2017

A Mobile App Illustrating Sensory Neural Coding Through an Efficient Coding of Collected Images and Sounds

Xiaolu Zhao
Loyola University Chicago

LOYOLA UNIVERSITY CHICAGO

A MOBILE APP ILLUSTRATING SENSORY NEURAL CODING THROUGH AN EFFICIENT CODING OF COLLECTED IMAGES AND SOUNDS

A THESIS SUBMITTED TO

THE FACULTY OF THE GRADUATE SCHOOL

IN CANDIDACY FOR THE DEGREE OF

MASTER OF SCIENCE

PROGRAM IN COMPUTER SCIENCE

BY

XIAOLU ZHAO

CHICAGO, IL

AUGUST 2017

ACKNOWLEDGEMENTS

First of all, I would like to give my gratitude to my amazing professor Dr. Mark V. Albert from Loyola Computer Science Department. I am truly grateful for having the opportunity to be involved in this interesting and challenging project, and he is a marvelous director and I could never wish for more. It is my honor to be able to work with him, and I have learned more than I ever expected during my time at Loyola.

I give my thanks to my committee members. I appreciate every piece of their precious advice and the time spent on my thesis.

Many thanks to Mary Makarious, who worked on this project before me as part of the 2015 CS summer research program and who later presented the initial concept as a poster at the 2015 Society for Neuroscience conference. She did a wonderful job preparing the python code used to prototype the basic concept of the app. And many thanks for to the summer research students who worked with me on the app - Diego Tavares, Laryssa Seabra, and Carlee Bettler - they were instrumental in producing the app and deserve a fair share of the credit.

Last but not the least, I give my thanks to my loving family. They are always there for me, and have been unconditionally supportive in every way.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Sensory neuroscience in the early auditory and visual systems appears distinct not only to outside observers, but to many trained neuroscientists as well. However, to a computational neuroscientist, both sensory systems represent an efficient neural coding of information. In fact, on a computational level it appears the brain is using the same processing strategy for both senses - the same algorithm with just a change in inputs. Insights like this can greatly simplify our understanding of the brain, but require a significant computational background to fully appreciate. How can such illuminating results of computational neuroscience be made more accessible to the entire neuroscience community?

We built an Android mobile app that simulates the neural coding process in the early visual and auditory system. The app demonstrates the type of visual or auditory codes that would develop depending on the images or sounds that an evolving species would be exposed to over evolutionary time. This is done by visually displaying the derived image and sound filters based on an optimal encoding that information, and comparing them to visual representations of neural receptive fields in the brain.

Image patches (or equivalently, sound clips) are efficiently encoded using Independent Components Analysis (ICA) as a proxy for the coding objective of the early visual system. As has been observed for the past two decades, the resulting code from natural images resembles the 2D Gabor filter receptive fields measured from neurons in primary visual cortex (V1). Similarly

this efficient encoding demonstration has been done for a mixture of “natural sounds” to create linear filters resembling the gammatone filters of the spiral ganglia from the cochlea.

The app demonstrates the relationship between efficient codes of images and sounds and related sensory neural coding in an intuitive, accessible way. This enables budding neuroscientists, and even the general public, to appreciate how an understanding of computational tools (like ICA or sparse coding) can bridge research across seemingly distinct areas of the brain. This enables a more parsimonious view of how the brain processes information, and may encourage early-program neuroscientists to consider improving their computational skills.

CHAPTER I

INTRODUCTION

To understand the results of this thesis, it is important to understand the role of computational neuroscience in understanding the brain. Computational neuroscience is a field combining neuroscience and cognitive science with engineering, computer science, and mathematics. Computational neuroscience focuses on information processing in the brain's nervous system - in other words, how the brain functions in computational terms (Churchland, 1988). Traditional neuroscience approaches focus on the biological or cognitive phenomena being studied, but are often less generalizable in the way that computational approach can be used and reused to solve a variety of problems; these modeling approaches are more apt to lead to generalizable insights.

Computational neuroscience plays an important role in understanding how brains process information, particularly in sensory systems. There are many areas of traditional neuroscience that, when studied in isolation, may appear distinct and unrelated, but can be combined with other results to produce a coherent, computational model. Such a model can form a complete picture succinctly describing the experimental results and better enable prediction of future experimental results. In this thesis, we rely on a model of neural coding in the early visual system that shares deep similarities to neural coding in the early auditory system – an parsimonious analogy made possible due to the level of abstraction applied

(Lewicki, 2002). Furthermore, the coding insights have led to an improved understanding of visual development prior to eye opening, which was not even considered during the initial formulation of this approach (Albert, Schnabel & Field, 2008).

Computational neuroscience can also provide insights that reach beyond understanding the brain. Machine learning is a subfield of computer science, which develops dynamic algorithms that enables machines to learn and apply without being explicitly programmed. Despite the term 'learning' most machine learning algorithms have no direct relation to cognitive science. However, subfields within machine learning, such as semi-supervised learning approaches are beginning to borrow tools and techniques which have a rich history in cognitive and computational neuroscience. For example, deep learning is a supervised machine learning approach that uses multilayer neural networks in order to automatically extract features from high-dimensional data sets for easier classification. Deep learning neural networks currently outperform other standard machine learning classifiers in tasks that were traditionally the realm of cognitive science – for example, object and speech recognition. In this way, the insights gleaned from computational neuroscience can impact fields that are not in any way concerned about understanding how the brain works.

Level Of Analysis

When trying to understand how the brain works, it is important to be aware that there are various approaches, even when studying a single system, that provide distinct insights. For example, one can understand neural response at the level of ion channels and current (e.g. Hodgkin Huxley-level modeling), at the level of a neural circuit, or as the overall ecological goal of the animal. These various levels are not competing models, but rather distinct views of the

same system, offering various useful perspectives. To be more clear, Marr's level of analysis breaks down an information processing system into three distinct, complementary level of analysis (Marr, 1982):

- Computational theory: this level describes the goals of the system and the general problem to be solved.
- Representation and algorithm: this level describes the way in which the problem is solved, while abstracting away some low-level details.
- Hardware implementation: this level describes the physical implementation.

These three levels are not competing models, but different strategies to understand a system. Each level has its own advantages and disadvantages. The hardware level can be used when very detailed information of a system if needed, but due to the massive details contained, it is best used for a system with limited scope. The algorithmic level can be applied to systems with larger scales, but depending on the aspect being studied, there may not be sufficient detail depending on what aspect of the system is being studied.

The computational level, which is the approach that we chose for this thesis, is the highest, most abstract analysis level among the three. The computational level treats the system as a black box; in this case we do not pay attention to the details of the implementation or even the algorithms itself, but focus on the overarching goal of the system given its environment. For example, in this project we do not concern ourselves with the precise way in which the brain *initially* encodes images and sounds, but rely on the fact that the system does have access to such information for further processing (through photoreceptors in the eye and

hair cells in the ear, for example). We focus on what the system *should* do given the type of information the animal is exposed to over evolutionary time.

Efficient Coding In Sensory Neuroscience

The efficient coding approach is a computational-level modeling method to explain how the brain represents and interprets information from the outside world (Barlow, 1961). Neurons communicate through electrical impulses, meaning the sensory information that arrives to, and is processed by, the brain is primarily encoded and transmitted along the axons. For simplicity, we limit ourselves to a rate-coding model, meaning that information is transmitted through an increase or decrease of the firing rate of neural spikes. These neural spikes make up a neural code to represent sensory information, with some representations considered more efficient than others. However, the relevant details of encoding depend on *what* particular sensory information is being encoded, and precisely *how* the system is efficient.

What are the right images to study visual coding? Animals have evolved and adapted over millennia to process stimuli from the natural world. It has been critical to their survival to quickly and accurately process natural scenes, such as forests, rivers, and prairies. In this way, if we want to understand how the brain works, it is important to find a coding strategy that works best for these natural images, as opposed to other types of images. Luckily, research in efficient coding has found that natural scenes of many different types contain the same low-level statistical structure, making the distinction between different classes (e.g. forest scenes versus prairie) less important. A similar analogy can be made to understand auditory coding using "natural sounds" - a combination of sounds that an animal might hear over evolutionary time.

Precisely how is the system efficient? Efficiency metrics typically rely on the number of spikes and the number of neurons needed to represent information. One type of efficiency is known as a compact code, which tends to minimize the number of neurons used for representation (or equivalently, the numbers of 0's and 1's in a binary code). Although arguments can be made that some neural coding may be guided by a compact coding principle (e.g. like the optic nerve with a limited number of axons relative to the number of photoreceptors in the eye) generally this coding strategy is less relevant in the brain where neurons are plentiful. Another strategy, which is much more relevant to neural processing, is a sparse code. Sparse codes aim to minimize the number of spikes that each neuron fires – with some neurons firing strongly but most being relatively silent (similar to minimizing the number of 1's but being less concerned about the numbers of 0's in a binary code). Neural spikes are metabolically expensive, making this a reasonable, ecologically-relevant objective for efficient processing. The efficient coding strategy that we will focus on in this thesis is independent coding. An independent code minimizes the statistical dependencies among neurons which has a number of advantages; interestingly such a code also tend to be sparse, as will be discussed.

As we will see in the rest of this thesis, the choice of natural images (or sounds) and an assumption of a sparse/independent coding strategy is sufficient for creating a neural code similar to what we observe in primary visual cortex and the auditory nerve.

CHAPTER II

BACKGROUND

The techniques involved in this research fall into two general categories, signal processing and neuroscience. Signal processing introduces the standard computational tools of image and sound processing from an applied/engineering perspective without regard to how the brain functions. The neuroscience background introduces the relevant areas of sensory processing and the way that more traditional neuroscience characterizes these sensory areas. This chapter will end by explaining how these two traditionally distinct areas converge.

Signal Processing

Signal processing is a common task that occurs repeatedly in the technology involved in our daily lives; electronic devices such as cameras, televisions, mobile phones, and fitness wearables all process continuous signals. Consider when we talk over a phone, the variations in air pressure caused by speech are transduced into a signal; the waveform of the sound pressure variations is not transmitted directly because there is a great deal of redundancy in a speech waveform. It is often more efficient to break down a signal into a few repeating patterns to improve transmission, and simply pass along the presence of each of these patterns in the signal over time. This not only reduces redundancy, but is a way of noting the higher-order structure of the signal, which can be used for later processing or removing noise as well. We will begin by discussing one standard strategy of encoding information.

Linear Coding

There are many ways to encode information; in fact the possibilities are limitless. However, from an engineering design point of view, one of the most practical starting points is to limit ourselves to linear codes since they are fast, well defined, and theoretically tractable for many computational goals.

Linear codes are formally defined using the following functional form, where a signal vector $x = (x_1, x_2, ... x_n)$ is transformed by a series of functions $f_i(x)$ simply by multiplying each component of the original signal by a predetermined constant, a_i, for each feature, x_i, of the signal vector. In equation form that is:

$$f_i(x_1, ..., x_k) = a_0 + a_1x_1 + \dots + a_kx_k \quad \text{(Equation 1)}$$

Although this algebraic description may be quite abstract, we can note that it immediately simplifies the space of input/output functions for our encoding a signal; from an infinite, arbitrary set of possible mappings from x to f, we are now simply trying to find appropriate a_{ik} to transform the signal. Although the brain is not directly using algebra in this way, it is well understood that the brain is highly capable of summing and reweighing inputs at the cellular level. Additionally, as we will see below, because these coefficients, a_i, are associated directly with each feature of the original signal, this can help with visually interpreting a set of coefficients.

This focus on linear codes is both a boon for visually and geometrically interpreting what the encoding represents, and a major limitation in what is possible. Clearly more complex feature selection (like face-selective cells) cannot be described through a simple linear coding, but the limited, low-level types of sensory neural responses we will focus on in this thesis can

be reasonably well understood as a linear model of the inputs. Note, even the cells which we will later describe by their linear response properties to stimuli (so-called “simple cells”) are strictly not linear, but are generally approached using linear models for many of the reasons that have been suggested.

A linear filter is a means of transforming a continuous signal into another form. A linear filter performs this using a set of coefficients applied to a window over the continuous signal. To see the applicability of linear filters we can consider a few intuitive examples; there are common operations in transforming signals that can be done in a straightforward way using a linear filtering approach. For example, if we want to ‘smooth’ a 1-dimensional signal that may have added noise, we can simply use the following filter which averages the closes four samples of the original signal: $f_i = 0.25\ x_i + 0.25\ x_{i-1} + 0.25\ x_{i-2} + 0.25\ x_{i-3}$ and slide that filter function over the signal at each x_i. Similarly, if we wanted to create a filter that would be highly responsive to ‘edges’ in the signal and would not be responsive to constant-valued parts of the signal, we could create a filter $f_i = -0.5\ x_i + 0.5\ x_{i-1}$ and note that f will only deviate from 0 when there are large differences between successive samples in the original signal.

Although linear filtering is primarily introduced through the transformation of 1-dimensional (1D) time-varying signals, the concept of filtering goes well beyond a single dimension and can be performed over time or space. For example, the averaging filter mentioned above can also be applied to a 2D signal (e.g. $f_{(i,j)} = 0.25\ x_{(i,j)} + 0.25\ x_{(i,j-1)} + 0.25\ x_{(i-1,j)} + 0.25\ x_{(i-1,j-1)}$); in particular, such a filter could be applied over an image represented by a grid of pixel intensities. The resulting image represented by f would be similar to the original image, x, but would have single-pixel variations ‘smoothed’ from the image.

As we will see, the use of 2D linear filtering is the primary tool to understand neural response properties when an animal is viewing a static image.

Gabor Functions

The encoding scheme that simple cells in primary visual cortex use can be seen as a merging of two common linear filter transformation of images. A pixel-based code is the most common encoding scheme where a number represents the light intensity at each point in the image. This type of code is highly localized. A Fourier code is one that captures large-scale periodic aspects of the signal by reporting the amount of each frequency that is present. Of course, the problem with a Fourier transform is that localized structure of the signal is lost. However, there is a coding strategy used in image processing which is both localized, like a pixel code, and periodic, like a Fourier code – a wavelet code. A common choice of wavelet used to create a wavelet code is a Gabor function.

To translate these different coding strategies into approach filters, the linear filters for a pixel-based code are relatively simple - filter coefficients are ‘1’ at a given pixel, and ‘0’ everywhere else. A Fourier code effectively uses filter coefficients, which are represented by sine waves or cosine waves of particular frequencies over the entire window. However, a Gabor filter is represented as a sine or cosine wave limited in spatial extent - traditionally windowed by a Gaussian to avoid edge effects. Figures 1 demonstrates how 1D Gabor filters can be formed by simply multiplying a Gaussian and a sine wave (Prasad & Domke, 2005).

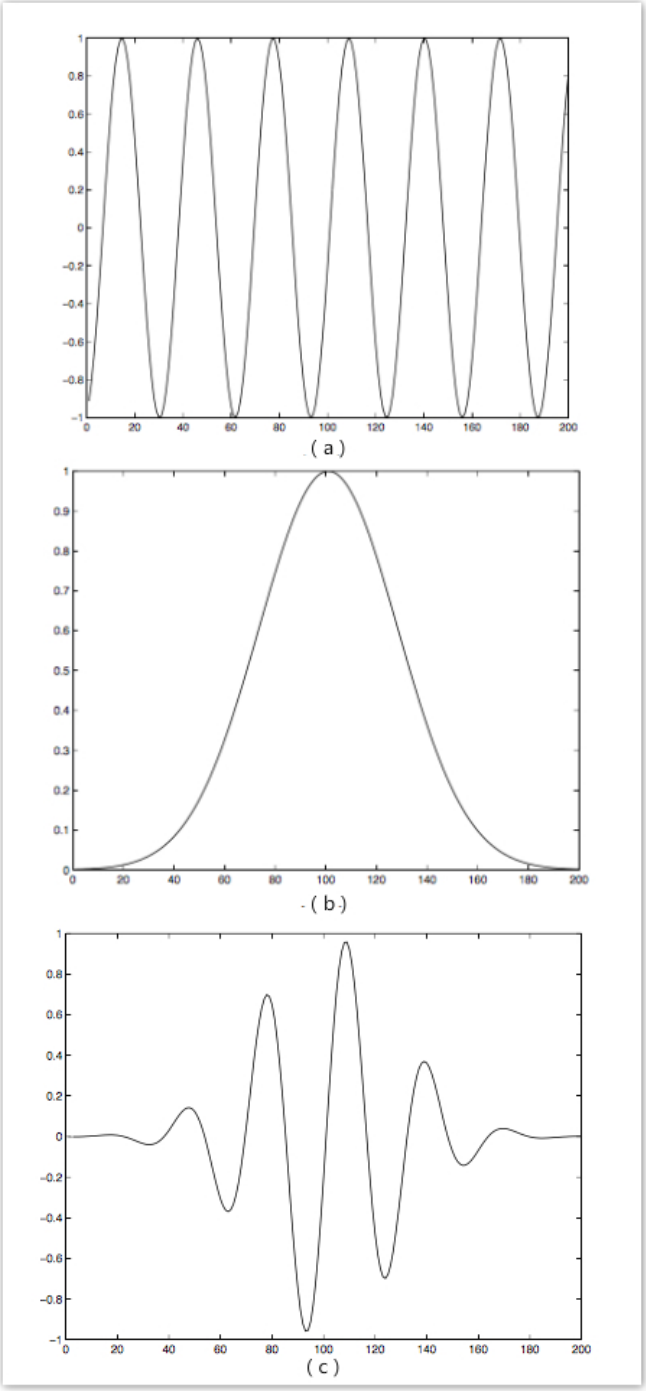

Figure 1. Visualizing The Components Of Gabor Wavelets. (a) A sinusoidal wave; (b) a Gaussian wave; (c) a 1D Gabor wave that is generated by multiplying the wave in (a) and (b).

1D Gabor filters cannot represent visual receptive fields because a visual signal is represented over two spatial dimensions. The way that Gabor filters are extended to additional

dimensions is to use a planar sine wave enveloped by a 2D Gaussian (visually represented by a series of bright and dark lines that vary sinusoidally along one dimension of an image, and are constant on the other). In fact, equation 2 demonstrates how the 1D Gabor concept is easily extended to 2D. Note, the first exponent represents the Gaussian envelop, and the second represents the sine wave using complex numbers (Lee, 1996)

$$G(x,y) = \frac{1}{2\pi\sigma\beta} e^{-\pi\left[\frac{(x-x_o)^2}{\sigma^2}+\frac{(y-y_o)^2}{\beta^2}\right]} e^{i[\xi_o x + v_0 y]} \quad \text{(Equation 2)}$$

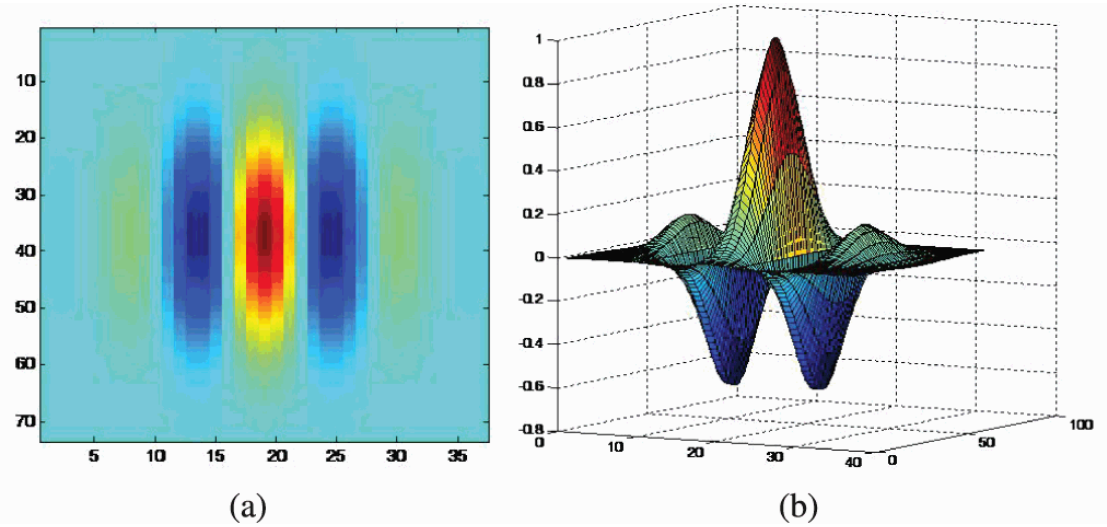

Figure 2. Visualization Of The 2D Gabor Model. Note the Gaussian window is longer along the x-axis than the y-axis ($\sigma > \beta$ in equation 1) and the planar sine wave varies along x and not along y ($v_0 = 0$) (Li & Savvides, 2013).

We will discuss in the neuroscience section how these 2D Gabor filters are used to approximate the filters describing the response properties of neurons in primary visual cortex.

Image And Auditory Processing

In order to encode images, whether on a computer or in the brain, it is important to understand how images are represented. At a fundamental level, images on digital devices are

represented as a two dimensional array of pixels. The number of pixels that make up the width and the height of the image determine the resolution of an image. In a gray scale image, the only value needed to describe a pixel is the intensity of light. When representing color images, three values are needed to describe color with various color encodings available; one most widely used is the RGB color model, which uses additive mixtures of red, green, and blue to display various colors. Although we focus on grayscale images in this thesis for convenience, it is worth noting that many of the approaches we take would work equally well using color images, which will be addressed thoroughly in the discussion section.

Sounds, or auditory signals, are fundamentally represented by time-varying variations in air pressure. A key aspect of transforming a sound to a digital signal is to sample the time-varying signal at an appropriate rate. Given the range of human hearing is roughly 2 Hz to 20 kHz; a sampling rate that captures the highest frequency of human perception is generally sufficient for encoding and reproduction.

Images and sounds can both be sampled as 1D vectors of numbers. For sounds this is straightforward. A sound signal sampled for 10 milliseconds at a 44 kHz sampling rate (which is common for human listening) provides a 440-dimension vector over that time window. 2D images can be easily clipped and transformed to 1D signals. For example, an 8 by 8 pixel patch can be selected from a larger image. This 8x8 array can be reshaped into a 64-dimension vector.

After representing images or sounds as vectors, there are many goals that further encoding of the signal can attain. A common goal, especially on computing devices, is to decrease the size of the representation - literally represent much of the same information will fewer numbers or bits. This is done in a process called compression, where we minimize the bit

size of an image (or sound), ideally while maintaining quality as much as possible. Good compression can often result in image (or sound) files that are a small fraction of the original representation in size. We will describe a common compression strategy, Principal Component Analysis (PCA), in a later section of this thesis as a common alternative efficient coding strategy that is generally not relevant in neural efficient coding in order to contrast with efficient coding objectives more in line with neural processing (e.g. ICA or sparse coding).

On a more abstract, ecological level, one of the fundamental goals of image processing in animals is the ability to identify what is represented by an image or sound. This ranges from animals having to identify predators or mates to students recognizing words and images during a presented lecture. Identification is a crucial component of daily life, and much of neuroscience and computer science is geared toward performing recognition on images or sounds.

To perform identification on images (or sounds) there is a fairly standard approach, both in neuroscience and in computer vision. Features are extracted from the original representations that are likely to help in the recognition task. For example, if trying to detect outdoor scenes, the percentage of blue in the image (representing sky) can help to identify outdoor scenes from indoor scenes. Sometimes these features can be simple, such as the amount of blue, or they can be the output of lower-level identification models themselves (e.g. detecting wheels to identify cars, detecting eyes to identify faces). These features don't need to be perfect for classification, but when used in combination with a larger set of additional features, a set of individually weak features can become quite powerful in the aggregate for

identification. Commonly, machine learning classifiers automate the mathematical combination of features to create a model capable of classifying or identifying objects.

Principal Component Analysis

In this thesis, we will discuss encoding strategies for sensory information. In particular, we will emphasize efficient encoding strategies - algorithms that encode data optimally given a set of criteria for the encoding. It is important to note that 'efficient' can have a wide variety of meanings, some which have little to do with how the brain works. For this reason, we will begin with a common efficient coding strategy that is decidedly not neurally relevant.

Principal Component Analysis (PCA) is a statistical method that has many interpretations - data compression, dimensionality reduction, finding "latent variables" in the data, etc. The main purpose of using PCA is to re-encode information with fewer variables/numbers/bits (Abdi & Williams, 2010). Generally, this is done by transforming the data to a new set of coordinates that that are ranked based on their usefulness in representing the data.

Conceptually, PCA works by successively finding axes represented by vector directions that capture the maximum variance when the data is projected on that vector. The vector which accounts for the most variance in the data is the first principal component. Then the contribution from that dimension is geometrically projected out of the data, and the process is continued until there are no more dimensions left in the data. Figure 3 provides a visual demonstration of PCA. Note, because there are as many PCA dimensions as original dimensions, and all the PCA dimensions are orthogonal, the data is fully represented if all PCA dimensions are used. However, because there is a clear ranking of components, typically only

the earlier components are used (thus reducing the dimensionality of the information being represented).

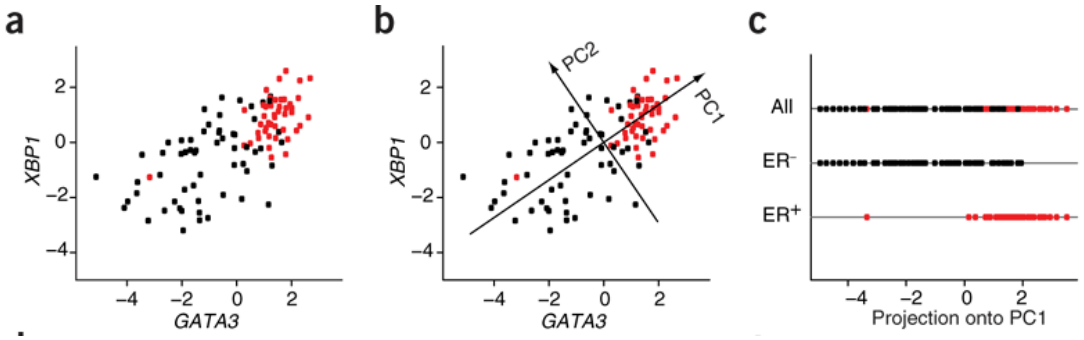

Figure 3. Illustrative Example Of Principal Components Analysis (PCA). (a) Each dot represents a sample. (b) PCA identifies the direction (PC1) along which the data has the largest variance. (c) Samples plotted using their projections onto the first principal component (PC1) (Ringnér, 2008).

PCA is a very useful technique for a variety of (again, typically non-neural) purposes, which is why when the concept of an "efficient code" is discussed it is a common choice:

- Visualizing multidimensional data. We can plot data that has 2 dimensions easily, or 3 if we are artistically skilled, but beyond that it is generally not possible to visualize data well. However, if we take a higher dimensional data set and project the data onto the first 2 or 3 principal components, we can plot the data in 2 or 3D and visually observe the relationships between the samples.
- Finding "latent" variables. Ask a person 100 questions related to personality, and we generally know that most of their responses can be explained from few aspects of their personality - introversion, agreeableness, conscientiousness, etc. If you have 100 personality questions with numeric answers, PCA can reduce each one of those questionnaires (each represented as a point in a 100-dimensional space) to only the first

few principal components. Although the meanings of the PCA components meanings may be difficult to interpret, they represent underlying aspects of personality that are predictive of response - essentially finding the “latent variables” underlying the data.

- Compression. There is a lot of redundancy in images. A 32 x 32 pixel grayscale image requires 1024 intensity values, however, many of those pixels are similar (e.g. nearby pixels on an object, or of the sky). By taking a large number of 32x32 pixel images (each represented by a 1024 length vector) and encoding using PCA, much of the original image can be represented using far fewer numbers. In fact, in this work we reduce the dimensionality of 32x32 pixel image patches from 1024 to 40 using PCA, and then submit the reduced data to neurally-appropriate efficient coding algorithms.

Figure 4. PCA Compression Examples. The original image is tiled as 32x32 pixel blocks - 15 along the width and 10 along the height. The original image (0% compression) uses all 1024 PCA components for each image patch. The 50% compressed image uses the first 512 PCA components, and is essentially indistinguishable to the naked eye. 90% compression uses only the best 100 components reducing the file size to 10% of the original. Note, that even when the image patches are represented using only 10 PCA components (1% of the original file size), the image is still recognizable (Hofman, n.d.).

Independent Component Analysis

Independent component analysis (ICA) is another type of efficient coding representing the original data, but with a different objective. Much like PCA, it is a linear transformation derived by an unsupervised learning algorithm to re-representing the data using a new set of coordinates. However, the goal of ICA is to find a representation where the response for the derived filters (the ICA component vectors) is as statistically independent as possible (Hyvärinen & Oja, 2000). In other words, given a new image, the response to one derived filter should provide very little information about the responses to other filters.

We will explain an application of ICA through its use in a classic signal processing problem, blind source separation in the “cocktail party” problem. Consider a room with as many microphones as there are sound sources. For example, one person is playing a saxophone and another is singing, and there are two microphones. The microphones would record the mixed signals of both the saxophone and the singing, however due to the different distances of the sound sources to the microphones, each microphone would capture a different combination of the sounds. Note, due to the physics of this problem, this mixture can be well modeled as a linear combination of the sound sources. Given only the mixed signals, ICA can find the linear combination of the mixed signals that is most statistically independent. In this case, that produces the original two, unmixed sounds - the saxophone player and the singer.

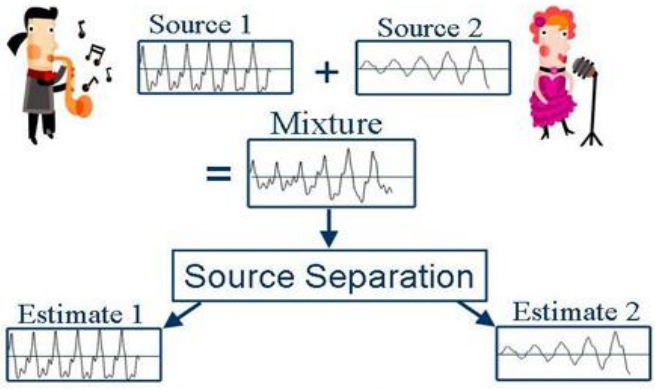

Figure 5. Steps Used In Blind Source Separation Using ICA. Source 1 is saxophone signal, source 2 is the voice of the singer. The mixture is one waveform in the center is something a single microphone might pick up in the room (though for ICA to work, two microphones are needed). By applying ICA to the mixtures, the original sounds of the saxophone and the singing can be separated even when only the two mixed signals are available.

How does ICA find these independent components? The short answer is that it searches for components that lead to data distributions projected along that component which are the least Gaussian. Why “least Gaussian”? The central limit theorem loosely states that the sum of a set of random variables tends toward a Gaussian distribution. It is the reason why many things like height and IQ, which are effectively the sum of many different factors, tend to be Gaussian distributed. For this reason, we would suspect that a random combination of independent factors would tend to be more Gaussian than the original factors themselves. In this way, finding the combinations of mixed components that are least Gaussian tends pull out the original sources if they can be found by a linear combination.

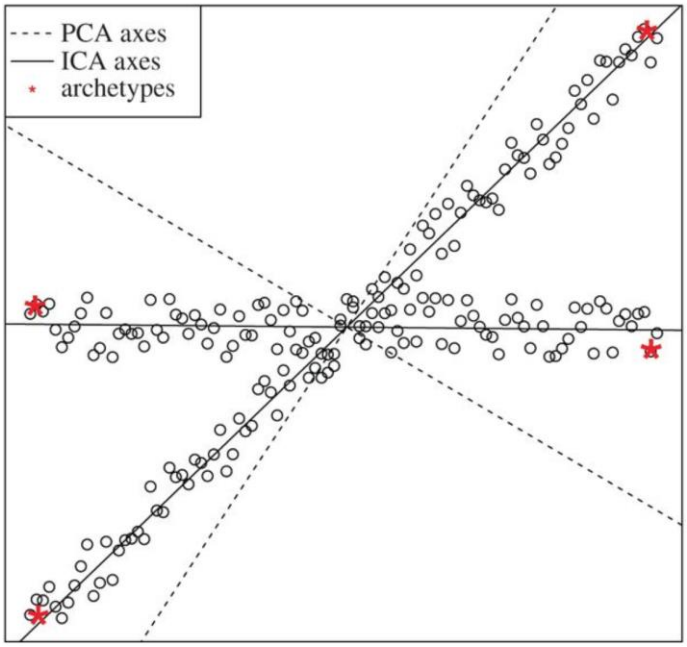

Figure 6. A Data Distribution Noting Advantage Of ICA Over PCA. Visually, we have sense of what components are more reasonable, and it appears ICA is best at determining those components. The dotted line indicates the PCA components, and the solid line denotes the ICA components which match our geometric intuitions about the underlying structure of the signal (Elze, Pasquale, Shen, Chen, Wiggs, Bex & Soc, 2014).

ICA can solve problems that PCA or other unsupervised learning strategies cannot; however, it has two limitations which make wider acceptance more difficult than more straightforward algorithms like PCA. First, in ICA there is not intrinsic ordering of the components indicating which are more or least important to encoding the information - unfortunately this means that if you are trying to isolate a source, you have no prior expectation of which component is which. Also, due to the fact that component independence is the goal, there is no guidance in the algorithm for what the amplitude of the original signal should be. These are mathematical limitations of the algorithm that make it difficult to explain and validate for easy demonstration and understanding (Hyvärinen & Oja, 2000).

Interestingly, both PCA and ICA can be described and applied without any reference to how the brain processes information; however, as we will see, the encoding objective used has a dramatic impact on the resulting code - one which can tell us about the brain, and another which, though useful, has very little to do with the goals of neural processing.

Sensory Neuroscience

Now that we have explored the necessary background in signal processing, it is important to understand the visual and auditory stages of processing that relate to the efficient coding approach.

Sensory neuroscience attempts to understand how sensory information is encoded in the brain. The sensory systems include visual, auditory, somatosensory, gustatory, olfactory, and vestibular. Only some of these senses encode input using a space that is easily identified - visual encoding uses a 2D map of visual space, auditory uses a 1D tonotopic map of frequencies at the lowest level, and somatosensory is encoded based on parts of the body. The area of the sensory space in which stimuli can affect the response of an individual sensory neuron is referred to as the neuron's receptive field - particular parts of an image for visual neurons, particular frequencies for auditory neurons, or particular parts of the body for somatosensory.

When a stimulus falls within a neuron's receptive field, the neuron will become activated. During this activation, the neuron fires action potentials. Action potentials are the primary means of transmitting information between neurons. In its simplest form, the neural code can be represented by the firing rate of these action potentials - higher activity means more action potentials are generated per second. This is referred to as a rate-code. Although it is well understood that neural coding is not limited to rate coding (e.g. sometimes the precise

timing of spikes matters), much of early sensory neural response can be understood as a rate encoding of information.

Visual System

Visual information is first transduced by photoreceptors in the retina, where layers of interconnected neurons in the eye then process the information. These neural signals in the eye eventually reach the retinal ganglion cells whose axons make up the optic nerve transmitting the information from the eye as a series of action potentials. On a related note, estimates of the information transmission rate of optic nerve are approximately 1 megabyte per second (Reilly, 2006). The lateral geniculate nucleus (LGN) receives action potentials directly from retinal ganglion cells and from this point the information is relayed to the primary visual cortex.

Primary visual cortex, also known as V1, is the earliest cortical visual area in the brain. Because of its relatively low level of sensory signal processing, the way V1 encodes information is fairly well understood, with a series of models that range from straightforward and incomplete, to very complicated but capturing much of the variation in neural response. In this area, there is a classic dichotomy of cell types referred to either as either “simple” or “complex” (Hubel & Wiesel, 1962), with an understanding now that the dichotomy is helpful, but not completely accurate. So-called simple cells are the primary cell type we are concerned with here - the receptive fields can be described by linear filters. Complex cells in V1 are a type of cell that requires a nonlinear receptive field description that is beyond the scope of this thesis.

One the one hand, there are approaches characterizing simple cell neural receptive fields in V1 as tuned to detect oriented lines and edges, and their behaviors tend to represent a linear code (Movshon, Thompson & Tolhurst, 1978). Others have used sinusoidal gratings

(planar sine waves of bright and dark lines) and observed that simple cell receptive fields are responsive to the position, orientation, spatial frequency, and phase of the gratings used to generate a response. However, these parametric stimuli do not lead to a generalized description of the neural response.

A more complete picture of V1 neural response can be obtained by observing the pattern of bright and dark in an image that a cell is responsive to. Interestingly, the pattern for simple cells in V1 strongly resembles a 2D Gabor filter (Figure 7). Researchers have parametrically fit Gabor filters to these patterns finding the appropriate ranges for parameters in the 2D Gabor filter model to match what is observed in primate primary visual cortex (Jones & Palmer, 1987; Lee, 1996).

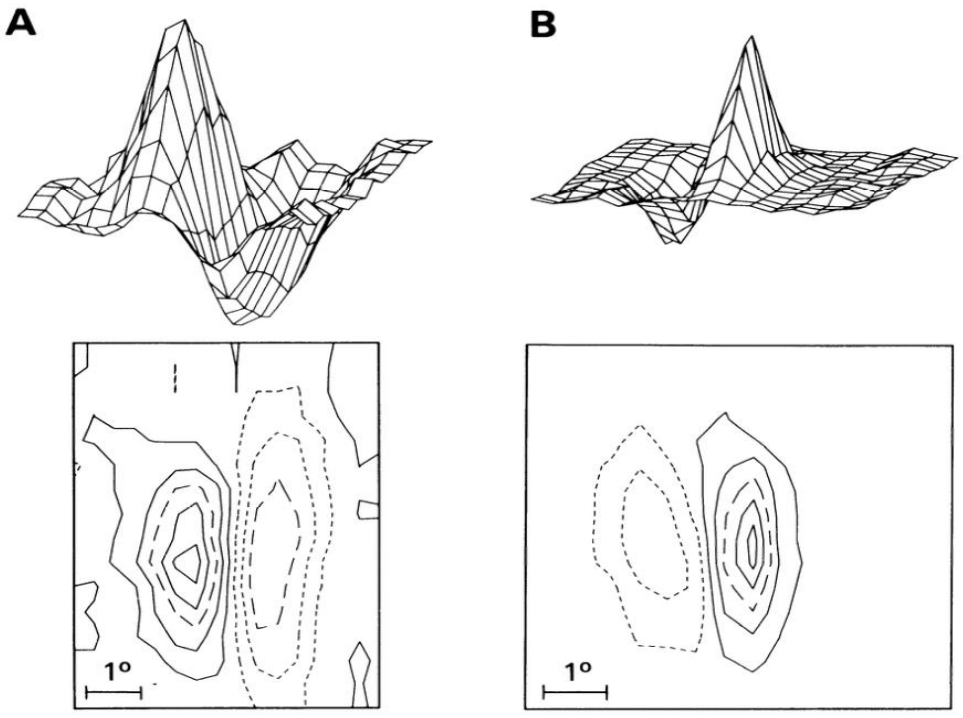

Figure 7. Simple Cell Receptive Field Model From Cat Cortex. Mesh plots on the top correspond to contour plots below. Note the strong resemblance to 2D Gabor filters (Jones & Palmer, 1987).

Later visual stages process the information further. For example, the lower, ventral visual stream transforms the visual signal into a series of more complex features for object identification while the upper, dorsal visual stream processes visual information related to spatial processing. These later stages of processing have been modeled with varying degrees of success. Clearly, to go beyond simple cells in V1 a more complex modeling objective is necessary in part because later stages of visual processing are certainly not linear transformations of the signal. These later stages of processing are beyond the scope of this thesis, but it should be noted that the computational approach applied in this thesis has successfully addressed many other more complex levels of neural processing.

Auditory System

The auditory system also takes an external signal - in this case variations in air pressure over time - and transduces it into neural impulses for further processing. When a sound is heard, the sound wave passes through the ear canal and eventually reaches the cochlea. The physical structure of the cochlea causes sounds at different frequencies to vibrate differently along the spiral structure of the cochlea - performing a crude Fourier-like transformation of the signal to vibrations in different locations. Hair cells are present along the basilar membrane which, when vibrating, cause a neural response – analogous to how photoreceptor cells in the visual system convert light energy to neural responses. After some minor processing in the cochlea, the signal reaches the neurons of the spiral ganglia. The axons of these cells are what make up the auditory nerve.

The receptive fields of neurons on the spiral ganglia, which make up the fibers of the auditory nerve from the cochlea, have been characterized is a very similar way to how simple

cells have been characterized. Whereas visual receptive fields are represented as patterns of light and dark over a 2D image, auditory receptive fields are represented as patterns of high and low air pressure over time. When particular patterns of sounds are presented to an animal, and the neural response is recorded, this can be used to infer the patterns of high and low pressure that make a neuron in the spiral ganglia fire. From this, the receptive field can be plotted (Figure 8).

When the receptive fields of spiral ganglia cells are observed, there is a similar, characteristic structure not unlike 1D Gabor filters. However, the more appropriate parameterization of these filters is what is called a gammatone filter - sine waves in a gamma function envelop. These gammatone filters have similar properties in that they produce responses that are localized simultaneously in frequency and time.

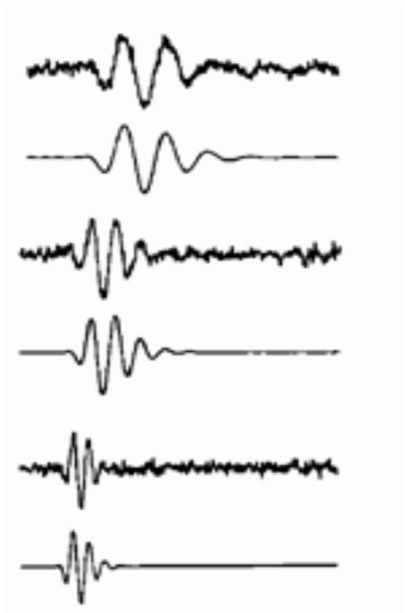

Figure 8. Receptive Fields Of Auditory Nerve Fibers. In each pair of signals, the upper ones are the filters measured from the cochlea, whereas the lower ones are modeled gammatone filters (Lewicki, 2002). Temporal extent of the filters is 20ms and peak frequency response is 364, 642, and 999 Hz, top to bottom.

Intersection Of Sensory Neuroscience And Efficient Coding

Now that we have introduced the prerequisite knowledge of both signal processing coding strategies and sensory neuroscience, it is time to make it clear how these concepts relate to one another in a powerful way.

It's important to observe how an efficient code can be significant from an ecological point of view. The efficiency of the neural code representation should increase the evolutionary fitness of the animal. A code that causes neurons to be less active overall can reduce metabolic costs, which can have a dramatic effect on evolutionary fitness. Additionally, coding strategies which lead to faster processing for later processing goals, like identifying predators, prey, or mates, would also lead to better evolutionary fitness. It is also important to note that at this stage of processing, the goals across animals tend to be quite similar, meaning that this stage of encoding is likely to be highly conserved and optimized over evolutionary time. In essence, the right efficient coding strategies are essentially mathematical expressions of ecological goals of the animals at this stage of processing.

We observed earlier that V1 simple cell receptive fields could be modeled as 2D Gabor filters. Similarly, the receptive fields of auditory nerve fibers can be modeled as gammatone filters. But why of all possible receptive fields do we see these filters? Simply put, it is because these are the best filters in terms of the efficient, ecological coding strategies.

Of course, this ecological description can be reversed to derive these receptive fields from scratch. First, what does the animal have to efficiently encode - natural images and a range of sounds that animals experience over evolutionary time. If you collect such images and sounds - again, emphasizing "natural" - and use an efficient coding strategy that is appropriate

for the given stage of processing being modeled, the result code should resemble the coding strategy seen by receptive fields in the brain.

This process is discussed in more detail in the results section, but a succinct visual description of this process is available in the following figure, which was the preliminary work for this thesis performed by Mary Makarious and presented as part of her poster in the Society for Neuroscience conference in 2015.

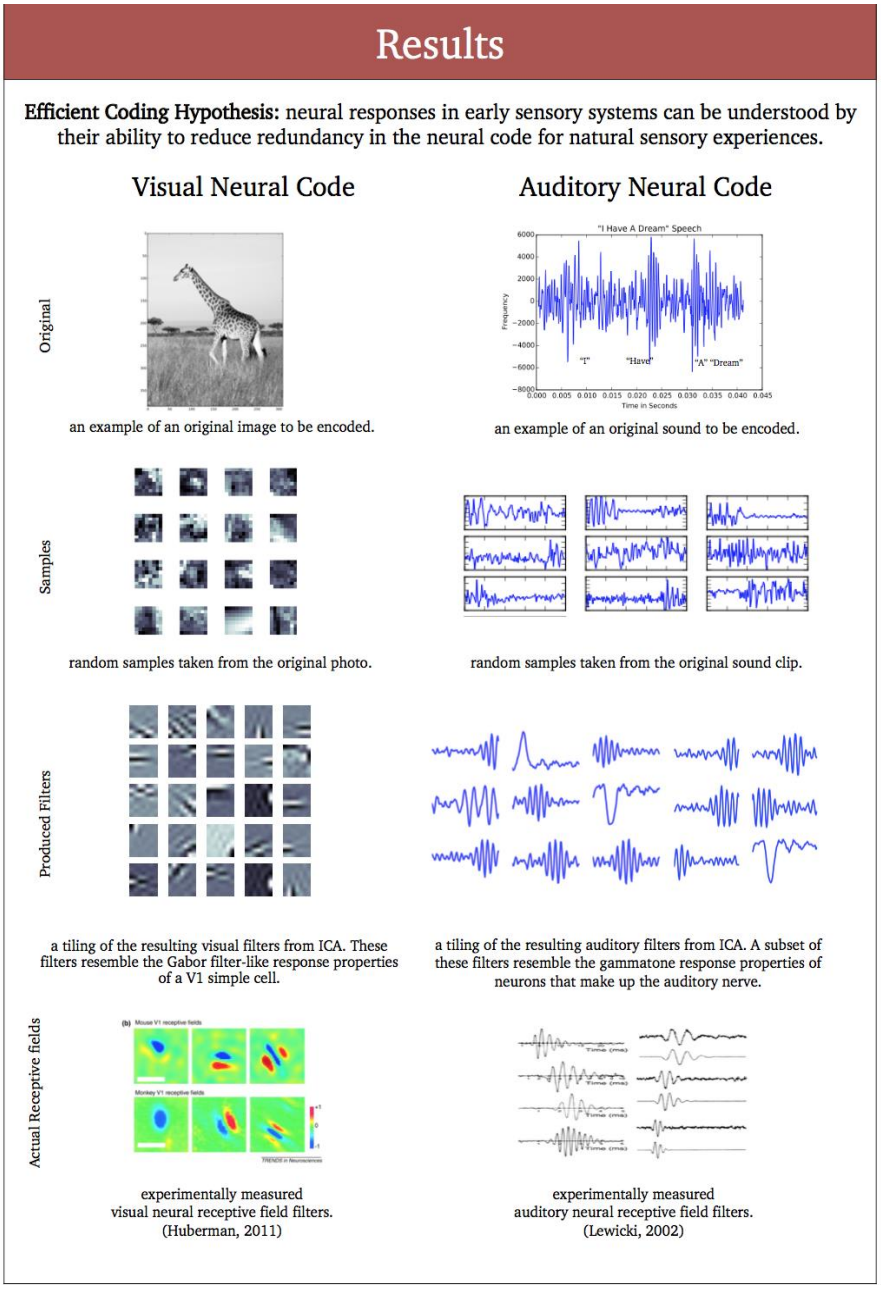

Figure 9. ICA Applied To Images And Sounds. The rows are stages of processing natural images and sounds. The left column is for images, and the right column the equivalent step for sounds. (1st row) We collect natural images or "natural sounds" for encoding. (2nd row) We sample those images into a large number of image patches (or small sound clips). (3rd row) ICA is the efficient coding algorithm applied to this data. Since ICA is an objective that is appropriate for neural coding, the resulting filters in this row appear similar to (4th row) the receptive fields as measured directly from simple cells in primary visual cortex and receptive fields of the auditory nerve fibers. This figure is extracted from the following poster presentation describing the project design goals: (Makarious, Moe & Albert, 2015).

Our efficient coding app is, in essence, a demonstration of figure 9 made more palatable for a casual user. We will take a deeper look into the steps of this approach in the following section.

CHAPTER III

RESULTS

The Purpose Of The App

It is surprising to learn from the previous sections that in computational neuroscience, visual and auditory information is processed using the same computational strategy. However, to most people, even to many neuroscientists, the early visual system and auditory system are still two completely distinct systems that are treated differently. An understanding of the efficient coding approach to neuroscience can be beneficial, but there are impediments to spreading the word that need to be overcome.

To fully understand the efficient coding strategy as a bridge across these disparate systems, one often seeks a sense of what these efficient coding algorithms do by trying to understand how they work. This however is a flawed approach for most people since a thorough understanding of an algorithm being used is often not necessary; try asking someone who uses PCA or ICA how it works, and most won't have a succinct, coherent answer even if they use the output of these algorithms on a regular basis. Luckily, these algorithms are already implemented so this is less of a concern for most people. An intuition for what these algorithms do in context is most important. That is, the main point at an introductory level is to make people aware that "neurally-appropriate" efficient code exists, whatever they may be. That is the approach we take with this app.

Second, even if students do not know the details of the efficient coding algorithms used to demonstrate this principle of neural processing, they still need enough familiarity with a programming environment and access to appropriate data to run the algorithms themselves. Although this process is trivial for students who have programmed before, it can be daunting for students that haven't. Again, the reason we have created this app is to remove the obstacles to students appreciating the efficient coding principle. Simply by showing that "neurally-appropriate" efficient coding strategies on natural scenes lead to V1 receptive field pictures - and other coding strategies or image selections don't - is often enough to demonstrate this idea.

We have streamlined the process of collecting appropriate images, applying an efficient coding strategy, visualizing the resulting filters, and comparing the results to those from real neurons - making it all possible with only a few clicks on our app. The complex procedures such as sampling the data, preprocessing the inputs, applying PCA or ICA, and the process of visualizing the filters are hidden from users. Anyone, as long as they have a phone, will be able to easily observe how an efficient coding of natural scenes (or sounds) produces a visual (or auditory) code like we see in the brain.

Efficient Coding Procedure

In our app, there are a number of processing steps to produce receptive field-like linear filters from the raw image or sound data that were done offline and hidden from the user. But as a reminder, despite the different input sources for the visual and auditory systems, the processing steps are equivalent. We will take image processing as the primary example to explain the methodology we use in the project.

1. Collecting the appropriate data for encoding: e.g. natural images.
2. Extracting image patches (or sound clips) from the data.
3. Reducing dimensionality with Principal Component Analysis (PCA) (optional, for speeding the convergence of ICA).
4. Apply an encoding algorithm to derive the proper coding strategy on that data - e.g. ICA.
5. Displaying the resulting linear filters produced by that code, which are equivalent to the linear receptive fields as measured in animals if the appropriate data and coding strategy are used.

The first step is to collect appropriate input for encoding. Since the brain is tuned to encode natural images and sounds, we would use the same for our app. We chose images that are taken in the natural scenes, such as animals, rocks, and plants. For sounds, we chose to record sounds heard in nature such as animal vocalizations, rain, and rivers. We also used human vocalizations, as that is an important signal to process for people. The natural images were collected via camera, and natural sounds are collected via microphone. Additionally, we chose a set of “non-natural” images and sounds to show how filters can be produced that are not similar to what is observed in the brain.

The second step is to extract samples from these images or sounds. We convert the image into grayscale, and then cut them it into small patches - for the app we used small 8 by 8 pixel patches reshaped into a 64 dimensional vector for each patch. For sounds, we extract clips of 100 samples from an original frequency of 44.1 kHz, which represents clips of approximately

2ms in length. It is important to collect many clips - perhaps hundreds of thousands, as it can be difficult to converge on the best coding strategy over so many dimensions.

Independent Component Analysis (ICA) is used to process the image patches and audio clips. The FastICA algorithm is provided by Python scikit-learn library. The patches we generated from previous step are the input samples for the algorithm. Commonly, there is a preprocessing step where the dimensionality is reduced using PCA (e.g. 8x8 pixel patches with 64 dimensions were reduced to 40) which speeds up processing and helps to remove noise. This PCA preprocessing is included in the FastICA library.

After PCA preprocessing, the input data matrix is submitted to ICA to derive a code producing a set of linear filters that have response properties that are maximally statistically independent. These linear filters represent neural receptive fields if “natural” images or sounds are used as input.

The last step is to take the linear filters we derived from ICA and reconstruct them into image patches for viewing. If PCA preprocessing was done and so the filters returned from ICA are in the reduced dimensionality, the original PCA model is used to transform the filters back into the original (e.g. 64 dimensional) space. These 64 dimensional vectors are then reshaped back into 8x8 matrices for display. This would allow us to have a direct view of the linear filters, and we can compare them visually with the receptive field images that we measured from neural responses to stimuli in the animal’s brain.

The App

We designed our app to be simple and user friendly. The welcome slides give users a walk through of basic concepts of efficient coding and what our app does, and are accessible at

any point after. The GUI itself is self explanatory in its functionality. There are two main sections in our app, one for images and one for sounds. Within these sections, we have subsections to allow users to compare different inputs and outputs:

- Natural images (which we expect to produce filters mimicking real receptive files) and non-natural images (e.g. pictures of carpet, mosaics images)
- Natural sounds including harmonic sounds (e.g. birds chirping) and anharmonic sounds (e.g. footsteps, crunching leaves) and non-natural sounds like keyboard strokes.

However, evolving and developing animals are know to process a mixture of these sounds, which, as we demonstrate in the app, lead to filters that resemble auditory receptive fields (Lewicki, 2002). Additionally, we use human speech as an input and note that resulting filters are similar to what is found in the auditory system, demonstrating a link between speech statistics and general auditory processing. By providing these examples, we demonstrate that the resulting neural code depends heavily on the statistics of the images or sounds that the animal is exposed to over evolutionary time.

Figure 10 to 17 are screenshots from the app. We will give a short walk through on the flow of the app and briefly explain the functionality of each screen.

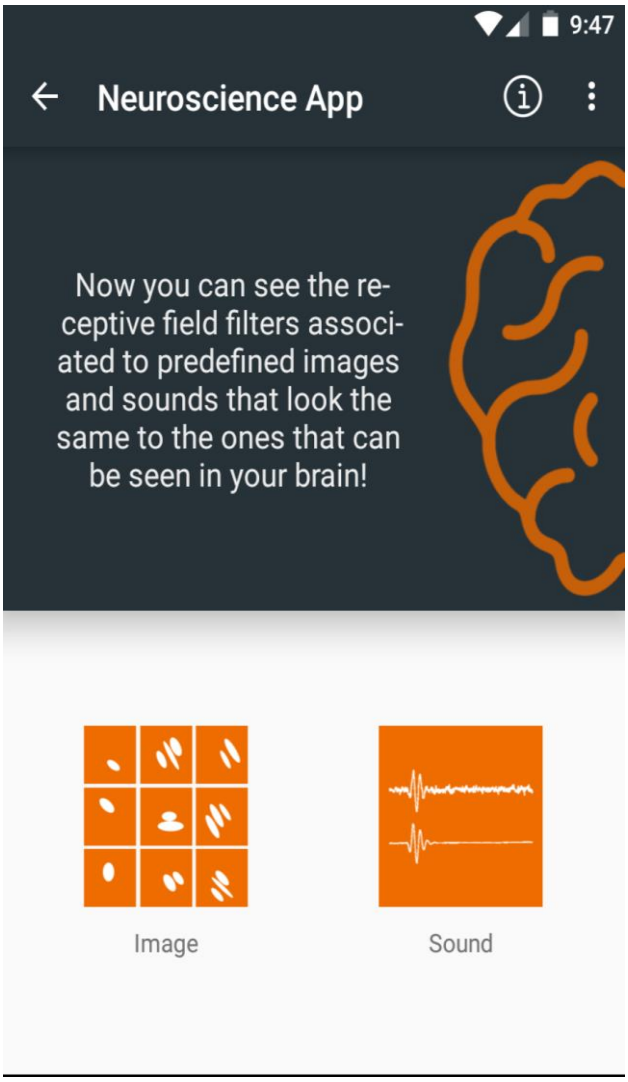

Figure 10. Main Screen Of The App. This screen will show up after the welcome slides once the app starts up. The main screen gives a brief introduction of the functionality of the app. On the bottom are two buttons for the users to choose whether they would like to see examples of efficient coding for images or sounds.

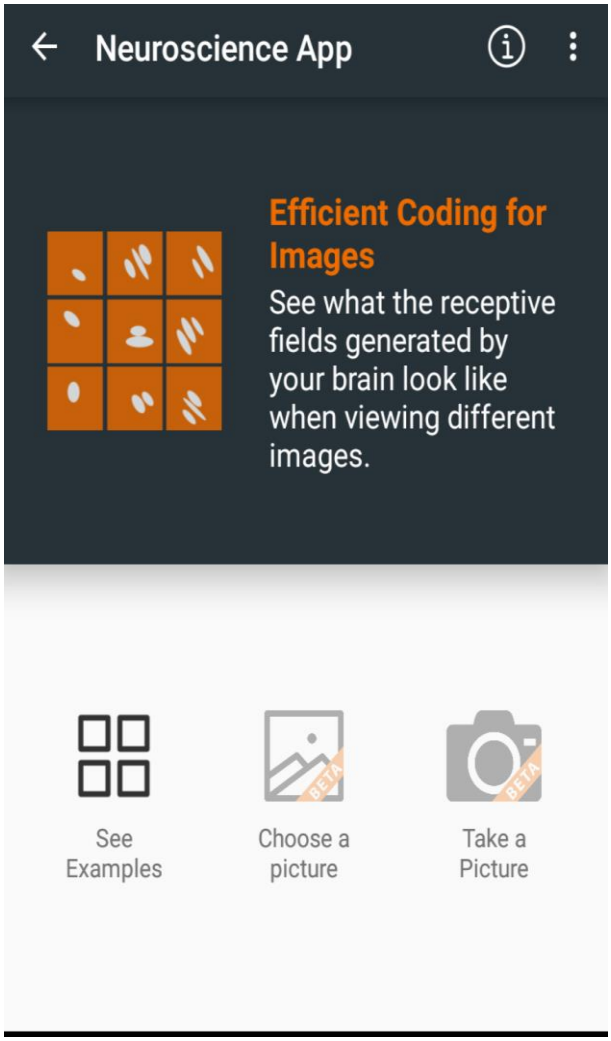

Figure 11. Image Processing Main Screen. When user chooses to see examples of image processing, this is the screen that will show up. It is available to users after they select the image button from the main screen shown in figure 10. Here we provide a link to see the preprocessed examples from the python code. We also left options for real time efficient coding demonstrations; users can use photos from their phone memories or take a picture. These functions will be implemented at a later time.

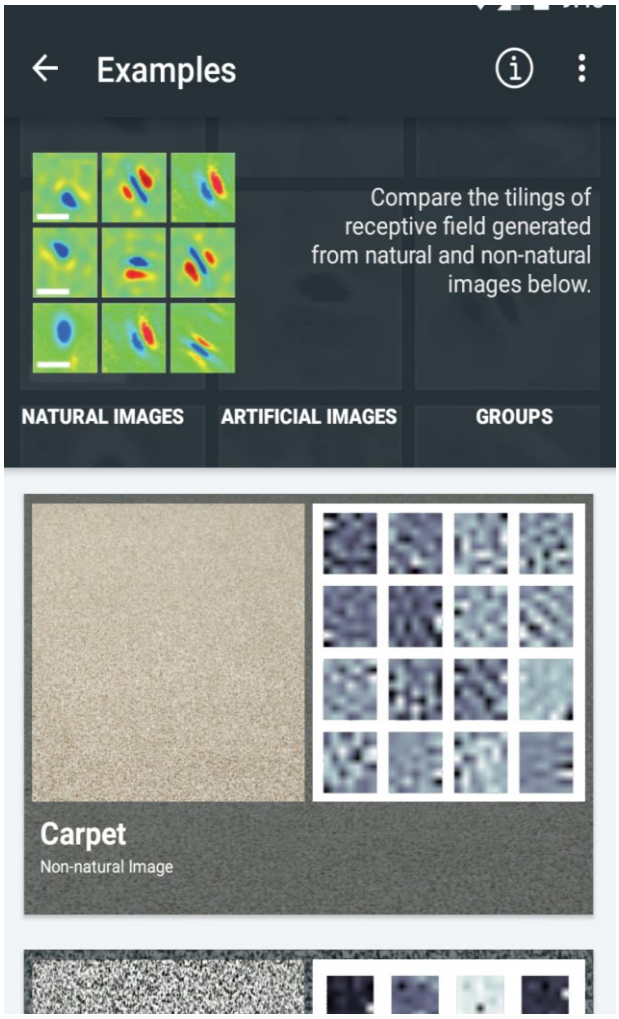

Figure 12. Image Example List. This screen will show up after the user clicks on the “see examples” buttons for images in the previous screen. We grouped these images into two categories for comparison purposes, the natural image group and the artificial image group provide distinct inputs for our purposes and further illustrate the importance of using natural images as input.

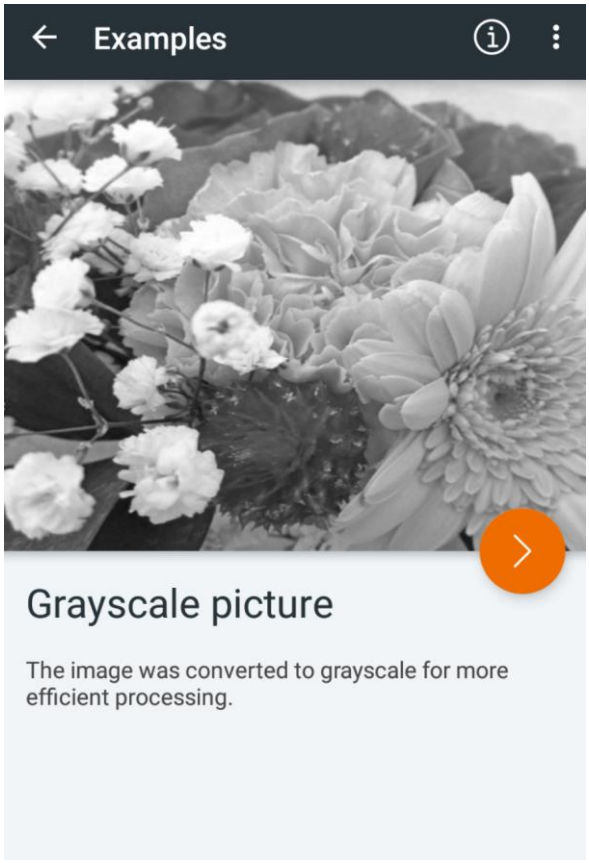

Figure 13. Display Of An Original Image In Grayscale. For each example image in our app, we provide a thumbnail of the original image for quick visual comparison. The user has the option of tapping on the image for a larger version.

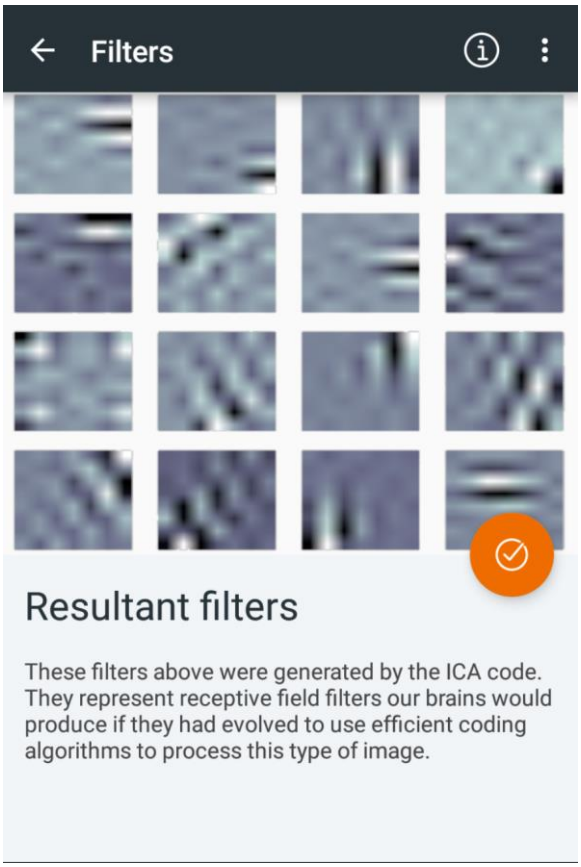

Figure 14. Display Of The Resulting ICA Filters For Images. We randomly selected 16 linear filters from the ICA result dataset and display the filters as a result. These filters do not come in any specific order, and each of them is equivalent to a representation of a receptive field in V1.

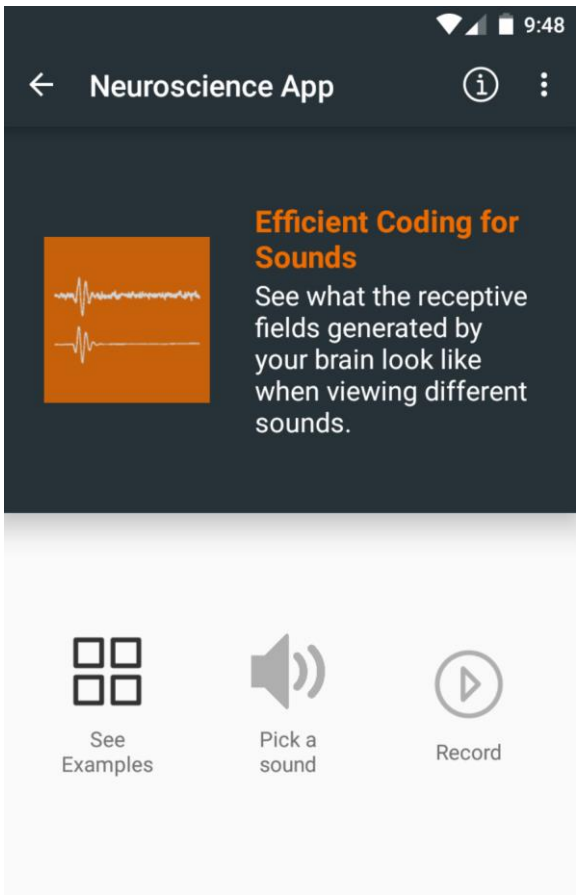

Figure 15. Sound Processing Main Screen. Similar to image processing screen as we introduced above, this screen will be shown when the user is interested in efficient coding for sounds. The user will be able to compare the encoded sound clips to their original sound sources by clicking on the “see examples” button. Future improvements will allow users to pick a custom sound from their phone memory or simply record one to apply to the efficient coding process.

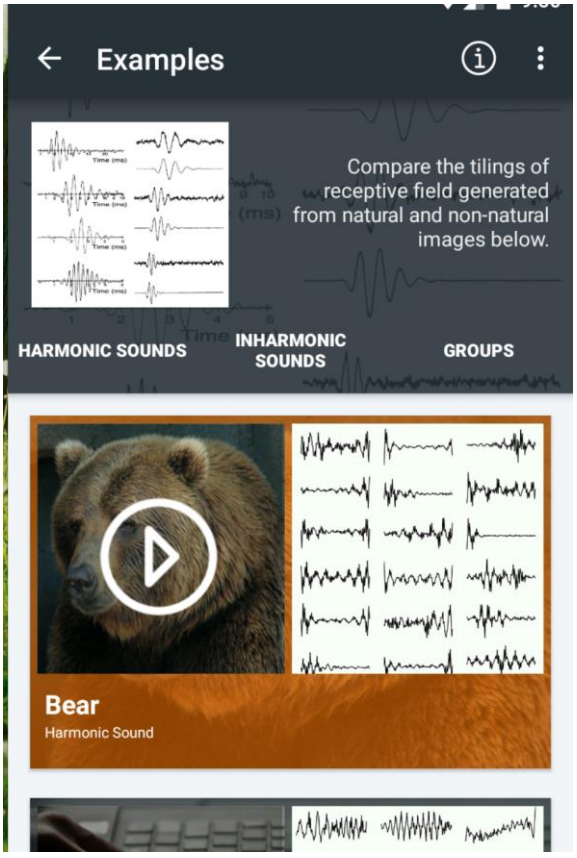

Figure 16. A List Of Efficient Coding Examples For Different Sounds. This screen will show up after the user clicks on the “see examples” buttons for sounds in the previous screen. Similar to image examples, we also grouped these sounds into four categories. Users will be able to compare the difference in resulting filters between harmonic sounds, anharmonic sounds, a mixture of the two, and human speech. For each sound clip, user can click an image representing that sound to access details of the sound clip.

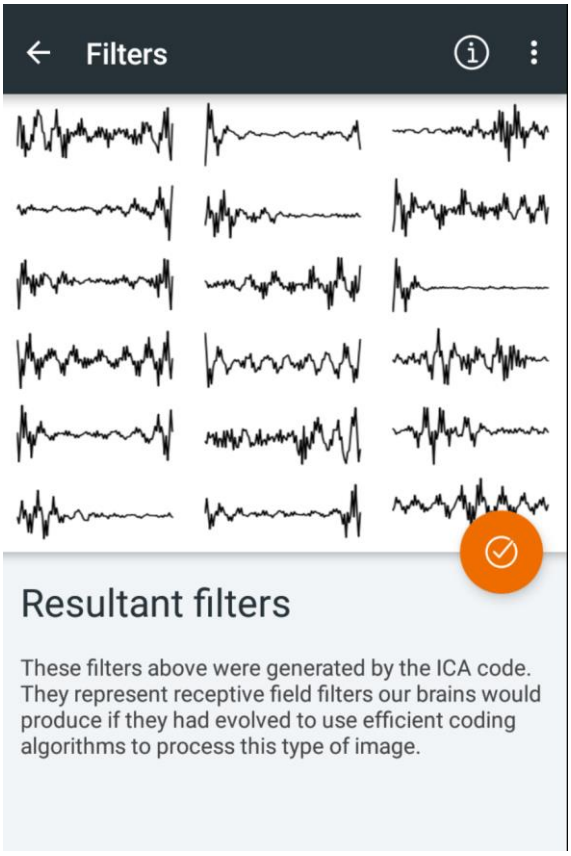

Figure 17. Display Of The Resulting ICA Filters For Sound. Similar to the results for images, we randomly selected 18 linear filters from the resulting ICA filter set for display which resemble the receptive field filters of the auditory nerve.

Future Improvements

We tested our app with Android mobile devices. The app works robustly in the current version accomplishing the main goal of the project, but there are two parts that were not added to due logistical constraints.

In a future version, the app will be able to run the ICA algorithms directly in the app (as opposed to using images produced and processed outside the app). This will be doing using images from the built-in camera, allowing the user to see the effect various images have on the resulting filters. Similarly, the same will be done with collected sounds using the microphone. This functionality would take full advantage of the existence of this app on a mobile device with a connected camera and microphone.

Second, although this app shows the variations in the resulting code by changing the input images and sounds, it is not clear to a user how important the choice of efficient coding strategy is. A naive user may get the impression that any efficient coding strategy works. To counter is, a future version would show the filters from other coding strategies. One obvious example would be to show PCA filters for this purpose, which are exceptionally non-neural, although it is usually a coding strategy more people would envision when hearing the words “efficient coding”.

Finally, the resulting filters were generated using a relatively small set of images, image patches, and sample numbers out of convenience. This leads to resulting filters that may deviate farther from observed neural receptive fields than necessary. Given more time, a large set of images, larger image patches, more samples, and more running time would be applied to

produce filters which are much more similar in appearance to the receptive fields measured in the brain - as has been observed in research papers using these techniques.

CHAPTER IV

DISCUSSION

Our app only shows a limited application of the efficient coding principle to understand sensory neuroscience, but so much more is possible. The exact same strategy used here on grayscale images works equally well to understand color processing, binocular image processing, and temporal encoding of image sequences. Additionally, this approach can be applied to other sensory processing, and even motor processing if the process is reversed (e.g. "muscle synergies" for typical movements instead of sensory receptive fields for natural sensory information, but the sample principle applies). In this section, we will discuss some of the common questions that people might have, and take a moment to appreciate the many other straightforward applications of this efficient coding approach.

Why Not Model With Neurons To Understand Neural Processing

In traditional neuroscience, when studying the neural code, scientists primarily want to understand the nature of the response of a neuron to sensory stimuli. What does the neuron respond to? What make it fire strongest? What is it responsive to or not responsive to? Of primary importance is how is the information encoded. The exact details of how the system performs the necessary steps to encode that information is often secondary to an overall understanding of what the system is doing.

It's true that many computational neuroscientists do use neural network models to understand processing in the brain, and that is a noble pursuit. However, it is a level of understanding a system that is distinct from the computational level of understanding – a statement that can be confusing since many "algorithmic level" models are referred to as computational models in other contexts. In Marr's levels of analysis, a neural model may help us to understand a specific approach to how the brain can process the information in a neurally plausible way, but it can sometimes the insights from such models can be difficult to generalize to other domains. Imagine if we have a perfect neural model of how the brain works; we will still be far from a compact description of the computational goals of the brain at various stages. A similar analogy can be made of processing in a CPU. If we recreated a network of transistors to perform a computation of a CPU, we still may not have a high-level understanding of the role of each part of a processor.

This is why we choose to use a computational level of analysis to understand sensory processing here. That level of abstraction allows us to have one simplified model for images and sounds, and all we need are different inputs to get the corresponding results. Furthermore, this model could be applied to other types of visual input and other sensory systems in a straightforward way (only a change in inputs!). In fact, a computation level of understanding makes our inferences immediately relatable to application. ICA was in fact invented to solve an applied engineering problem rather than to study how the brain works, and by focusing on this higher level of understanding, the same tools can also be used across domains of traditional computer science and engineering and neuroscience.

How Does This Efficient Coding Technique Apply To Richer Visual Information

We have demonstrated the efficient coding of both images and sounds in our project. For sounds, our inputs are exactly the same as what our ears actually receive from the environment. However, for images, we chose to convert the images to grayscale first before we start the encoding process. The reason for using grayscale images is that the resulting filters are more consistent - there is no variation in color of filters - which makes the explanation easier. But how would this method work if we used natural images in color?

The answer is that it would create a code quite similar to what we observe for color-coding in primary visual cortex - again simply by changing the inputs! To represent colored images, instead of using one number for each pixel, which is the intensity of grayscale image, 3 numbers will be used to include the color information. Hoyer and Hyvärinen have performed this experiment to demonstrate that the efficient coding strategy can be applied to colored image in a straightforward way (Figure 18) (Hoyer & Hyvärinen, 2000). It is interesting to note that red/green and blue/yellow opponency occurs which is similar to the color opponency seen in V1 color selective cells. Also we can note that most filters still primarily focus on luminance - again something observed in the visual system of most animals that process color. Finally, we can see that the color selective cells are generally of lower spatial frequency, which was one of the first observations noted of color selective cells in primary visual cortex.

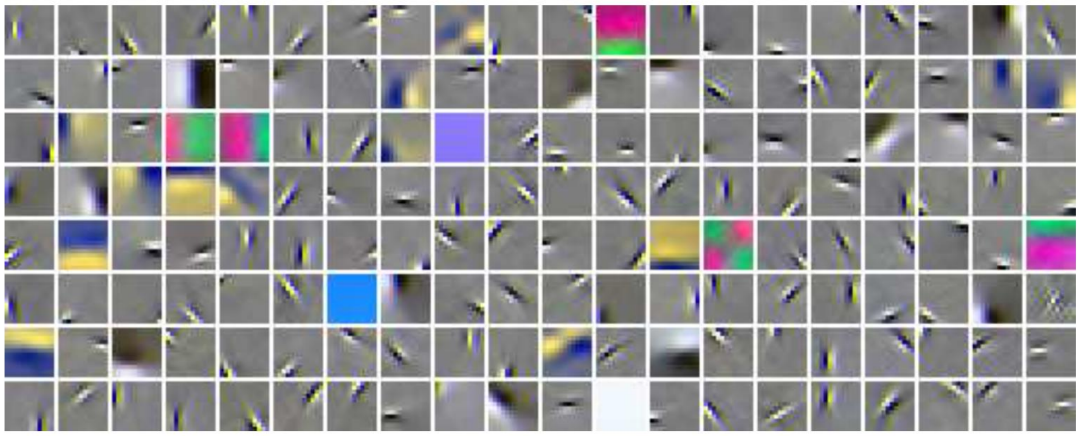

Figure 18. Receptive Field Linear Filters Derived From Colored Natural Images. A set of linear filters of colored image, produced by Hoyer and Hyvärinen (Hoyer & Hyvärinen, 2000).

The same process can be performed to observe what one would expect for binocular receptive fields. For binocular images, there will be two images as input instead of one - separated by a fixed distance (e.g. distance between the eyes). This means a single sample will contain two patches - one corresponding to the left eye and the other for the right eye.

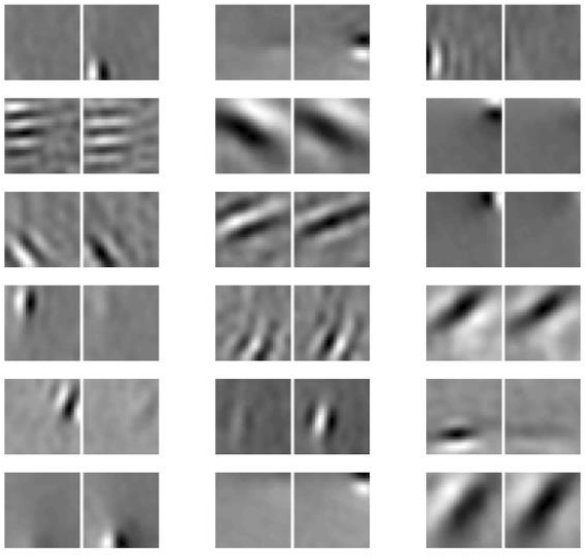

Figure 19. Receptive Field Linear Filters Derived From Binocular Images. (Hunt, Dayan & Goodhill, 2013).

When it comes to videos, again the same process can be applied but instead of using static 2D image patches, the samples will be 3D tensors - a 2D image patch over time for the third dimension. Hateren and Ruderman have applied ICA to a series of video clips during natural vision and observed that ICA produces spatiotemporal filters that resemble the measured spatiotemporal filters from simple cells of primary visual cortex (van Hateren & Ruderman, 1998).

Figure 20. A Receptive Field Spatiotemporal Linear Filter From Natural Video. The receptive field filters of 12 consecutive time-frames each of 12 x 12 spatial pixels (van Hateren & Ruderman, 1998).

All of these examples use ICA - the same algorithm - but only change the nature of the input to demonstrate properties of color, stereo, and temporal visual coding.

How Can Visual And Auditory Information Be Processed The Same Way In Two Different Sensory Systems

Clearly, sounds and images are processed in two different sensory systems in disparate parts of the brain. In fact, the auditory filter equivalents to ICA on natural images take place in the cochlea for the auditory system, which is very early at a pre-cortical processing stage, and in the visual system the equivalent filters occur at the first stage of cortical processing after a great deal of processing and transmission in the visual system. It's true these neurons are quite distinct in terms of stage and complexity of processing. However, from the computational point of view - however and wherever the processing is done - the key point is that the processes of evolution and adaptation eventually reach a point where the same computation is performed in

both locations. Noting that both systems perform the same coding strategy does not imply they should be located in similar locations in the brain. In fact, one could reason why they are different. An independent code is important for later processing, but in a complex, rich signal coming in from the retina, a more critical first stage of processing would be one that might help to compress the signal for transmission to the cortex. Only when the signal reaches cortex would it be appropriate to be concerned about independent codes rather than compact codes. For example, in V1 there are roughly 100 times as many neurons as there are axons in the optic nerve, minimizing the need for compression and freeing the system for more important objectives, like sparse or independent coding. Conversely, efficient coding of auditory signals is greatly simplified by the physical transduction process that occurs in the cochlea, which essentially performs a Fourier decomposition of incoming sounds along the cochlea using acoustics. The processing steps after that to obtain gammatone filters would be exceptionally straightforward, and these can all be done directly in the cochlea.

Note, we emphasize the visual and auditory systems here, but other sensory systems can be modeled by efficient coding too. For example, efficient coding theory also applies to understand aspects of the somatosensory system (Hafner, Fend, Konig & Kording, 2004).

What Alternatives Are There To ICA For A Neurally Appropriate Efficient Coding Strategy

ICA performs well as a candidate coding strategy demonstrating the principle of efficient coding in sensory neuroscience. However, it should not be considered the definitive coding strategy for these areas - just one of many variations that can explain these response properties. Notably, it is not the only coding strategy out there that can produce filters similar to the receptive fields in the brain for these areas. In this section, we will introduce some other

strategies that are often used to demonstrate efficient coding in a similar way. Though these strategies may appear distinct by definition, the mathematical and statistical relationships of these coding strategies are evident in practice; they are more similar in practice than they appear. To be clear, this thesis is not a promotion of ICA above these other strategies, but rather noting it as one of a number of potential objectives demonstrating the power of the efficient encoding approach.

One of the well-known strategies for efficient coding is sparse coding. Sparse coding delivers a complete family of localized, oriented, bandpass receptive fields much like ICA (Olshausen & Field, 1996) - in fact it was the first approach used for this purpose. While ICA intends to find the most independent components, sparse coding is trying to minimize the number of active neurons while still be able to represent the image. This is a reasonable goal of neurons, as spiking neurons are metabolically expensive. To note the practical similarities, we often observe that ICA codes tend to be sparse. Linear filters that are derived to be statistically independent are not often active in random natural scenes. Hence, ICA and sparse coding are known to be theoretically distinct but from a practical point of view given natural images, they can at times be indistinguishable. The filters provided by ICA and sparse coding basically look the same, as shown in figure 21.

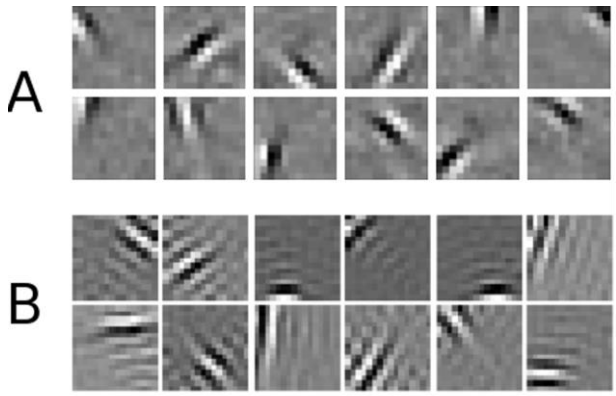

Figure 21. Filters From ICA And Sparse Coding From Natural Scenes. Group A shows receptive fields with sparse coding; group B shows receptive fields from ICA (Albert & Field, 2009).

Seemingly distinct from sparse and independent coding, another approach relies on the fact that the eventual goal of sensory coding is to be invariant to low-level features in the image that may vary quickly during natural viewing (e.g. translations, rotations, resizing, or lighting changes) but respond distinctly to high level changes, which tend to change more slowly over time (like a new person coming into view). With this reasoning, we may want to find filters that have the slowest response changes during natural viewing. This unsupervised learning of invariances strategy is called slow feature analysis. The goal of this approach is to find a code where neural activity changes slowest in time but still represents the image. Unlike ICA and sparse coding, this model requires temporal signal, such as video. Interestingly enough, this also produces 2D Gabor-like filters if you are restricted to a linear code; however, it is notable that this approach applies to codes that are linear or nonlinear.

It is important to note that these efficient coding strategies are likely not appropriate to go far beyond primarily visual cortex - at some point it is necessary to include additional ecological objectives in an encoding model, like object identification (for ventral visual

processing) or interacting in a 3D environment (for dorsal visual processing). But this thesis is confined to primarily visual cortex and the auditory nerve; therefore, we will not introduce the more complex sensory system models.

Conclusion

The efficient encoding approach is powerful. With only a statement of the goal of the sensory system in the encoding, and some math to define it and implement it, we are able to model many aspects of sensory processing in the brain. This is true for the visual system with grayscale images, but also applies to richer visual inputs in a rather trivial way (color, stereo, video by only changing the inputs, but using the same coding strategy). And, although we have demonstrated the computational similarity between visual and auditory processing, others have used the same approach for understanding somatosensory and, arguably, motor encoding as well. Our app demonstrates the relationship between the neurally relevant (e.g. ICA) efficient coding of images (or sounds) and early sensory neural coding in an intuitive, accessible way. This enables budding neuroscientists, and even the general public, to appreciate how an understanding of computational tools can bridge neuroscience research across areas of the brain, supporting a more parsimonious view of neural processing, and encouraging neuroscience students to improve their computational skills in order to meet the challenge to fully appreciate how the brain processes information.

REFERENCES

Abdi, H. and Williams, L. J. (2010), Principal component analysis. WIREs Comp Stat, 2: 433–459. doi:10.1002/wics.101

Albert M V., Schnabel A, Field DJ, Cheng H, Shatz C. Innate Visual Learning through Spontaneous Activity Patterns. Sporns O, ed. *PLoS Comput Biol*. 2008;4(8):e1000137. doi:10.1371/journal.pcbi.1000137.

Atick JJ, Redlich AN. Mathematical model of the simple cells in the visual cortex. *Biol Cybern*. 1990;63(2):99-109. doi:10.1007/BF00203031.

Barlow H. Possible principles underlying the transformations of sensory messages. *Sens Commun*. 1961;6(2):57-58. doi:10.7551/mitpress/9780262518420.003.0013.

Blauert J. Spatial Hearing- The Psychophysics of Human of sound. *J Acoust Soc Am*. 1985;77(1):334-335. doi:doi:http://dx.doi.org/10.1121/1.392109.

Churchland, Patricia S., Christof Koch, and Terrence J. Sejnowski. What is computational neuroscience?. *Computational neuroscience. MIT Press*, 1993.

David F. Efficient coding of binocular spontaneous activity for innate learning in V1 development. *Front Syst Neurosci*. 2009;3. doi:10.3389/conf.neuro.06.2009.03.302.

Elze T, Pasquale LR, Shen LQ, Chen TC, Wiggs JL, Bex PJ. Patterns of functional vision loss in glaucoma determined with archetypal analysis. *J R Soc Interface*. 2015;12(103):1-13. doi:10.1098/rsif.2014.1118.

Hafner VV, Fend M, K¨onig P, K¨ording KP. Predicting Properties of the Rat Somatosensory System by Sparse Coding. *Neural Inf Process -Letters Rev*. 2004;4(1). https://adapt.informatik.hu-berlin.de/pub/papers/hafner-niplr04.pdf. Accessed April 18, 2017.

Hohman F. Principal Component Analysis (PCA). http://fredhohman.com/assets/image_compression.pdf. Accessed April 17, 2017.

Hoyer PO, Hyvärinen A. Independent component analysis applied to feature extraction from colour and stereo images. *Network*. 2000;11(3):191-210. doi:10.1088/0954-898X.

Hubel DH, Wiesel TN. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol*. 1962;160(1):106-154. doi:10.1113/jphysiol.1962.sp006837.

Hunt JJ, Dayan P, Goodhill GJ. Sparse Coding Can Predict Primary Visual Cortex Receptive Field Changes Induced by Abnormal Visual Input. *PLoS Comput Biol*. 2013;9(5). doi:10.1371/journal.pcbi.1003005.

Hyvärinen A, Oja E. Independent component analysis: algorithms and applications. *Neural networks*. 2000;13(4-5):411-430. doi:10.1016/S0893-6080(00)00026-5.

Jones JP, Palmer LA. The two-dimensional spatial structure of simple receptive fields in cat striate cortex. *J Neurophysiol*. 1987;58:1187-1211.

Lee TS. Image representation using 2d gabor wavelets. *IEEE Trans Pattern Anal Mach Intell*. 1996;18(10):959-971. doi:10.1109/34.541406.

Lewicki MS. Efficient coding of natural sounds. *Nat Neurosci*. 2002;5(4):356-363. doi:10.1038/nn831.

Li YH, Savvides M. An automatic iris occlusion estimation method based on high-dimensional density estimation. *IEEE Trans Pattern Anal Mach Intell*. 2013;35(4):784-796. doi:10.1109/TPAMI.2012.169.

Makarious M, Moe G, Albert MV: Interactive introduction to early visual and auditory neural coding: for use in early neuroscience instruction*. Society for Neuroscience* (SfN 2015, Oct 2015)

Marr D. Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. WH San Francisco: Freeman and Company* 1.2 (1982).

Michael Reilly. Calculating the speed of sight | New Scientist. https://www.newscientist.com/article/dn9633-calculating-the-speed-of-sight/. Published 2006. Accessed April 19, 2017.

Movshon, J. Anthony, I. D. Thompson, and D. J. Tolhurst. Receptive field organization of complex cells in the cat's striate cortex. *The Journal of physiology* 283 (1978): 79.

Olshausen BA, Field DJ. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*. 1996;381(6583):607-609. doi:10.1038/381607a0.

Prasad VSN, Domke J. Gabor Filter Visualization. *Univ Maryl*. 2005;71(c):1478-1486.

Ringnér M. What is principal component analysis? *Nat Biotechnol*. 2008;26(3):303-304. doi:10.1038/nbt0308-303.

Thompson, Daniel M. Understanding audio: getting the most out of your project or professional recording studio. *Hal Leonard Corporation*, 2005.

van Hateren JH, Ruderman DL. Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proc Biol Sci*. 1998;265(1412):2315-2320. doi:10.1098/rspb.1998.0577.

VITA

Xiaolu Zhao, also known as Anne Zhao, joined Loyola University Chicago in August 2015 to pursue a Master's degree in Computer Science. In April 2016, she received a fellowship award offered by Loyola University Chicago Computer Science Department. Later in October, she was awarded a GHC scholarship from Anita Borg Institution grant to participate in the 2016 Grace Hopper Celebration for Women in Computing held in Houston, Texas. She was also a speaker at ChiPy talk on February 2017. In April, she accepted RA position to lead the development of a web app for Marcella Niehoff School of Nursing at Loyola University Chicago. Later in May, she won the most popular project among students award at Loyola University Chicago.

THESIS APPROVAL SHEET

The thesis submitted by Xiaolu Zhao has been read and approved by the following committee:

Mark V. Albert, Ph.D., Director
Associate Professor
Loyola University Chicago

Konstantin Laufer, Ph.D.
Professor of Computer Science
Loyola University Chicago

Robert G Morrison, Ph.D.
Associate Professor, Neuroscience, Undergraduate Program Director
Loyola University Chicago

Pavan Ramkumar, Ph.D.
Research Scientist
Northwestern University

The final copies have been examined by the director of the thesis and the signature that appears below verifies the fact that any necessary changes have been incorporated and that the thesis is now given final approval by the committee with reference to content and form.

The thesis is therefore accepted in partial fulfillment of the requirements for the degree of Master of Science.

_____ _____

Date Director's Signature

THESIS APPROVAL SHEET

The thesis submitted by Xiaolu Zhao has been read and approved by the following committee:

Mark V. Albert, Ph.D., Director
Associate Professor
Loyola University Chicago

Konstantin Laufer, Ph.D.
Professor of Computer Science
Loyola University Chicago

Robert G Morrison, Ph.D.
Associate Professor, Neuroscience, Undergraduate Program Director
Loyola University Chicago

Pavan Ramkumar, Ph.D.
Research Scientist
Northwestern University

The final copies have been examined by the director of the thesis and the signature that appears below verifies the fact that any necessary changes have been incorporated and that the thesis is now given final approval by the committee with reference to content and form.

The thesis is therefore accepted in partial fulfillment of the requirements for the degree of Master of Science.

_____ _____

Date Director's Signature

LOYOLA UNIVERSITY CHICAGO

A MOBILE APP ILLUSTRATING SENSORY NEURAL CODING THROUGH AN EFFICIENT CODING OF COLLECTED IMAGES AND SOUNDS

A THESIS SUBMITTED TO

THE FACULTY OF THE GRADUATE SCHOOL

IN CANDIDACY FOR THE DEGREE OF

MASTER OF SCIENCE

PROGRAM IN COMPUTER SCIENCE

BY

XIAOLU ZHAO

CHICAGO, IL

AUGUST 2017