimuli with the target spatial frequency, 16 different orientations, and 8 different phases. Finally, we compute the neuron's circular variance to measure the orientation

selectivity, which is a more generic measure than the traditional full-width at half-max in that it does not assume a Gaussian profile (Ringach et al. 2002).

Rate-distortion analyses

Independent Component Analysis (ICA) is one of the most widely used image coding algorithms and has been proposed as a model for simple cells in V1 (Bell et al. 1997; Hyvarinen 1999). The ICA algorithm explicitly optimizes for higher-order redundancy reduction, aiming to to reconstruct the input image as a linear superposition of a set of basis functions while minimizing the mutual information between the coefficients of data vectors in that basis. Eichhorn et al. (2009) compare the coding efficiency of ICA and PCA to obtain the surprising result that ICA performs no better than PCA on a rate-distortion trade-off metric. ICA is trained with the objective of minimizing the joint entropy of the activations and learns oriented filters that suggest it has succeeded in modeling higher-order pixel correlations, while PCA is a second-order method that does not learn oriented filters. Eichhorn et al. (2009) argue that if ICA had succeeded at capturing higher-order statistics, it should show an advantage in the rate-distortion trade-off. We present an alternate explanation of these findings by distinguishing *orientation selectivity* from *oriented filters*. As we have discussed earlier in this chapter, neurons achieve narrow orientation selectivity via a nonlinear process. We argue that although the ICA optimization algorithm is able to learn oriented filters, ICA's linear encoding process limits its capacity to perform orientation selectivity, which in turn limits its capacity to produce an efficient code. Here, we replicate the rate-distortion analyses from (Eichhorn et al. 2009) to show that the LCA's nonlinear encoding process produces lower entropy codes than linear encoders while being more perceptually robust to increasingly coarse quantization.

We trained LCA (Rozell et al. 2008), ICA (Bell et al. 1997), and PCA (see Jolliffe et al. 2016 for a recent review) on 1 million 16 x 16 pixel grayscale natural image patches. Then, using the learned filter matrices, we computed model activations for a test set of 100,000 patches and uniformly quantized these activations with varying degrees of granularity. For each level of granularity, we computed a reconstruction of the test input using the quantized activations and computed the mean squared error. Figure 4.5 plots the rate (mean marginal discrete entropy of the activations) against the distortion (mean squared error). Results for the LCA trained with different values of λ are also shown, where a larger λ indicates higher sparsity. We replicate the findings from (Eichhorn et al. 2009) that (orthogonal) PCA performs slightly better than ICA in the rate-distortion trade-off. The LCA shows an advantage over both ICA and PCA. Additionally, we find that representations that are more sparse are capable of achieving increasingly lower rates.

Discussion

Our results suggest the importance of nonlinear encoding for producing efficient codes of natural images and we demonstrate that narrowly orientation selective neurons are capable of

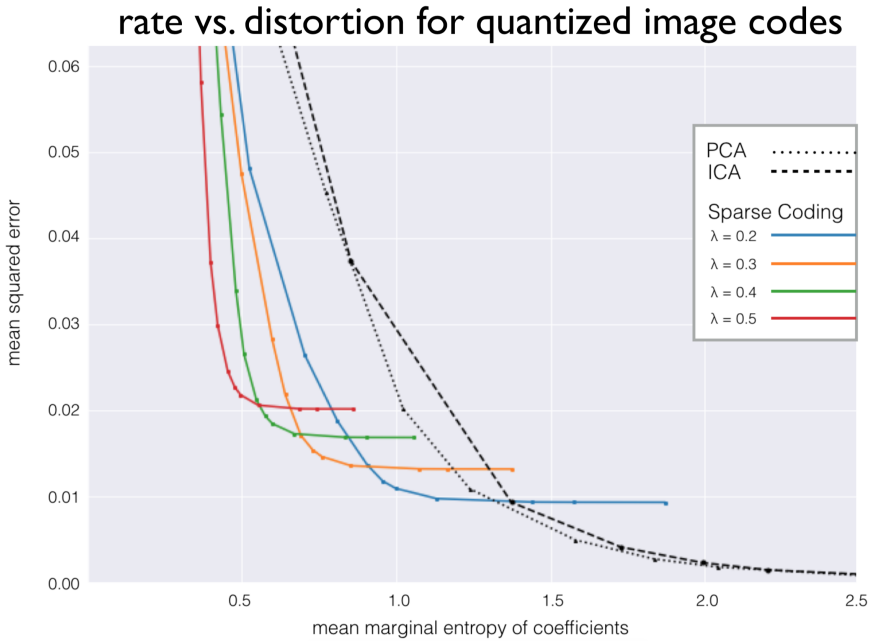


Figure 4.5: **LCA provides better compression than ICA or PCA.** The rate-distortion plot compares the LCA trained with different sparsity levels against linear transforms. We replicate the finding from (Eichhorn et al. 2009) that PCA performs slightly better than ICA in the rate-distortion trade-off, however the LCA shows an advantage over both PCA and ICA. Higher values of lambda indicates higher sparsity (i.e. fewer active elements). This figure was reproduced from unpublished work with permission from (Sanborn et al. 2018).

reducing higher-order redundancy over second order linear transforms. We demonstrate that the linear ICA neuron and the sigmoid nonlinear neuron used in the sparse autoencoder have broad orientation selectivity when compared to the LCA. We also show that although the ICA algorithm is able to learn oriented filters, ICA's linear encoding process limits its capacity to perform genuine orientation selectivity, which in turn limits its capacity to produce an efficient code.

4.4 Iso-response contours predict adversarial attack directions

Biological neurons are highly interconnected, both across and within layers, which gives rise to strong population nonlinear effects. However, a large majority of research in neuron modeling uses pointwise nonlinearities due to the ease in interpretation and implementation. Additionally, much of the research in adversarial analysis has focused on discriminative models (A. Y. Ng et al. 2002). Generative models, on the other hand, should not be as susceptible to adversarial attacks as discriminative models. This is because they are

trying to fit their internal model to the data as a "sanity check", which ideally should not succeed for adversarial perturbations. Indeed, we identify classification of natural scenes as a perceptual problem, and we have argued in previous chapters that perception is a problem of inference. In the absence of inference, the classification problem is insoluble, and adversarial examples are one aspect of that. One method by which generative models can perform such a sanity check is via recurrent maximum a-posteriori (MAP) inference. The LCA network uses population nonlinear interactions via lateral connectivity among neurons to facilitate this process, resulting in an "explaining away" effect commonly referenced in Bayesian inference literature (Olshausen 2013b). This inference process also produces neurons that have exo-origin curved iso-response contours. These two concepts are related, and the degree of exo-origin curvature can indicate how strong the explaining away effect is (Vilankar et al. 2017). Here we bring these concepts together by comparing adversarial attacks against generative networks with curved iso-response contours and attacks against discriminative networks without.

Adversarial Examples

Adversarial examples are a demonstration of an artificial neural network's (ANNs) inability to cope with distributional shift, which is any transformation that adjusts the distribution of inputs from that of the training data without changing semantic content (Ford et al. 2019). They represent a worst-case demonstration of this weakness, which is a general problem that all classification ANNs are susceptible to (Hendrycks et al. 2018). We postulate that designing machine learning algorithms that are both accurate and robust to adversarial attacks requires a radical reformulation in how such algorithms are currently constructed. In particular, we propose to allow our models to undergo a period of unsupervised learning that mimics the cognitive development of biological nervous systems instead of a strict reliance on reducing classification loss. This is motivated by the observation that biological systems are robust to adversarial examples that affect deep networks. However, in time-limited regimes that result in predominantly feed-forward brain computation, adversarial attacks have been shown to influence human decision making (Elsayed et al. 2018), suggesting that slower recurrent feedback loops aid in adversarial robustness. Congruent with current theories for probabilistic inference in the brain (Lee et al. 2003), we augment the standard ANN architecture with lateral connectivity and recurrence. We propose that, together, these modifications will improve the networks specificity to signals that lie along the data manifold. We use adversarial attacks as a mechanism for understanding both traditional pointwise nonlinear ANNs and a population nonlinear alternative.

Adversarial attacks represent a flaw in typical deep neural networks where small targeted perturbations in the input space can cause large undesired changes in the output space. Although the discovery of adversarial examples in deep networks is relatively new, it has had a considerable amount of academic attention and an understanding of these attacks has begun to emerge. In the pioneering work of Szegedy et al. (2013), they introduce the idea of "adversarial attacks" for deep networks and propose that a non-smooth decision

boundary in classification networks produces adversarial "pockets", or isolated regions on the data manifold, interspersed among dataset examples. They also suggest that adversarial images represent a counter-example to the hypothesis that deep networks are able to achieve local generalization to input space regions in the vicinity of training examples. Although this latter hypothesis has continued to hold, the presence of universal and transferable adversarial directions (Moosavi-Dezfooli et al. 2017; Kurakin et al. 2016a) suggests that there are not adversarial pockets in the input manifold. The follow up work of Goodfellow et al. (2014) make several proposals, some of which have been refuted (for example by Jetley et al. 2018). However, we would like to emphasize two important arguments from their work: 1) they present evidence suggesting that the direction of perturbation is more important than the specific point in space (further refuting the "pockets" theory), and 2) the adversarial perturbations are highly aligned with the weight vectors of a model. The last of those observations has been explored in much more detail by Jetley et al. (2018), who suggest that deep networks produce classification boundaries based on a small number of features that correspond to high curvature directions in the input space and that these high curvature directions are also the most effective adversarial directions. The works of Gilmer et al. (2018) and Ford et al. (2019) further expand on this idea by suggesting that adversarial attacks confined to the l_∞ box represent the worst-case example of the model's inability to account for distributional shift generally.

Recent work has also shown that nearly all proposed adversarial defense mechanisms fail in the face of more careful attacks (Carlini et al. 2017; Athalye et al. 2018). For example, the recent defense method proposed by Sun et al. (2018) claims to produce an adversarially robust preprocessing step, although they use basis pursuit in their methods, which possibly achieves robustness through gradient obfuscation (Athalye et al. 2018). However, their intuition that images should be forced onto the image manifold as a preprocessing step has great relevance to our work. The work of Jacosen et al. (2018) makes clear the trade-off between adversarial robustness and test accuracy and suggests that excessive invariance of a network should also be considered when understanding adversarial weakness. Adversarial perturbations push inputs across decision boundaries such that the semantic quality of the image is largely the same but the label is changed to an incorrect classification. It is also possible to arrive at the same result by generating an image such that the network's output is unchanged but the semantic quality is largely different, which they call an invariance-based adversary. Our proposed framework is consistent with this idea and we provide an alternative perspective for excessive invariance.

The works of Kos et al. (2018) and Goodfellow et al. (2014) demonstrate the ability to produce adversarial examples for pointwise-nonlinear generative models that have applications in image denoising. We would expect generative models to preserve information from their inputs and therefore be less susceptible to adversarial attacks. Additionally, it has been suggested that denoising in general should help with robustness. This work suggests a more nuanced explanation of adversarial susceptibility.

It will always be possible to produce an image perturbation that results in a change in class label, or an alternate reconstruction. The academic interest in the subject comes from

observations of how small that perturbation can be and the (lack of) semantic content of the perturbation. Following the intuition provided in (Ford et al. 2019), successful defenses appear to require accounting for, or at a minimum detecting, distributional shift from the training data.

Neuron response geometry predicts adversarial robustness

In section 4.2, we visualized the input-output maps of model neurons in the form of iso-response contours to better understand the difference between pointwise and population nonlinearities. Golden et al. (2016) used iso-response contours to explain several non-linear properties as well as variance/invariance properties for V1 simple and complex cell models. Vilankar and Field (2017) further elaborate on this theory by proposing that neurons with exo-origin bent iso-response contours have a high degree of selectivity to their preferred stimulus. In this section we propose to adopt this single-neuron iso-response analysis to better understand adversarial attacks on ANNs with varying degrees of biological realism. We use iso-response contours as a lens to understand untargeted adversarial perturbations for individual neurons and predict important properties about the neurons' susceptibility to adversarial attacks. We show that the response geometry of LCA neurons predicts data-aligned adversarial perturbations, resulting in semantically meaningful adversarial attacks. We conduct our experiments on networks with both pointwise and population nonlinearities. Although there are many types of networks that fit into these two categories, we focus this study on three specific instances: feedforward autoencoders (AEs), feedforward multi-layer perceptrons (MLPs), and the Locally Competitive Algorithm (LCA). All analysis in this section was done using the MNIST dataset, which was normalized to have zero mean and unit variance.

Adversarial examples are closely tied to neuron iso-response contours. While an iso-response contour represents a perturbation direction in stimulus space that produces no change in the output, an adversarial example is a perturbation direction that produces a maximal change in the output. Below, we present analytic arguments that these two response directions are orthogonal for individual neurons.

If we extend the work of Marzi et al. (2018) to include nonlinear functions, then we can define adversaries as seeking to maximize the following objective, Δ, with respect to a perturbation, e:

$$\begin{aligned} \max_e \Delta(s, s+e) = \max_e |f(s+e) - f(s)| \\ s.t. \|e\|_\infty < \varepsilon, \end{aligned} \tag{4.1}$$

where $f(\cdot)$ can include a non-linear activation function, s and e are equal sized column vectors representing the data and perturbation, respectively, and ε is some small scalar value. To solve this equation, one must find a perturbation that results in a model output that is maximally different from the output with respect to the original input. For our

analysis, e indicates a targeted perturbation of fixed magnitude that is subject to $\|e\|_\infty < \varepsilon$. By definition, for any vectors u and v, $\langle u, v\rangle = |u| \cdot |v| \cdot \cos(\Theta_{u,v})$. Going forward, we will reference the inner angle between two vectors as $\Theta_{u,v}$ for some vectors u , v. If we assume that the perturbation magnitudes are fixed and equal, i.e.$\|e_{adv}\| = \|e_{iso}\| = k < \varepsilon$, then we can optimize over the angle between the perturbation vector and the input signal $\Theta_{e,s}$. We can now include perturbation directions that result in a minimal change in the neuron's output:

Adversarial	**Iso-Response**
$\max_{\Theta_{e,s}} \lvert f(s+e) - f(s)\rvert$	$\min_{\Theta_{e,s}} \lvert f(s+e) - f(s)\rvert$

These iso-response directions define contours in the neuron's response field. We can use the Fréchet definition of the derivative to better understand how the neural response geometry relates to adversarial images:

$$\begin{aligned} f(s+\varepsilon) &= f(s) + \langle \nabla_s f(s), \varepsilon\rangle + o(\varepsilon) \\ &\therefore \\ f(s+\varepsilon) - f(s) &= \langle \nabla_s f(s), \varepsilon\rangle + o(\varepsilon), \end{aligned} \tag{4.2}$$

where $\langle , \rangle$ indicates a dot product and $o(\varepsilon)$ is a tight bound that means "terms that are, in the limit $\varepsilon \to 0$, dominated by ε". This is of similar character to a Taylor series expansion. We can now combine these ideas to rewrite the adversarial and iso-response attack objectives:

$$\begin{aligned} &\textbf{Adversarial:} \max_{\Theta_{e,s}} |\nabla_s f(s)^\top e + o(\|e\|)|, \\ &\textbf{Iso-Response:} \min_{\Theta_{e,s}} |\nabla_s f(s)^\top e + o(\|e\|)|, \end{aligned} \tag{4.3}$$

where $\nabla_s f(s)$ is the output gradient of the target neuron. For small $\|e\|$, the linear terms in equation (4.3) dominate and these equations are solved when $e \parallel \nabla_s f(s)$ for the adversarial perturbation and $e \perp \nabla_s f(s)$ for the iso-response perturbation. This tells us that the optimal adversarial directions will always be perpendicular to the iso-response directions for an untargeted adversarial attack.

Estimating adversarial directions

The response function of a model neuron for a perturbation, e, added to an input signal, s, can give us insight to that neuron's iso- and adversarial-response properties. First, we will describe this for a linear neuron, $f_k(s) \coloneqq \phi_k^\top s$:

$$\begin{aligned} f_k(s+e) &= \phi_k^\top (s+e) \\ &= \phi_k^\top s + \phi_k^\top e, \end{aligned} \tag{4.4}$$

where ϕ_k is a column weight vector. Next we match equation (4.4) to the top part of equation (4.2):

$$f(s) + \langle \nabla_s f(s), e \rangle + o(\|e\|) = \phi_k^\top s + \phi_k^\top e. \tag{4.5}$$

This tells us that the first term on the right hand side is $f(s)$ and the second term is $\langle \nabla_s f(s), e \rangle$, which means the third term is $o(||e||) = 0$ (this is expected because it is a linear function). Therefore, $\nabla_s f(s) = \phi_k$ and for any $e \perp \phi_k$, $f_k(s + e) = f_k(s)$.

For a fixed amplitude perturbation, the maximum adversarial direction is when the inner angle $\Theta_{\phi_k, e} = 0$. Additionally, perturbations that are orthogonal to ϕ_k are iso-response directions because they do not change the function's output, which is illustrated in figures 4.1 and 4.3.

We can again use equation (4.2) to gain insight about the iso-response directions and adversarial directions for pointwise nonlinear functions, such that $f_k(s) \coloneqq g_k(\phi_k^\top s)$:

$$\begin{aligned} f_k(s + e) &= g_k(\phi_k^\top (s + e)) \\ &= g_k(\phi_k^\top s + \phi_k^\top e) \end{aligned} \tag{4.6}$$

where $g_k(\cdot)$ is a pointwise non-linear activation function for neuron k. If we perform a change of variables such that $a = \phi_k^\top s$ and $b = \phi_k^\top e$, then we can fit equation (4.6) to match equation (4.2):

$$\begin{aligned} f_k(s + e) = g_k(a + b) &= g_k(a) + \nabla_s g_k(a) \cdot b + o(b) \\ &= g_k(\phi_k^\top s) + \nabla_s g_k(\phi_k^\top s) \cdot \phi_k^\top e + o(\phi_k^\top e) \\ &= g_k(\phi_k^\top s) + \langle \nabla_s g_k(\phi_k^\top s) \cdot \phi_k^\top, e \rangle + o(\|e\|), \end{aligned} \tag{4.7}$$

Now we match equation (4.7) to the bottom part of equation (4.2) to show

$$|f_k(s + e) - f_k(s)| = |\nabla_s g_k(\phi_k^\top s) \cdot \phi_k^\top e + o(\|e\|)|. \tag{4.8}$$

Again, the function is maximized when $\phi_k \parallel e$ and minimized when $\phi_k \perp e$, which tells us that a pointwise nonlinearity can only change the spacing between the contours, it cannot bend the contours in any way.

Finally, we extend this analysis to population nonlinear networks. For population nonlinear networks, the output is now a function of the the other neurons in the layer as well as the input. Thus, Φ represents the entire weight matrix, with rows set to $\phi_k^\top$. To determine the activation of a single neuron, k, we use a one-hot selection vector, p_k. Thus, we define the activation function, f_k as:

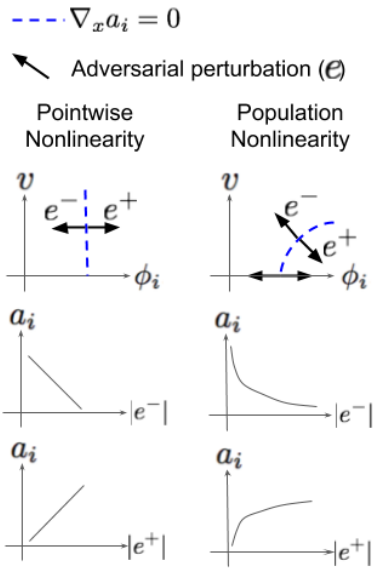


Figure 4.6: **Schematic description of adversarial attacks relative to iso-response contours.** Adversarial gradients (dark arrows) are always orthogonal to the iso-response contours (blue dashed line).Φ_k indicates a weight vector for the neuron a_k, and v represents a random orthogonal direction.

$$\begin{aligned} f_k(s+e) &\coloneqq p_k^\top g(\Phi(s+e)) \\ &= p_k^\top g(\Phi s + \Phi e) \\ &= p_k^\top g(\Phi s) + (\nabla_s g(\Phi s)^\top p_k)^\top \Phi e + o(\|\Phi e\|) \\ &= p_k^\top g(\Phi s) + (\Phi^\top \nabla_s g(\Phi s)^\top p_k)^\top e + o(\|e\|), \end{aligned} \tag{4.9}$$

where $g(\cdot)$ is a function of the weight matrix times the input, and $\nabla_s g(\Phi s)$ is its Jacobian. We can no longer simply describe the contours of these neurons because the gradient of the non-linearity is now a function of (an arbitrary linear transformation of) s. Thus, for individual neurons, the contours can be straight or bent.

Although we know that the adversarial directions are perpendicular to iso-response contours, we cannot analytically derive the directions for population nonlinearities. Instead, we can estimate adversarial perturbation directions by measuring the iso-response contours empirically. Figure 4.6 shows schematic drawings of the contours measured for population- and pointwise-nonlinear neurons along with the expected adversarial directions.

4.5 The LCA provides defense against adversarial attacks

Recall that the adversarial objective in equation (4.1) has an absolute value and is thus bi-directional. For a locally linear approximation of an untargeted attack, these two directions are equal. However, if we impose a rectifying constraint on our neuron outputs, then the two solutions are not equal for an individual neuron. For a sufficiently large e, the direction that lowers the neuron's output to below threshold will result in $f(s+e) = 0$, i.e. $\max_{e^-} |f(s+e) - f(s)| = f(s)$. This means that, for rectified units, $\max_{e^-} |f(s+e) - f(s)|$ could be less than $\max_{e^+} |f(s+e) - f(s)|$. This would not be known to a simple gradient method for computing a non-targeted adversarial attack, such as that proposed in equation (4.1). However, in practice actual adversarial attacks have an additional goal in mind: to produce an output corresponding to some other data sample. Turning off all of the units in a layer is not a solution to this problem, so some set of adversarial perturbations must be in the e^+ direction. By definition, these perturbations will have greater contribution to the model's output than e^- perturbations.

If the LCA is rectified, then the e^- direction will eventually turn off units. From figure 4.6 one can deduce that the only e^+ direction that will cause the output of neuron k to grow without bound is along Φ_k. What's more, *all* directions that increase that neuron's output are towards, or along, Φ_k. From this analysis we hypothesize that adversarial attacks on networks with exo-origin bent contours will be more aligned with the their weight vectors than attacks on networks with straight contours. If the weight vectors are data aligned then this will result in adversarial attacks that are data aligned. Recall from section 4.2 that the LCA has exo-origin curvature in nearly all orthogonal directions. Therefore, the high-dimensional iso-response geometry is that of an irregular hyper-cone, which indicates protection from orthogonal perturbations. In figure 4.7, we show that the LCA network has qualitatively more data-aligned perturbations for an untargeted attack when compared to rectified and sigmoid autoencoders with a single hidden layer. The rectified network performed worse, likely because of the unbounded nature of the activation function. However, even the bounded sigmoid network produced noisy adversarial images with pixels that are irrelevant to the stimulus space becoming active.

Untargeted attacks

Deep networks can approximate complex functions if they have enough training data. Therefore, it is reasonable to expect a deep network to learn to produce exo-origin bent iso-response contours from neurons in its latent space if efficiency or selectivity is a primary goal of the model. To test this, we trained a deep rectified bottleneck autoencoder and measured its iso-response curvature as well as its susceptibility to single-unit untargeted adversarial attacks. The autoencoder is trained to produce high quality reconstructions (approximately matched to the LCA) while communicating the information through a restricted channel of

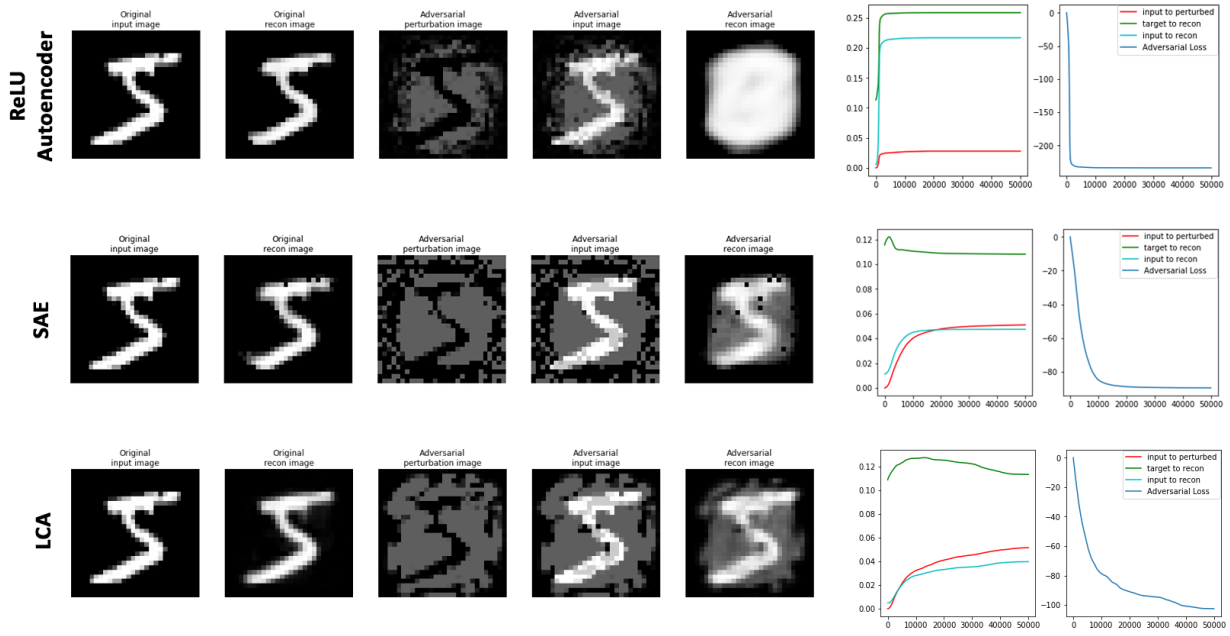


Figure 4.7: **LCA results in more data-aligned adversarial attacks.** Here we compare the LCA against shallow (single hidden layer) autoencoders with rectified and sigmoid nonlinearities. Although all three networks learn similar structured basis functions, the LCA model shows the most data-aligned adversarial perturbations. The attack for this figure follows equation (4.1), which is an untargeted attack on the network output. All networks in this experiment had 768 hidden units.

64 neurons, which should encourage an efficient representation. We were unable to train a deep architecture with a sigmoid activation that learned structured weights for comparison. Figure 4.8 shows that the autoencoder learns both exo- and endo-origin curvatures, which makes the quality of adversarial attacks less easy to predict. To produce the curvature plots, we had to select a relevant image plane for the target neuron. Following the methods outlined in (Mahendran et al. 2016), we used gradient ascent to find an optimal stimulus for a given latent neuron. We then followed the same procedure as outlined for figure 4.1 with the neurons' optimal stimuli as our target and comparison vectors.

In figure 4.9, we show that if we perform the attack defined in equation (4.1) on a single neuron, such that we minimally change the input while maximally activating a single neuron, then the output for the deep ReLU autoencoder becomes similar to that neuron's optimal stimulus. However, the LCA network simply enhances the feature associated with the target neuron without destroying other structure in the image.

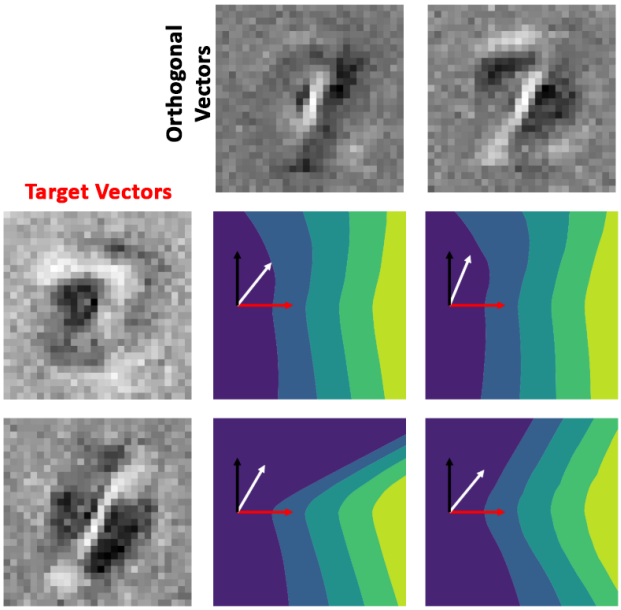


Figure 4.8: **Deep ReLU autoencoders exhibit non-uniform curvature.** The left plot shows example curvatures for the autoencoder neuron. On the right, a single neuron was selected and curvatures were computed following the procedure used for figure 4.2.

Targeted attacks

For the targeted MNIST experiments, we followed the procedures outlined in (Kos et al. 2018; Kurakin et al. 2016b; Carlini et al. 2017) to construct various adversarial attacks. We consider the LCA as an autoencoder model, $\hat{s} = g(f(s))$, which encodes images into a latent space, $a = f(s)$, and then produces reconstructions from this encoding, $\hat{s} = g(a)$. We compare the LCA against the following models:

- A sparse autoencoder (SAE, A. Ng et al. 2011) with a single overcomplete (768 units) hidden layer that uses a pointwise sigmoid nonlinearity.
- A deep bottleneck pointwise rectified (ReLU) (Hahnloser et al. 2000; Nair et al. 2010) autoencoder with 5 encoding layers that have 768, 256, 256, 128, and 64 units, a dropout keep probability of 50%, 70%, 70%, 70%, and 100%, and a mirrored decoder.
- A deep variational autoencoder (Kingma et al. 2013) with three encoding layers that have 768, 512, and 50 units, respectively, with a mirrored decoder. We used the leaky-ReLU activation function (Maas et al. 2013) with dropout keep probability of 80% for the first and last two layers and 100% otherwise. We also used a weight decay regularization loss with a trade-off multiplier of 0.003.
- A two-layer ReLU multi-layer perceptron (MLP) discriminative model.

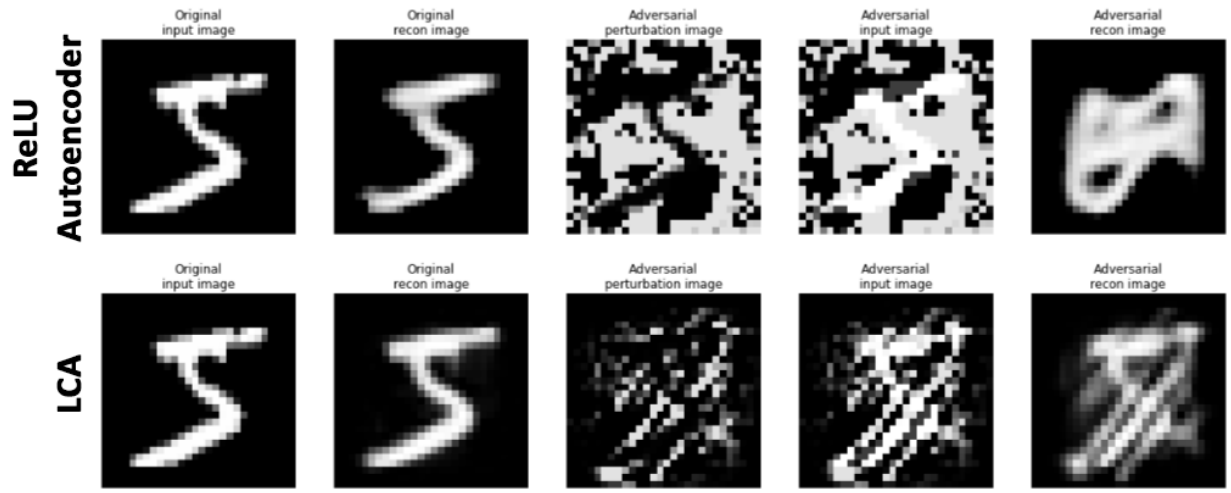


Figure 4.9: **LCA preserves image structure with single unit adversarial attacks.** We attack individual latent units in a deep rectified autoencoder and the LCA. The attack destroys any structure corresponding to the original digit class for the rectified autoencoder, but the structure is largely preserved for the LCA attack.

- An alternative implementation of the ISTA sparse coding framework called Learned ISTA, (Gregor et al. 2010).

In figure 4.12, we show that the LCA requires stronger adversarial perturbation magnitudes to achieve the same change in the network output when used as an initial encoding layer for classification tasks. In figure 4.14, we show the improved robustness of the LCA against more standard feedforward autoencoders using a generative adversarial attack. Adversarial perturbations for the LCA are more aligned with the data than those for the alternative models tested, which we demonstrate in figures 4.12 and 4.10. We also found that for attacks that did not include a clipping step to force the perturbation to be a certain size, the SAE perturbation grew without bound while the LCA perturbation settled to a fixed distance from the input (figure 4.11). We have shown that these properties can be explained by an LCA neuron's exo-origin bent iso-response contours. We believe that this study synergizes with the works of (Zhu et al. 2013; Golden et al. 2016; Vilankar et al. 2017; Olshausen and Field 1997), which together suggest that the explaining-away process inherent in the sparse coding model produces a more selective and robust code of input data.

Comparisons against a sparse autoencoder

In section 4.4, we proposed that adversarial gradient directions for LCA neurons will point in data directions. To assess whether this result extends to the entire network, we measured the cosine similarity between an input image and a successful generative adversarial perturbation using the iterative fast sign gradient (Kurakin et al. 2016b) variant of the generative adversarial

attack defined in (Kos et al. 2018). Figure 4.10 demonstrates that adversarial images for the LCA have a higher cosine similarity to the target image than the SAE after a fixed number of equal-sized perturbations. This indicates that the perturbations are more closely aligned to data directions, which should result in more semantically meaningful perturbations.

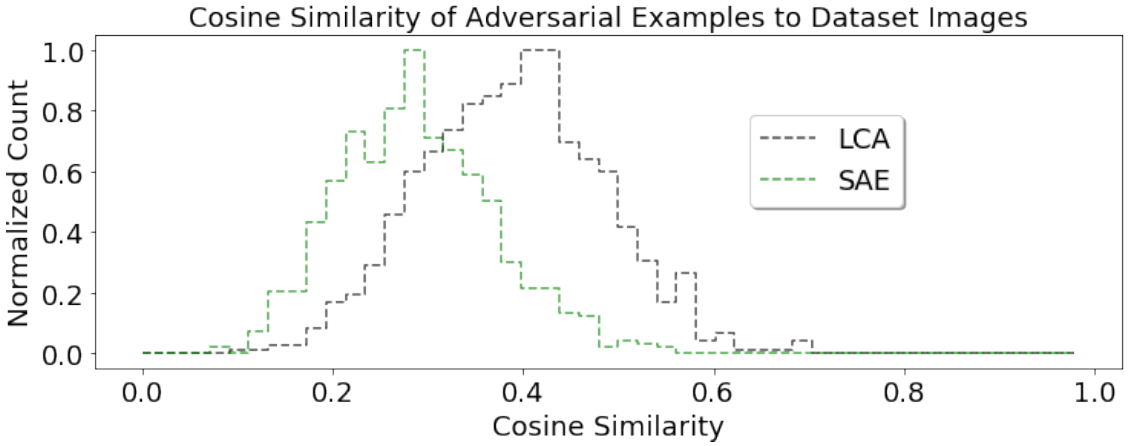


Figure 4.10: **Adversarial attacks using gradients from the LCA have a larger cosine similarity to dataset images than attacks for the SAE.** Following (Kos et al. 2018; Kurakin et al. 2016b), we attacked each model using a target image that had an alternate class label. The histograms are computed over 800 image examples.

In addition to being data-aligned, we found that the sigmoid nonlinearity in particular was susceptible to unbounded adversarial attacks. To demonstrate this, we again followed the iterative fast sign gradient method from (Kurakin et al. 2016b) but applied to the generative adversarial models as is done in (Kos et al. 2018). We additionally removed the clipping step and allowed the perturbation vector to grow without bound. Figure 4.11 shows that the LCA is protected against this unbounded attack and the vector ultimately settles on some length. However, the perturbation vector for the SAE model continues to grow, which we hypothesize is a consequence of the sigmoid nonlinearity.

Replacing the first layer of an MLP with an LCA layer boosts robustness

In chapter 3 we discussed potential benefits of using LCA as an unsupervised training method for semi-supervised classification tasks. Here, we show that the same network setup as we described in section 3.3 has additional robustness to adversarial attacks. Although we did not include feedback during inference in our adversarial experiments, we do identify this as a promising direction for future work. In this experiment we first trained a two-layer fully-connected ReLU (MLP) model on the MNIST dataset as our control. Our comparison model is the LCA trained without supervision and a single-layer perceptron (SLP) classifier trained on LCA activations. Both models were trained to have comparable validation accuracy. Figure 4.12 gives the original input to perturbed image MSE for three attack types: most-

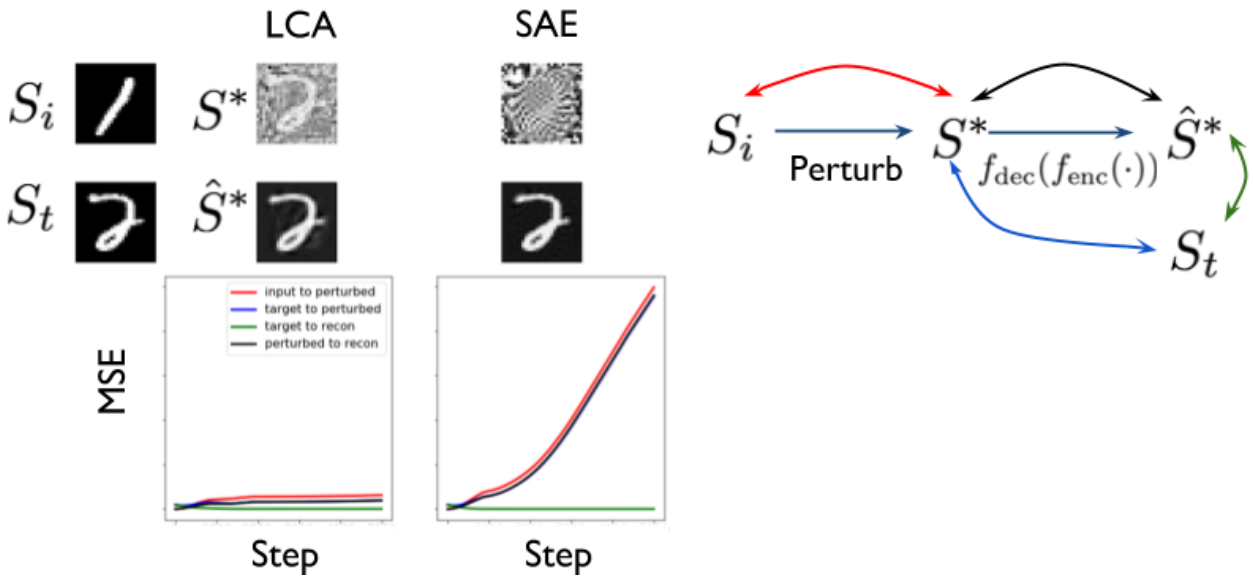


Figure 4.11: **The LCA is protected against unbounded adversarial attacks.** We attack a single layer sparse autoencoder and the LCA network with an unbounded Kurakin attack (Kurakin et al. 2016b). The attack is able to find a perturbation direction for the SAE model that can grow without bound while producing an approximately equivalent reconstruction. However, with the LCA model the attack converges and the perturbation vector is unable to continue to grow.

likely adversarial targeted, least-likely adversarial targeted, and untargeted. The first attack type takes the second highest candidate class (in terms of the softmax confidence output) and adversarially perturbs the input to that class. The second type takes the lowest candidate class and perturbs to it. The third attack minimizes the negative of the original cross-entropy training loss. The figure also shows reconstructions for a "7" digit adversarially perturbed to a "6". It is plain to see that for equal confidence the LCA+SLP model requires more semantically significant perturbations in that the perturbed image looks much more like a "6". This result was consistent for many of the attack images that we sampled.

Generative adversarial attacks

Results from (Kos et al. 2018; Gondim-Ribeiro et al. 2018; I. J. Goodfellow et al. 2014) show that adversarial stimulus can be constructed for generative models. This is especially compelling because generative models are explicitly trained to preserve information about the input and produce a veridical reconstruction, whereas classification networks are typically trained to throw away large amounts of information and only preserve that which is relevant for the given task. We show that the LCA model is robust against generative adversarial attacks when compared to the deep ReLU AE, SAE (A. Ng et al. 2011) and VAE (Kingma et al. 2013) networks.

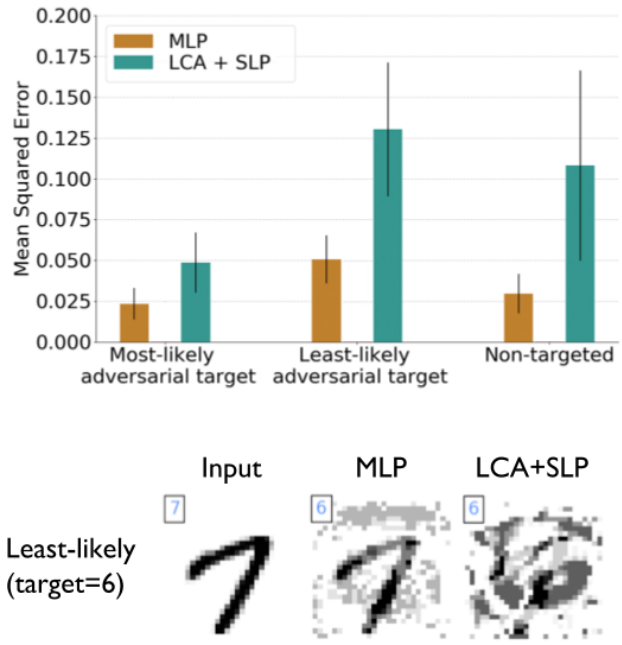


Figure 4.12: **LCA protects against traditional adversarial attacks.** Here we perform three types of attacks based on the iterative fast sign gradient method (Kurakin et al. 2016b; Kos et al. 2018). For all three types of attacks the LCA + SLP network outperforms the equal-sized MLP network. Both were trained to comparable validation accuracy. The lower images are example adversarial attacks with equal classification confidence. The blue digit in the top left indicates the original label for the left-most image and the adversarial target for the right two images.

For the first generative attack, an input image, s, was minimally perturbed before being passed through the autoencoder to produce a reconstruction of an entirely different target image, s_t. We reproduced the Carlini style attack (Carlini et al. 2017; Kos et al. 2018) using the following equation:

$$\mathcal{L} = \frac{1}{2}\|s_t - \hat{s}^*\|_2^2 + \frac{\alpha}{2}\|s - s^*\|_2^2. \tag{4.10}$$

The LCA shows robustness against this attack in that the adversarial input has a strong resemblance to the target output category, or the input and output look like a superposition of the two classes. However, for the deep rectified autoencoder, it was possible to produce an attack which had an input that can be identified as one class and an output that is identified as an alternate class. We demonstrate this result in figure 4.13.

For the second generative attack, the perturbation was done by minimizing a loss that resembles equation (4.10), but modifies the trade-off parameter to be a balanced weighting between the perturbation magnitude and the distance between the reconstruction and the target:

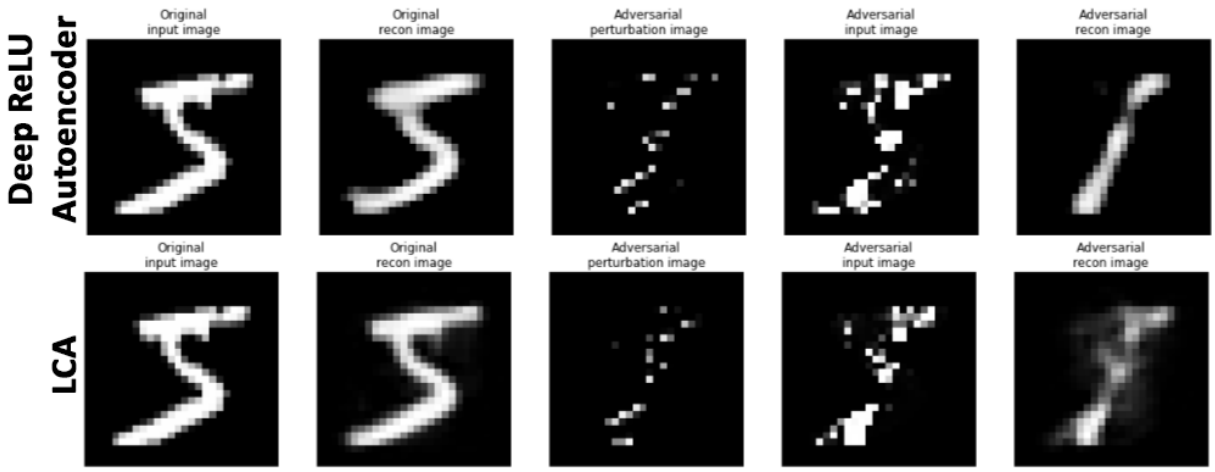


Figure 4.13: **Generative adversarial attacks succeed on deep rectified networks but fail on the LCA.** The deep rectified network can be attacked such that the input resembles one digit class and the output resembles another. Single-layer autoencoders (not shown) and the LCA are protected against this attack. The attack follows the methods outlined in (Kos et al. 2018) and equation (4.10).

$$\mathcal{L} = \frac{\alpha}{2}\|s_t - \hat{s}^*\|^2 + \frac{1-\alpha}{2}\|s - s^*\|^2. \tag{4.11}$$

where $s^* = s + e$ is the perturbed image. This modification allows us to sweep the α parameter between the two loss terms. We show in figure 4.14 that the LCA outperforms the comparison models for all parameters tested.

Comparing to the LISTA network

The lateral connectivity in the LCA network is implemented via a recurrent computation. These dynamics facilitate a form of explaining away, where neurons that have a high degree of correspondence with the input stimulus suppress other neurons in the network. This results in the network ignoring input pixels that are not aligned with primary data directions. To better understand the role of the recurrent computation, we trained a network that acts as an unrolled version of LCA, where the lateral connectivity and feedforward weight matrices are learned. The LISTA model is a recurrent network with many fewer inference steps that can produce codes that have a small Euclidean distance to the outputs of sparse coding (Gregor et al. 2010). We trained three LISTA models with 5, 20, and 50 layers. All three models were trained to produce codes that had approximately the same l_2 distance from the codes produced by 50 steps of LCA inference. We show in figure 4.15 that the LISTA network does not perform as well as LCA at defending against adversarial attacks and that the deeper LISTA networks perform no better. The attacks follow methods in (Carlini et al.

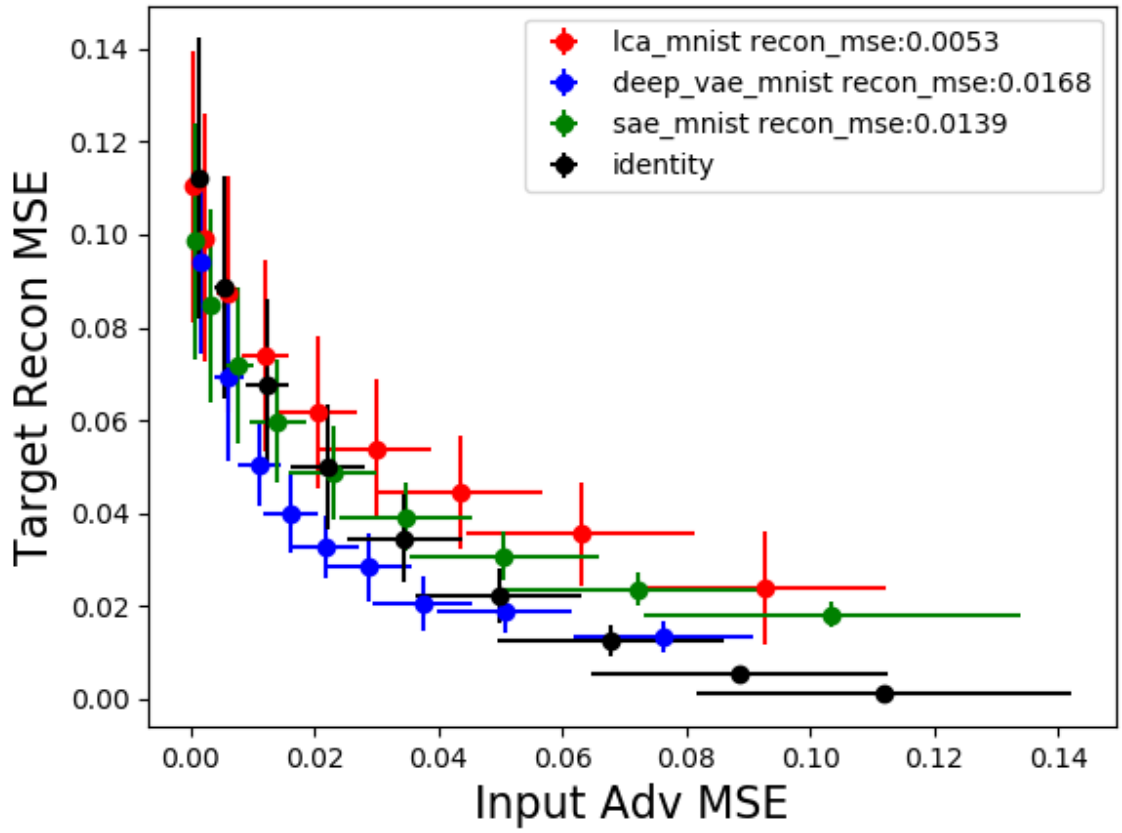


Figure 4.14: **LCA protects against MNIST generative adversarial attacks.** The Carlini-style generative attack introduced in (Kos et al. 2018) has a trade-off parameter for importance between the target image to reconstruction MSE and the input image to perturbed image MSE. When we sweep this parameter using the loss defined in equation (4.11), we find that the LCA consistently outperforms comparison models. The numbers in the labels represent the average reconstruction MSE for each model. Error bars indicate standard deviation for 100 images.

2017, "Carlini" targeted attack) and (Kurakin et al. 2016b, "Kurakin" untargeted attack). For the Carlini attack we ran 10,000 steps with a step size of 0.001 and report the resulting MSE. The average adversarial confidences for the Carlini attack were 55%, 55%, 56%, and 48% for LISTA 5, 20, 50, and LCA, respectively. For the Kurakin attack we run the attack until it reaches an average adversarial confidence of $\sim 95\%$.

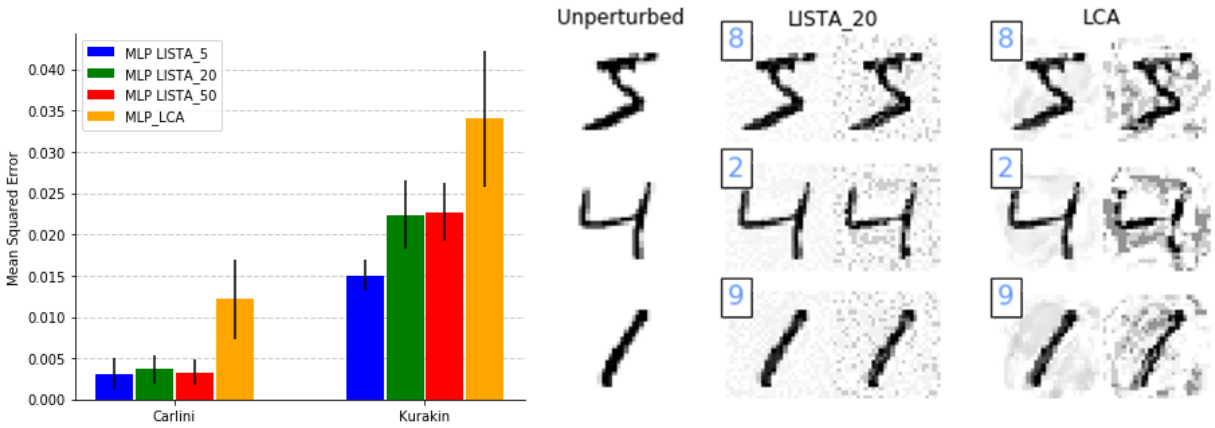


Figure 4.15: **LCA protects better than LISTA against latent classification attacks.** We generated adversarial examples from equal-sized MLPs trained from the latent codes of each model. MSEs were calculated between the original image and the perturbed image. Error bars indicate standard deviation for 100 example inputs. The images on the right show the resulting adversarial attacks. The left most column are the original unperturbed images. The next column has adversarially perturbed inputs the 20 layer LISTA model using the Carlini method. Next are perturbed inputs using the Kurakin method. The fourth column are adversarially perturbed images using the Carlini method on the LCA model. The final column are adversarially perturbed images using the Kurakin method on the LCA model. The blue digit in the top left of the Carlini attack images is the target digit class for each adversarial attack.

Discussion

Although the softmax nonlinearity used in most classification models is a population nonlinearity, it has little effect on the computation performed by the neural network as it can be, and is, without loss of generality, incorporated into the loss function. We hypothesize that the deep network does not produce consistently exo-origin bent contours in class-relevant data directions, which will result in more adversarial susceptibility.

Additional control models need to be explored, including alternative population nonlinearities such as those present in Boltzmann machines (Salakhutdinov et al. 2009), divisive normalization (Ballé et al. 2016), and local response normalization (Krizhevsky, Sutskever, et al. 2012). Each of these nonlinearities has had significance in the deep learning and neuroscience communities. The iso-response analysis provides a methodology for contrasting them and will give us valuable insight into how each of them may respond to adversarial attacks. We also wish to scale up the models to include larger datasets of more naturalistic images.

The hierarchical extension to the sparse coding model proposed in (Y. Chen et al. 2018) has been shown to perform a better job of mapping input data onto a smooth manifold. We hypothesize that this will further increase the semantic relevance of attacks and model

robustness against adversarial perturbations. We identify the model defined in (Y. Chen et al. 2018) as a candidate for exploring how the adversarial and iso-response properties change as we increase the network depth.

This methodology has a high potential for impact in the deep learning community. We advocate for biologically motivated computations that go beyond the simple pointwise nonlinear model. We have shown that these types of networks learn a more robust representation of data without tedious and biased human labeling. We also provide strong theoretical support for our hypotheses and an analysis method that allows us to fully characterize how the more complicated neurons will respond to input perturbations.

4.6 Conclusion

In this chapter we expanded on a recently proposed method for understanding neurons by measuring their response geometry. While earlier work only demonstrated neuron response contours on toy datasets with very low dimensional stimulus, we outlined a scalable method for computing neuron iso-response contours on image patches and MNIST digits. Additionally, we performed a population-level analysis to show that the LCA has exo-origin bent contours in almost all orthogonal directions. We provide evidence supporting hypotheses from the literature that exo-origin curvature is indicative of a higher degree of selectivity and efficiency. Finally, we propose that the response contours can be used to predict iso-response directions. We provide a mathematical justification as well as experimental support for this hypothesis.

Chapter 5

Conclusion

5.1 General Conclusions

In this thesis we present the Locally Competitive Algorithm as a model for computation in V1. While other works (Zhu et al. 2013; Olshausen and Field 1997; Vinje et al. 2000) have demonstrated that the LCA matches well with response properties of V1 simple cells and population activity, here we dig deeper in to the actual model computation to better understand it and expand on it. Lateral connectivity is a core component of the LCA and allows for population nonlinear interactions that facilitate an explaining-away property when encoding natural stimulus. We argue that this type of connectivity, which is also ubiquitous in neural circuits, is highly beneficial for biologically relevant and machine learning relevant tasks. We demonstrate that the LCA can be expanded to produce hierarchical representations, and present several promising directions for future work in this direction. We then show that sparse inference produces more a more efficient, robust, and selective encoding over networks without lateral connectivity and recurrence. Our hope is that a detailed analysis of the model will demystify its complexity and encourage computational neuroscientists to abandon the normative pointwise linear/nonlinear neuron models in favor of those with population nonlinear interactions and dynamic inference.

5.2 Future work

We alluded to many promising directions for future work. In chapter 1, we noted that there are many important brain processing areas that are largely ignored in the sparse coding model. We see this as an important area for additional study. Work from Shan et al. (2013) and Lindsey et al. (2019) provide promising attempts to include brain regions that precede V1. In addition to trying to represent the entire pathway as a single model, there are several examples in the literature of models that account for individual vision system components. For example, Doi et al. (2007) provide a promising direction for deriving components of retinal computation from a statistical modeling framework. Cheung et al. (2016) have also

proposed that the foveated sampling found in many retinas can emerge from models trained to attend to objects in visual scenes. Ratnam et al. (2017) and Burak et al. (2010) show promising directions for understanding how the brain can utilize eye motion for computation. Olshausen et al. (1993) proposes a dynamic routing theory for computation in the thalamus. These models can be combined with models of coding in V1 to produce a more holistic account of early vision, which we identify as an exciting area of research.

In chapter 2 we noted that deriving a locally-connected variant of the LCA would be a valuable contribution (Le, Ranzato, et al. 2011; Ngiam et al. 2010). We believe this type of connectivity provides for a good trade-off between the faster-but-constrained convolutional models and the slower-but-unconstrained fully-connected models.

In chapter 3 we proposed a hierarchical model for producing invariant representations of natural scenes and we provide promising steps towards the achieving model. Modifying the hierarchical ICA model proposed in (Karklin et al. 2003) using a similar derivation as what was used for our proposed subspace LCA network would be a valuable contribution in that it would allow us to more effectively explore how overcompleteness and recurrent inference influences the learned representations and coding properties. We also proposed a semi-supervised learning framework for extending the LCA. We identified several areas of future work, namely how feedback influences inference and learning, how the depth of the classifier influences inference and learning, as well as making comparisons to alternative methods. In the next section we proposed the subspace LCA. We are currently working on drawing comparisons to independent subspace analysis (Hyvärinen et al. 2000). We think extending the model to a topographic variant and incorporating dynamic routing ideas (Olshausen, C. H. Anderson, and Van Essen 1993) are promising future directions. We are also interested in combining the subspace LCA network and the weakly-supervised objectives defined in sections 3.3.

In chapter 4 we describe and extend a method for geometrically describing the selectivity and invariance of neurons. We measured selectivity to changing orientation for various neuron models, but, as we outlined in chapter 2, the LCA learns receptive fields that also tile spatial frequency and position. Therefore, we propose to extend the orientation selectivity analysis to other image transformations. We also think that models that utilize dynamic routing or attention would exhibit particularly narrow orientation tuning. We believe adapting the iso-response contour and adversarial analyses to include the LCA variant that has feedback signals (defined in chapter 3) would also be an exciting direction for future work. Finally, the adversarial experiments we conducted should be scaled up to larger datasets to verify that robustness holds.

Bibliography

Adelson, E. H. (2000). "Lightness Perception and Lightness Illusions". In: *The New Cognitive Neurosciences* 2.

Ahissar, E. and Arieli, A. (2012). "Seeing via miniature eye movements: a dynamic hypothesis for vision". In: *Frontiers in Computational Neuroscience* 6, p. 89.

Anderson, A., Ratnam, K., Roorda, A., and Olshausen, B. A. (2019). "High acuity vision from retinal image motion". Submitted to the Journal of Vision.

Arathorn, D. W., Stevenson, S. B., Yang, Q., Tiruveedhula, P., and Roorda, A. (2013). "How the unstable eye sees a stable and moving world". In: *Journal of Vision* 13.10, pp. 22–22.

Athalye, A., Carlini, N., and Wagner, D. (2018). "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples". In: *arXiv preprint arXiv:1802.00420*.

Atick, J. J. (1992). "Could information theory provide an ecological theory of sensory processing?" In: *Network: Computation in neural systems* 3.2, pp. 213–251.

Atick, J. J. and Redlich, A. N. (1990). "Towards a theory of early visual processing". In: *Neural Computation* 2.3, pp. 308–320.

— (1992). "What does the retina know about natural scenes?" In: *Neural computation* 4.2, pp. 196–210.

Attneave, F. (1954). "Some informational aspects of visual perception." In: *Psychological review* 61.3, p. 183.

Aytekin, M., Victor, J. D., and Rucci, M. (2014). "The visual input to the retina during natural head-free fixation". In: *Journal of Neuroscience* 34.38, pp. 12701–12715.

Baldi, P. (2012). "Autoencoders, unsupervised learning, and deep architectures". In: *ICML workshop on unsupervised and transfer learning*. Vol. 1, pp. 37–49.

Ballé, J., Laparra, V., and Simoncelli, E. P. (2016). "End-to-end optimization of nonlinear transform codes for perceptual quality". In: *2016 Picture Coding Symposium (PCS)*. IEEE, pp. 1–5.

Barlow, H. B. (1961). *Possible Principles Underlying the Transformations of Sensory Messages*. The MIT Press.

— (1972). "Single units and sensation: a neuron doctrine for perceptual psychology?" In: *Perception* 1.4, pp. 371–394.

— (2009). "Redundancy reduction revisited". In: *Network: computation in neural systems* 12.3, pp. 241–253.

Barron, J. T. and Malik, J. (2012). "Shape, albedo, and illumination from a single image of an unknown object". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 334–341.

Beck, A. and Teboulle, M. (2009). "A fast iterative shrinkage-thresholding algorithm for linear inverse problems". In: *SIAM journal on imaging sciences* 2.1, pp. 183–202.

Bell, A. J. and Sejnowski, T. J. (1997). "The fffdfffdfffdindependent componentsfffdfffdfffd of natural scenes are edge filters". In: *Vision research* 37.23, pp. 3327–3338.

Bengio, Y., Courville, A. C., and Vincent, P. (2012). In: *arXiv Computing Research Repository arXiv:1206.5538*.

Bridgeman, B. (2010). "How the brain makes the world appear stable". In: *i-Perception* 1.2, pp. 69–72.

Buchsbaum, G. and Gottschalk, A. (1983). "Trichromacy, opponent colours coding and optimum colour information transmission in the retina". In: *Proceedings of the Royal society of London. Series B. Biological sciences* 220.1218, pp. 89–113.

Burak, Y., Rokni, U., Meister, M., and Sompolinsky, H. (2010). "Bayesian model of dynamic image stabilization in the visual system". In: *Proceedings of the National Academy of Sciences* 107.45, pp. 19525–19530.

Cadieu, C. and Olshausen, B. A. (2008). "Learning transformational invariants from natural movies". In: *Advances in neural information processing systems*, pp. 209–216.

Carandini, M. and Heeger, D. J. (2012). "Normalization as a canonical neural computation". In: *Nature Reviews Neuroscience* 13.1, p. 51.

Carlini, N. and Wagner, D. (2017). "Towards evaluating the robustness of neural networks". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, pp. 39–57.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). "Atomic decomposition by basis pursuit". In: *SIAM review* 43.1, pp. 129–159.

Chen, Y., Paiton, D. M., and Olshausen, B. A. (2018). "The Sparse Manifold Transform". In: *Advances in Neural Information Processing Systems*, pp. 10513–10524.

Cheung, B., Weiss, E., and Olshausen, B. (2016). "Emergence of foveal image sampling from learning to attend in visual scenes". In: *arXiv preprint arXiv:1611.09430*.

Daubechies, I., Defrise, M., and De Mol, C. (2004). "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint". In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 57.11, pp. 1413–1457.

Davis, G., Mallat, S., and Avellaneda, M. (1997). "Adaptive greedy approximations". In: *Constructive approximation* 13.1, pp. 57–98.

Doi, E. and Lewicki, M. S. (2007). "A theory of retinal population coding". In: *Advances in neural information processing systems*, pp. 353–360.

Dong, D. W. and Atick, J. J. (1995). "Statistics of natural time-varying images". In: *Network: Computation in Neural Systems* 6.3, pp. 345–358.

Eichhorn, J., Sinz, F., and Bethge, M. (2009). "Natural Image Coding in V1: How Much Use is Orientation Selectivity?" In: *PLOS Computational Biology* 4, e1000336. arXiv: `0810.2872v2 [q-bio.NC]`.

Elsayed, G. F., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I., and Sohl-Dickstein, J. (2018). "Adversarial examples that fool both human and computer vision". In: *arXiv preprint arXiv:1802.08195*.

Engbert, R., Mergenthaler, K., Sinn, P., and Pikovsky, A. (2011). "An integrated model of fixational eye movements and microsaccades". In: *Proceedings of the National Academy of Sciences* 108.39, E765–E770.

Ernst, M. O. and Banks, M. S. (2002). "Humans integrate visual and haptic information in a statistically optimal fashion". In: *Nature* 415.6870, p. 429.

Field, D. J. (1987). "Relations between the statistics of natural images and the response properties of cortical cells". In: *Journal of the Optical Society of America* 4.12, pp. 2379–2394.

— (1994). "What is the goal of sensory coding?" In: *Neural computation* 6.4, pp. 559–601.

— (1999). "Wavelets, vision and the statistics of natural scenes". In: *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 357.1760, pp. 2527–2542.

Ford, N., Gilmer, J., Carlini, N., and Cubuk, D. (2019). "Adversarial Examples Are a Natural Consequence of Test Error in Noise". In: *arXiv preprint arXiv:1901.10513*.

Gardner-Medwin, A. R. and Barlow, H. B. (2001). "The Limits of Counting Accuracy in Distributed Neural Representations". In: *Neural Computation* 13.3, pp. 477–504.

Ghodrati, M., Khaligh-Razavi, S.-M., and Lehky, S. R. (2017). "Towards building a more complex view of the lateral geniculate nucleus: recent advances in understanding its role". In: *Progress in Neurobiology* 156, pp. 214–255.

Gilmer, J., Metz, L., Faghri, F., Schoenholz, S. S., Raghu, M., Wattenberg, M., and Goodfellow, I. (2018). "Adversarial spheres". In: *arXiv preprint arXiv:1801.02774*.

Golden, J. R., Vilankar, K. P., Wu, M. C. K., and Field, D. J. (2016). "Conjectures regarding the nonlinear geometry of visual neurons". In: *Vision research* 120, pp. 74–92.

Gollisch, T. and Meister, M. (2010). "Eye smarter than scientists believed: neural computations in circuits of the retina". In: *Neuron* 65.2, pp. 150–164.

Gondim-Ribeiro, G., Tabacof, P., and Valle, E. (2018). "Adversarial Attacks on Variational Autoencoders". In: *arXiv preprint arXiv:1806.04646*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572*.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Gregor, K. and LeCun, Y. (2010). "Learning fast approximations of sparse coding". In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Omnipress, pp. 399–406.

Guillery, R. and Sherman, S. M. (2002). "Thalamic relay functions and their role in corticocortical communication: generalizations from the visual system". In: *Neuron* 33.2, pp. 163–175.

— (2011). "Branched thalamic afferents: what are the messages that they relay to the cortex?" In: *Brain research reviews* 66.1-2, pp. 205–219.

Hahnloser, R. H., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., and Seung, H. S. (2000). "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit". In: *Nature* 405.6789, p. 947.

Hateren, J. H. van (1992). "A theory of maximizing sensory information". In: *Biological cybernetics* 68.1, pp. 23–29.

Hateren, J. H. van and Schaaf, A. van der (1998). "Independent component filters of natural images compared with simple cells in primary visual cortex". In: *Proceedings of the Royal Society of London B: Biological Sciences* 265.1394, pp. 359–366.

Helmholtz, H. von (1878). *The Facts of Perception*. Wesleyan University Press.

Hendrycks, D. and Dietterich, T. G. (2018). "Benchmarking Neural Network Robustness to Common Corruptions and Surface Variations". In: *arXiv preprint arXiv:1807.01697*.

Hoyer, P. O. and Hyvärinen, A. (2003). "Interpreting neural response variability as Monte Carlo sampling of the posterior". In: *Advances in neural information processing systems*, pp. 293–300.

Hubel, D. H. and Wiesel, T. N. (1959). "Receptive fields of single neurones in the cat's striate cortex". In: *The Journal of physiology* 148.3, pp. 574–591.

Hyvarinen, A. (1999). "Fast and robust fixed-point algorithms for independent component analysis". In: *IEEE transactions on Neural Networks* 10.3, pp. 626–634.

Hyvärinen, A. and Hoyer, P. O. (2000). "Emergence of phase-and shift-invariant features by decomposition of natural images into independent feature subspaces". In: *Neural computation* 12.7, pp. 1705–1720.

Jacobsen, J.-H., Behrmann, J., Zemel, R., and Bethge, M. (2018). "Excessive Invariance Causes Adversarial Vulnerability". In: *arXiv preprint arXiv:1811.00401*.

Jetley, S., Lord, N., and Torr, P. (2018). "With friends like these, who needs adversaries?" In: *Advances in Neural Information Processing Systems*, pp. 10772–10782.

Jolliffe, I. T. and Cadima, J. (2016). "Principal component analysis: a review and recent developments". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065, p. 20150202.

Kant, I. (1790). *The Critique of Judgement*.

Karklin, Y. and Lewicki, M. S. (2003). "Learning higher-order structures in natural images". In: *Network: computation in neural systems* 14.3, pp. 483–499.

Kenyon, G. T., Theiler, J., George, J. S., Travis, B. J., and Marshak, D. W. (2004). "Correlated firing improves stimulus discrimination in a retinal model". In: *Neural Computation* 16.11, pp. 2261–2291.

Kersten, D., Mamassian, P., and Yuille, A. (2004). "Object perception as Bayesian inference". In: *Annual Review of Psychology* 55, pp. 271–304.

Kingma, D. P. and Welling, M. (2013). "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114*.

Kos, J., Fischer, I., and Song, D. (2018). "Adversarial examples for generative models". In: *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, pp. 36–42.

Kowler, E. and Steinman, R. M. (1979). "Miniature saccades: eye movements that do not count." In: *Vision research* 19.1, p. 105.

Krizhevsky, A. and Hinton, G. E. (2009). *Learning multiple layers of features from tiny images*. Tech. rep. University of Toronto.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems 25*, pp. 1106–1114.

Kurakin, A., Goodfellow, I., and Bengio, S. (2016a). "Adversarial examples in the physical world". In: *arXiv preprint arXiv:1607.02533*.

— (2016b). "Adversarial machine learning at scale". In: *arXiv preprint arXiv:1611.01236*.

Laughlin, S. (1981). "A simple coding procedure enhances a neuron's information capacity". In: *Zeitschrift für Naturforschung c* 36.9-10, pp. 910–912.

Le, Q. V., Karpenko, A., Ngiam, J., and Ng, A. Y. (2011). "ICA with reconstruction cost for efficient overcomplete feature learning". In: *Advances in neural information processing systems*, pp. 1017–1025.

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. (2011). "Building high-level features using large scale unsupervised learning". In: *arXiv preprint arXiv:1112.6209*.

LeCun, Y. (1998). *The MNIST database of handwritten digits*. URL: `http://yann.lecun.com/exdb/mnist/`.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). "Gradient-based learning applied to document recognition". In: *IEEE*. Vol. 86. 11, pp. 2278–2324.

Lee, T. S. and Mumford, D. (2003). "Hierarchical Bayesian inference in the visual cortex". In: *JOSA A* 20.7, pp. 1434–1448.

Lewicki, M. S. and Olshausen, B. A. (1999). "Probabilistic framework for the adaptation and comparison of image codes". In: *Journal of the Optical Society of America A* 16.7, pp. 1587–1601.

Lewicki, M. S. and Sejnowski, T. J. (1997). "Bayesian unsupervised learning of higher order structure". In: *Advances in neural information processing systems*, pp. 529–535.

— (2000). "Learning overcomplete representations". In: *Neural computation* 12.2, pp. 337–365.

Lindsey, J., Ocko, S. A., Ganguli, S., and Deny, S. (2019). "A unified theory of early visual representations from retina to cortex through anatomically constrained deep CNNs". In: *arXiv preprint arXiv:1901.00945*.

Livezey, J. A., Bujan, A. F., and Sommer, F. T. (2016). "On degeneracy control in overcomplete ICA". In: *arXiv preprint arXiv:1606.03474*.

Lundquist, S. Y., Paiton, D. M., Schultz, P. F., and Kenyon, G. T. (2016). "Sparse encoding of binocular images for depth inference". In: *2016 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*. IEEE, pp. 121–124.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1, p. 3.

Mahendran, A. and Vedaldi, A. (2016). "Visualizing deep convolutional neural networks using natural pre-images". In: *International Journal of Computer Vision* 120.3, pp. 233–255.

Marr, D. (1982). "Vision: A computational investigation into the human representation and processing of visual information". In: *Henry Holt and Co Inc., New York, NY* 2.4.2.

Marzi, Z., Gopalakrishnan, S., Madhow, U., and Pedarsani, R. (2018). "Sparsity-based Defense against Adversarial Attacks on Linear Classifiers". In: *arXiv preprint arXiv:1801.04695*.

McIntosh, L., Maheswaranathan, N., Nayebi, A., Ganguli, S., and Baccus, S. (2016). "Deep learning models of the retinal response to natural scenes". In: *Advances in neural information processing systems*, pp. 1369–1377.

Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., and Frossard, P. (2017). "Universal adversarial perturbations". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Ieee, pp. 86–94.

Mostofi, N., Boi, M., and Rucci, M. (2016). "Are the visual transients from microsaccades helpful? Measuring the influences of small saccades on contrast sensitivity". In: *Vision research* 118, pp. 60–69.

Mumford, D. (1994). "Pattern theory: a unifying perspective". In: *First European congress of mathematics*. Springer, pp. 187–224.

Murakami, I. and Cavanagh, P. (1998). "A jitter after-effect reveals motion-based stabilization of vision". In: *Nature* 395.6704, p. 798.

Nair, V. and Hinton, G. E. (2010). "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814.

Ng, A. et al. (2011). "Sparse autoencoder". In: *CS294A Lecture notes* 72.2011, pp. 1–19.

Ng, A. Y. and Jordan, M. I. (2002). "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes". In: *Advances in neural information processing systems*, pp. 841–848.

Ngiam, J., Chen, Z., Chia, D., Koh, P. W., Le, Q. V., and Ng, A. Y. (2010). "Tiled convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1279–1287.

Norton, T. T., Corliss, D. A., and Bailey, J. E. (2002). *The psychophysical measurement of visual function*. Vol. 362. Butterworth-Heinemann Woburn.

Olshausen, B. A. (1996). "Learning linear, sparse, factorial codes". In: *AI Memo 1580*.

— (2003a). "Learning sparse, overcomplete representations of time-varying natural images". In: *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*. Vol. 1. IEEE, pp. I–41.

— (2003b). "Principles of image representation in visual cortex". In: *The visual neurosciences* 2, pp. 1603–1615.

— (2013a). "20 years of learning about vision: Questions answered, questions unanswered, and questions not yet asked". In: *20 Years of Computational Neuroscience*. Springer, pp. 243–270.

— (2013b). "Perception as an Inference Problem". In: *The Cognitive Neurosciences*, p. 295.

Olshausen, B. A. and Anderson, C. H. (2010). "Does the brain de-jitter retinal images?" In: *Proceedings of the National Academy of Sciences* 107.46, pp. 19607–19608.

Olshausen, B. A., Anderson, C. H., and Van Essen, D. C. (1993). "A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information". In: *Journal of Neuroscience* 13.11, pp. 4700–4719.

Olshausen, B. A. and Field, D. J. (1996). "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". In: *Nature* 381.6583, pp. 607–609.

— (1997). "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision research* 37.23, pp. 3311–3325.

Olshausen, B. A. and Lewicki, M. S. (1999). "Probabilistic framework for the adaptation and comparison of image codes". In: *JOSA A* 16.7, pp. 1587–1601.

Packer, O. and Williams, D. R. (1992). "Blurring by fixational eye movements". In: *Vision research* 32.10, pp. 1931–1939.

Paiton, D. M., Brumby, S. P., Kunde, G., Schultz, P., Lundquist, S., Patel, K., and Kenyon, G. T. (2013). "Deep, Locally Competitive, Neurally Inspired Algorithms for Generating Sparse Representations from Video and Text". Work was done in collaboration with the DARPA Innovation House.

Paiton, D. M., Lundquist, S. Y., Shainin, W. B., Zhang, X., Schultz, P. F., and Kenyon, G. T. (2015). "A Deconvolutional Competitive Algorithm for Building Sparse Hierarchical Representations". In: *In press for the 9th EAI International Conference on Bio-Inspired Information and Communications Technologies*. BICT.

Papert, S. (1966). "The summer vision project. Technical Report Memo". In: URL: `http://dspace.mit.edu/bitstream/handle/1721.1/6125/AIM-100.pdf`.

Priebe, N. J. and Ferster, D. (2012). "Mechanisms of neuronal computation in mammalian visual cortex". In: *Neuron* 75.2, pp. 194–208.

Rao, R. P. and Ballard, D. H. (1997). "Dynamic model of visual recognition predicts neural response properties in the visual cortex". In: *Neural computation* 9.4, pp. 721–763.

— (1999). "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects". In: *Nature neuroscience* 2.1, pp. 79–87.

Rasmus, A., Berglund, M., Honkala, M., Valpola, H., and Raiko, T. (2015). "Semi-supervised learning with ladder networks". In: *Advances in neural information processing systems*, pp. 3546–3554.

Ratnam, K., Domdei, N., Harmening, W. M., and Roorda, A. (2017). "Benefits of retinal image motion at the limits of spatial vision". In: *Journal of vision* 17.1, pp. 30–30.

Rebollo-Neira, L. and Lowe, D. (2002). "Optimized orthogonal matching pursuit approach". In: *IEEE Signal Processing Letters* 9.4, pp. 137–140.

Rehn, M. and Sommer, F. T. (2007). "A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields". In: *Journal of computational neuroscience* 22.2, pp. 135–146.

Ringach, D. L., Shapley, R. M., and Hawken, M. J. (2002). "Orientation Selectivity in Macaque V1: Diversity and Laminar Dependence". In: *Journal of Neuroscience* 22.13, pp. 5639–5651. DOI: `10.1523/JNEUROSCI.22-13-05639.2002`. eprint: `http://www.jneurosci.org/content/22/13/5639.full.pdf`. URL: `http://www.jneurosci.org/content/22/13/5639`.

Rodieck, R. W. (1998). *The first steps in seeing*. Vol. 1. Sinauer Associates Sunderland, MA.

Rolfe, J. T. and LeCun, Y. (2013). "Discriminative recurrent sparse auto-encoders". In: *arXiv preprint arXiv:1301.3775*.

Rozell, C. J., Johnson, D. H., Baraniuk, R. G., and Olshausen, B. A. (2008). "Sparse coding via thresholding and local competition in neural circuits". In: *Neural computation* 20.10, pp. 2526–2563.

Rucci, M., Iovin, R., Poletti, M., and Santini, F. (2007). "Miniature eye movements enhance fine spatial detail". In: *Nature* 447.7146, p. 852.

Rucci, M. and Victor, J. D. (2015). "The unsteady eye: an information-processing stage, not a bug". In: *Trends in neurosciences* 38.4, pp. 195–206.

Ruderman, D. L., Cronin, T. W., and Chiao, C.-C. (1998). "Statistics of cone responses to natural images: implications for visual coding". In: *JOSA A* 15.8, pp. 2036–2045.

Salakhutdinov, R. and Hinton, G. E. (2009). "Deep boltzmann machines". In: *Artificial intelligence and statistics*, pp. 448–455.

Sanborn, S., Paiton, D. M., and A, O. B. (2018). "Efficient Coding in V1: Orientation Selectivity vs Oriented Filters". Presented at the Computational and Systems Neuroscience Conference in Denver, Colorado.

Schiller, P. H. and Malpeli, J. G. (1978). "Functional specificity of lateral geniculate nucleus laminae of the rhesus monkey". In: *Journal of Neurophysiology* 41.3, pp. 788–797.

Schmielau, F. and Singer, W. (1977). "The role of visual cortex for binocular interactions in the cat lateral geniculate nucleus." In: *Brain research*.

Schultz, P. F., Paiton, D. M., Lu, W., and Kenyon, G. T. (2014). "Replicating kernels with a short stride allows sparse reconstructions with fewer independent kernels". In: *arXiv preprint arXiv:1406.4205*.

Series, P., Lorenceau, J., and Frégnac, Y. (2003). "The "silent" surround of V1 receptive fields: theory and experiments". In: *Journal of physiology-Paris* 97.4-6, pp. 453–474.

Shainin, W. B., Paiton, D. M., and Kenyon, G. T. (2016). "Sampling l0 Sparse Codes for Task-Optimal Representations". Presented at the Neurally Inspired Computational Elements workshop in Berkeley, California.

Shan, H. and Cottrell, G. (2013). "Efficient visual coding: From retina to V2". In: *arXiv preprint arXiv:1312.6077*.

Shan, H., Zhang, L., and Cottrell, G. W. (2007). "Recursive ica". In: *Advances in neural information processing systems*, pp. 1273–1280.

Simoncelli, E. P., Freeman, W. T., Adelson, E. H., and Heeger, D. J. (1991). *Shiftable multiscale transforms*. Tech. rep. MIT and Cambridge department of Electrical Engineering.

Simoncelli, E. P. and Olshausen, B. A. (2001). "Natural image statistics and neural representation". In: *Annual review of neuroscience* 24.1, pp. 1193–1216.

Sun, B., Tsai, N.-h., Liu, F., Yu, R., and Su, H. (2018). "Adversarial Defense by Stratified Convolutional Sparse Coding". In: *arXiv preprint arXiv:1812.00037*.

Switkes, E., Mayer, M. J., and Sloan, J. A. (1978). "Spatial frequency analysis of the visual environment: Anisotropy and the carpentered environment hypothesis". In: *Vision research* 18.10, pp. 1393–1399.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199*.

Tishby, N. and Zaslavsky, N. (2015). "Deep learning and the information bottleneck principle". In: *2015 IEEE Information Theory Workshop (ITW)*. IEEE, pp. 1–5.

Torralba, A. and Oliva, A. (2003). "Statistics of natural image categories". In: *Network: computation in neural systems* 14.3, pp. 391–412.

Van Essen, D. C. and Anderson, C. H. (1995). "Information processing strategies and pathways in the primate visual system". In: *An introduction to neural and electronic networks* 2, pp. 45–76.

Vilankar, K. P. and Field, D. J. (2017). "Selectivity, hyperselectivity, and the tuning of V1 neurons". In: *Journal of vision* 17.9, pp. 9–9.

Vinje, W. E. and Gallant, J. L. (2000). "Sparse coding and decorrelation in primary visual cortex during natural vision". In: *Science* 287.5456, pp. 1273–1276.

Westheimer, G. (2008). "Was Helmholtz a Bayesian?" In: *Perception* 37.5, pp. 642–650.

Weyand, T. G. (2016). "The multifunctional lateral geniculate nucleus". In: *Reviews in the Neurosciences* 27.2, pp. 135–157.

Zeiler, M. D., Krishnan, D., Taylor, G. W., and Fergus, R. (2010). "Deconvolutional networks." In: *Cvpr*. Vol. 10, p. 7.

Zetzsche, C., Krieger, G., and Wegmann, B. (1999). "The atoms of vision: Cartesian or polar?" In: *JOSA A* 16.7, pp. 1554–1565.

Zhu, M. and Rozell, C. J. (2013). "Visual nonclassical receptive field effects emerge from sparse coding in a dynamical system". In: *PLoS Computational Biology* 9.8, e1003191.

— (2015). "Modeling inhibitory interneurons in efficient sensory coding models". In: *PLoS computational biology* 11.7, e1004353.

Zylberberg, J. and DeWeese, M. R. (2013). "Sparse coding models can exhibit decreasing sparseness while learning sparse codes for natural images". In: *PLoS computational biology* 9.8, e1003182.

Zylberberg, J., Murphy, J. T., and DeWeese, M. R. (2011). "A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields". In: *PLoS Computational Biology* 7.10, e1002250.