8-2017

Deep Learning and Transfer Learning in the Classification of EEG Signals

Jacob M. Williams
University of Nebraska - Lincoln, jakew42@gmail.com

DEEP LEARNING AND TRANSFER LEARNING

IN THE CLASSIFICATION OF EEG SIGNALS

by

Jacob M. Williams

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Ashok Samal and Professor Matthew Johnson

Lincoln, Nebraska

August, 2017

DEEP LEARNING AND TRANSFER LEARNING

IN THE CLASSIFICATION OF EEG DATA

Jacob M. Williams, M.S.

University of Nebraska, 2017

Advisors: Ashok Samal and Matthew Johnson

Deep learning is seldom used in the classification of electroencephalography (EEG) signals, despite achieving state of the art classification accuracies in other spatial and time series data. Instead, most research has continued to use manual feature extraction followed by a traditional classifier, such as SVMs or logistic regression. This is largely due to the low number of samples per experiment, high-dimensional nature of the data, and the difficulty in finding appropriate deep learning architectures for classification of EEG signals. In this thesis, several deep learning architectures are compared to traditional techniques for the classification of visually evoked EEG signals. We found that deep learning architectures using long short-term memory units (LSTMs) outperform traditional methods, while small convolutional architectures performed comparably to traditional methods. We also explored the use of transfer learning by training across multiple subjects and refining on a particular subject. This form of transfer learning further improved the classification accuracy of the deep learning models.

Grants

This thesis is partially funded by the following NSF grant:

CMMI 1538059
Biosensor Data Fusion for Real-time Monitoring of Global Neurophysiological Function.

Acknowledgements

First and foremost, I'd like to thank Dr. Ashok Samal. Without his patience and understanding over the years, I would not have made it this far. His mentorship has helped me not only succeed academically, but has also taught me much in my personal life. I have been very lucky to have him as my advisor. I would also like to thank my co-adviser Dr. Matthew Johnson, who has provided more hands on help and guidance than I could have ever asked for, all with a great sense of humor.

I would also like to thank the other members of my committee. Dr. Prahalad Rao took a risk in taking me on as a research assistant from outside his department, and has been a strong driving force in keeping our research focused. Dr. Stephen Scott has been a mentor since my undergraduate days, taught many excellent classes, and introduced me to machine learning research.

I would also like to thank my parents, Debbie and Gary, for their understanding, love, and support. I'm proud to have such wonderful people not only be my parents, but also friends. My brother, Jeremy, has always had my back, and been willing to listen to any and all rants I could dish out. Finally, I'd like to think my friends and fellow students Tony Schneider, Taylor Spangler, and Dan Guitterez and everyone else I haven't had the space to name individually.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Description and Motivation

Brain research has been highlighted as an area of national interest in recent years. In 2013, President Obama launched the BRAIN research initiative, seeking to do for neuroscience what the human genome project did for genomics (Tripp & Grueber, 2011). This research has the potential to impact many important areas, from the detection, treatment, and increased understanding of diseases such as Alzheimer's and epilepsy, to the neural control of devices to aid the handicapped, to a greater understanding of how the human brain functions on a basic level.

The classification of brain signals recorded by imaging devices using machine learning approaches is a very powerful tool in many of these areas of research. For example, machine learning techniques show promise in the early detection of Alzheimer's or giving warning before an epileptic seizure. These techniques are already being used in devices such as the P300 speller (Guan et al., 2004) to provide a communication device for the severely handicapped.

In addition, neuroscience problems present a unique set of challenges that require innovation in machine learning. The data obtained from brain activity monitoring devices are noisy, have high dimensionality, and are costly to collect, which limits the number of data samples that can be collected. The combination of these factors leads to exceedingly complex data, which are difficult to analyze or classify, even using the most sophisticated and modern techniques.

This thesis presents several novel results in the broad area of brain signal classification. First, it provides a comparative evaluation of standard machine learning and data preprocessing techniques in brain signal classification. Second, the use of deep learning techniques for brain signal classification is explored in detail. While these techniques are state of the art in many other applications of machine learning, there are relatively few published results of their use in brain signal classification. In particular, recurrent neural networks, which have proven to be powerful in time series analysis, and convolutional networks, which are remarkably efficient in image and video classification, are explored in this thesis (Prasad 2014; Simonyon 2014). Third, to address the relatively low number of samples able to be collected in neuroimaging tasks, a novel application of transfer learning within the dataset is explored.

1.2 Applications

The classification of brain signals is a growing area of research, with emerging applications in both applied and theoretical neuroscience. These applications can generally be divided into a few main areas including device control, brain state detection, medical diagnosis, and basic research.

1.2.1 Basic Research

In neuroscience, the use of machine learning techniques to classify brain signals is seen as a form of multivariate pattern analysis (MVPA). MVPA is used to examine phenomena that are difficult to measure with traditional techniques. For example, support vector machines (SVMs) were used to examine to "what extent item-specific information about complex natural scenes is represented in several category-selective areas of human extrastriate visual cortex during visual perception and visual mental imagery" (Johnson, McCarthy, Muller, Brudner, & Johnson, 2015). Other examples of basic research involving the classification of brain signals include topics such as affect recognition, semantic language representation in bilingual speakers, and exploring individual differences in pain tolerance (AlZoubi, Calvo, & Stevens, 2009; Correia, Jansma, Hausfeld, Kikkert, & Bonte, 2015; Schulz, Zherdin, Tiemann, Plant, & Ploner, 2012).

1.2.2 Brain State Detection

Another application of classification involves the continuous monitoring of brain states using imaging devices like the EEG. Rather than looking to control an external device, these techniques look to determining the subject's inner state. These applications tend to consider longer periods of data and focus on frequency analysis.

These techniques are being researched for use in areas such as seizure detection and prevention (Gabor, Leach, & Dowla, 1996; Ramgopal et al., 2014) and truth detection (Gao et al., 2013). Brain state detection is also used as part of larger applications, such as monitoring mood to allow a larger human computer interface system to adapt its display to user's current mental state (Molina, Nijholt, & Twente, 2009). There is even interest in the classification of more disparate states such whether the subject is resting quietly, remembering events from their day, performing subtraction or silently singing lyrics (Shirer, Ryali, Rykhlevskaia, Menon, & Greicius, 2012).

1.2.3 Medical Diagnosis

Brain signal classification is likely to play an increasing role in the diagnosis of brain diseases in the future. While convergent evidence will always be necessary, classification could be useful as a screening tool or another point of reference for diagnosis.

For example, efforts have been made into using SVM techniques to aid in Alzheimer's disease diagnosis (Trambaiolli et al., 2011). Other applications in diagnosis include drug addiction (Zhang, Samaras, Tomasi, Volkow, & Goldstein, 2005) and diagnosis of psychiatric disorders, such as schizophrenia (Koutsouleris et al., 2015), ADHD (Fair, Bathula, Nikolas, & Nigg, 2012), and bipolar disorder (Fair, Bathula, Nikolas, & Nigg, 2012).

1.2.4 Brain Computer Interfaces

Brain computer interfaces (BCIs) use monitored brain activity and computation to achieve an external activity. Classification of mental states and intentions based on patterns of brain signals is a common goal in such applications. While early methods of BCIs have tended to use manually determined features and relied on the user adapting to the machine, more modern techniques generally involve the use of machine learning and allow the machine to adapt to the user.

These BCIs have a wide range of applications, from control of robotic arms and other prosthetic devices (McFarland & Wolpow 2008) to speech synthesizers (Lotte, Congedo, Lécuyer, Lamarche, & Arnaldi 2007) and other communication devices (Guan et al., 2004). In general, most BCI applications are geared toward providing capability to the disabled.

1.3 Brain Imaging Techniques

A variety of devices and techniques capable of measuring brain signals are available. These techniques either measure primary signals of activity such as the electrical or magnetic signal produced by neural activity, or secondary signals such as the blood flow to regions of the brain that are active.

Functional Magnetic Resonance Imaging (fMRI) measures what is known as the blood oxygen level dependent (BOLD) signal. It is capable of high spatial resolution and imaging deep brain structures, but requires a room free of electromagnetic interference and very expensive equipment. Furthermore, the temporal resolution of the signal is quite poor since it relies on measuring blood flow rather than a direct marker of brain activity. The output of fMRI, after several statistical methods are applied, is a series of voxel based images of the BOLD signal.

Magnetoencephalography (MEG) measures the magnetic component created by the electricity moving through the brain. It has both a relatively high spatial resolution and a very high temporal resolution, since it measures a direct and quickly propagated marker of brain activity that is largely unaffected by the scalp. While the spatial resolution is fairly good, only outer portions of the brain can be accurately measured due to the drop off in strength of magnetic fields with the square of the distance. Additionally, it requires a highly magnetically shielded room and a constant supply of liquid helium to function, leading to very high costs and a lack of portability. The output of MEG is one time series per channel (generally 306 channels) representing the strength of the magnetic field at the channel.

Other brain imaging modalities such as standard *magnetic resonance imaging* (MRI) and *positron emission tomography* (PET) do not offer the temporal resolution necessary to monitor brain activity at time scales useful in most classification applications and may have other drawbacks, including exposure to ionizing radiation.

Thus, due to the disadvantages of other brain imaging modalities, this thesis will focus on data collected by *electroencephalography* (EEG). EEG functions by attaching electrodes to a subject's scalp in order to measure the changes in electrical potential that occur as a result of brain activity. Due to the near instantaneous propagation of these

voltage changes, information acquired by the EEG can be sampled with high temporal precision. However, the human skull and scalp are insulators, which have the effect of dispersing the signal, thus limiting the spatial resolution capable of being achieved by the EEG. Furthermore, localizing the source of activity involves solving an inverse problem with an infinite number of solutions, thus further limiting the spatial resolution. However, it is comparatively inexpensive and does not require onerous protections such as magnetically shielded rooms. Furthermore, it is, unlike the previously discussed modalities, more practical in applications that require mobility, such as BCI. The output of EEG is one time series per channel (anywhere from 32 to 256 channels is common) that represents the electrical potential on the scalp at the given channel with respect to a reference electrode, typically recorded at rates from 250 Hz to 1000 Hz.

1.4 Research Challenges and Contributions

Brain imaging data presents several challenging obstacles to machine learning, all of which are current topics of open research:

- *Brain signals are noisy*. The information is polluted by a variety of factors including muscle movement, measurement error, brain activity that is not of interest, and electromagnetic interference from the environment.

- *Brain signals have high dimensionality*. There are frequently hundreds of channels, sampled at up to 1000 Hz. The raw data frequently presents up to hundreds of thousands of features per trial.
- *The data are a time series and have spatial interactions*, potentially requiring investigation of temporal and frequency components in conjunction with spatial analyses.
- *Data collection is expensive and time consuming*. Thus, it can be difficult to collect sufficient data for many of the most powerful machine learning techniques. Generally thousands or tens of thousands of samples per class are desired in modern deep learning applications, but it is often impractical to produce more than a few hundred samples per subject in brain imaging tasks, particularly if they need to be derived for clinical or medical studies.
- *Brain activity can vary significantly between subjects* and even between data acquisition sessions within a subject. Thus, classifiers that can address high levels of variability are needed.

The goal of this thesis is to investigate techniques for addressing these challenges. In this thesis we have demonstrated the effectiveness of various deep learning architectures, particularly convolutional and recurrent, in classifying brain signals. We

have shown that certain recurrent architectures outperform traditional techniques. Furthermore, transfer learning strengthens the effectiveness of all deep learning architectures.

1.4 Thesis Outline

The rest of the thesis is organized as follows: Chapter 2 will review the related works that forms the basis for this research. Chapter 3 will define the specific problems and approaches used in this thesis. Chapter 4 will include a summary of the datasets used, the implementation of the techniques used, and the results of the experiments. Chapter 5 will present a summary of the findings and a discussion on future work.

Chapter 2

Related Works and Background

This chapter is made up of two sections, one covering the basic concepts behind the deep learning algorithms used in this paper and the other covering the current state of EEG classification.

2.1 Deep Learning

2.1.1 History of Neural Networks

Deep learning is a subfield of machine learning that has evolved out of the traditional approaches to artificial neural networks. Artificial neural networks are computational systems originally inspired by the human brain. They consist of many computational units, called neurons, which perform a basic operation and pass the information of that operation to further neurons. The operation is generally a summation of the information received by the neuron followed by the application of a simple, non-linear function. In most neural networks, these neurons are then organized into units called layers. The processing of neurons in one layer usually feeds into the calculations of the next, though certain types of networks will allow for information to pass within layers or even to previous layers. The final layer of a neural network outputs a result, which is interpreted

for classification or regression. Figure 1 shows a depiction of the structure of a simple neural network.

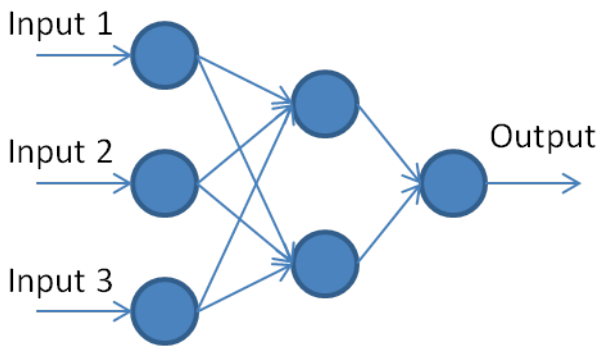

Figure 1 Structure of a basic neural network

The basic concepts date back to the McCulloch-Pitts neuron of the 1940s (McCulloch & Pitts, 1943). This model was very simple compared to modern neural networks—it only allowed for binary outputs from each neuron, summing the input and comparing the result to 0. Furthermore, there was no update rule defined. An update rule is a mathematical rule that allows for the adaptation of the neural network to new information. Without an update rule, all of the values in a neural network must be handcrafted.

In the 1950s, the perceptron algorithm was introduced. It generalized the McCulloch-Pitts neuron to have continuous valued weights on the connection and introduced a basic update rule to compute the weights at time $t + 1$ from the weights at time t:

$$w_i(t+1) = w_i(t) + (d_j - y_j(t))x_{j,i}$$

where w_i is the weight of the i^{th} neuron, d_j is the expected value of the j^{th} input, $y_j(t)$ is the calculated value of the j^{th} input, and $x_{j,i}$ is the i^{th} value of the j^{th} input.

The perceptron learning rule was only capable of training networks with a single layer, greatly limiting the power of the model. It was originally conceived as a hardware implementation, though it was also the first neural network to be implemented in software (Rosenblatt, 1958).

In the late 1960s, it was mathematically proven that a network with only a single layer lacks the representative power to classify many common types of problems, including the Exclusive OR (XOR) function (Minsky & Papert, 1969). Furthermore, attempts to use neural networks in speech recognition during this era were largely considered failures, capable of only recognizing a very limited vocabulary of words for a single user (Pierce, 1969). This combination of theoretical limitations and practical failures led to an "AI Winter". Funding for neural networks and other forms of AI research largely dried up over this period. The 1970s saw very little progress in the development of neural networks.

The ability to train a network with multiple layers was crucial for the continued development of neural networks. While Minksy and Papert's book was most famous for

showing that a single hidden layer network was lacking in representative power, it also showed that a network with even two hidden layers could model almost any function. A method known as automatic differentiation was proposed in Seppo Linnainmaa's Master's thesis in 1970 for calculating the derivative of a differentiable composite function represented by a graph (Linnainmaa, 1970). This method would later form the basis for backpropagation, a learning rule capable of training neural networks with multiple layers (Werbos, 1982). In essence, backpropagation allows multiple layer networks to be trained by iteratively using the gradient of a loss function with respect to the weights of the network to assign updates to previous neurons in the network, starting from the output neurons. It can be described in pseudocode as:

```
do
        prediction = networkOutput(training_examples)
        actual = label(trainin_examples)
        error = averageErrorFunction(prediction, actual)
        grad = gradient(l, error)
        for layer l in reverse(net)
                new_l = update(l, grad)
                grad = grad(l, grad)
                l = new_l
while !(all samples classified correctly or stopping criteria reached)
```

With the advent of backpropagation, neural networks began to see renewed interest and significant theoretical advancement in the 1980s. New forms of networks arose including Hopfield networks, which set the groundwork for modern recurrent neural networks (RNN) and the Neurocognitron, which inspired modern convolutional neural networks (CNN) (Hopfield, 1982; Fukushima, 1980). However, the lack of raw processing power and the lack of techniques to handle the vanishing gradient problem, the tendency for the backpropagated error gradient to approach zero causing early layers of the network to fail to train, led to disappointing performance from neural networks. By the beginning of the 1990s, neural network research had fallen into disfavor again. New techniques, including support vector machines, were introduced and proved far easier to train at the time (Cortes & Vapnik, 1995).

In 2006, Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh published a paper titled “A Fast Learning Algorithm for Deep Belief Nets” which showed promising results in the classification of the Modified National Institute of Standards and Technology handwritten dataset (MNIST), a standard dataset in the field. The paper proposed a greedy technique to train a neural network built out of statistical units, known as Restricted Boltzman Machines, layer by layer. By only training one layer at a time, this technique avoided the vanishing gradient problem and allowed for deeper networks than

previously viable, and far faster training. Hinton's paper reignited interest in neural networks and paved the way for modern deep learning.

2.1.2 Deep Learning Overview

Deep Learning is a set of techniques that are a natural progression of traditional neural network techniques. These include:

Stochastic Gradient Descent (SGD) algorithm and its descendants (e.g., Bottou, 2010). Gradient descent is a first-order iterative optimization algorithm used for finding the minimum of a function. In neural networks, it is used in conjunction with backpropagation to update the weights in the network. It is formally defined with the update rule:

$$\vec{x}_k = \vec{x}_{k-1} - \eta \nabla f(\vec{x}_{k-1}),$$

where $\vec{x}_k$ is the current point in the space, $\vec{x}_{k-1}$is the previous point in the space, η is the learning rate, and $\nabla f(\vec{x}_{k-1})$ is the gradient of the value of the function being optimized at the previous point.

SGD is a derivative of traditional gradient descent, differing in that the error function is calculated using only a subsample of the available data. This is both easier to use and more efficient for training datasets that do not fit in memory. Furthermore, adding randomness to the optimization can aid in breaking through plateaus and avoiding

local minima. The addition of momentum terms, which biases the gradient in the direction of recently calculated gradients, greatly improved the ability to train deep models by further increasing speed of convergence (Sutskever, Martens, Dahl, & Hinton 2013). Newer SGD derived algorithms, such as Adaptive Moment Estimation (ADAM), calculate per parameter adaptive learning rates, enabling even more efficient training, at the cost of memory (Kingma & Ba, 2015).

New Activation Functions: One of the largest and most persistent problems in the development of neural networks is known as the vanishing gradient problem. In essence, as the gradient is propagated back along the network, the error is multiplied by values between 0 and 1 repeatedly. This causes the error, and thus the update, to trend toward 0 exponentially, resulting in little to no ability to update the early layers in a multilayer network (Hochreiter, 1991). Sigmoid activation functions, which were historically the most prominently activation function, are especially susceptible to this problem due to having a first derivative that rapidly tends toward zero as a neuron saturates. The sigmoid function is defined as: $\frac{1}{1+e^{-x}}$. The Rectified Linear Unit (ReLU), which maps the input to 0 if it below 0, and to itself if the input is above 0, and other modern activation functions have larger gradients and saturate less quickly, thus avoiding the vanishing gradient problem more effectively (Jarrett, Kavukcuoglu, Ranzato, & LeCun, 2009).

New types of layers. Perhaps the biggest difference between traditional neural networks and deep learning is the adoption of new types of layers in the network. Traditional neural network research focused on fully connected layers, in which every neuron in one layer is connected to every neuron in the next. While many of these layer types existed in the past, they usually were not able to used to significant effect due to various issues in training.

Convolutional neural networks learn filter banks that are convolved with the original data. The filters can also be represented as a fully connected layer where the weights of the edges are tied together in way that replicates the convolution operation. This weight sharing structure allows for fewer parameters than having each weight be unique, and directly accounts for structure in the data. As shown in Figure 2, each filter creates a new, processed version of the image.

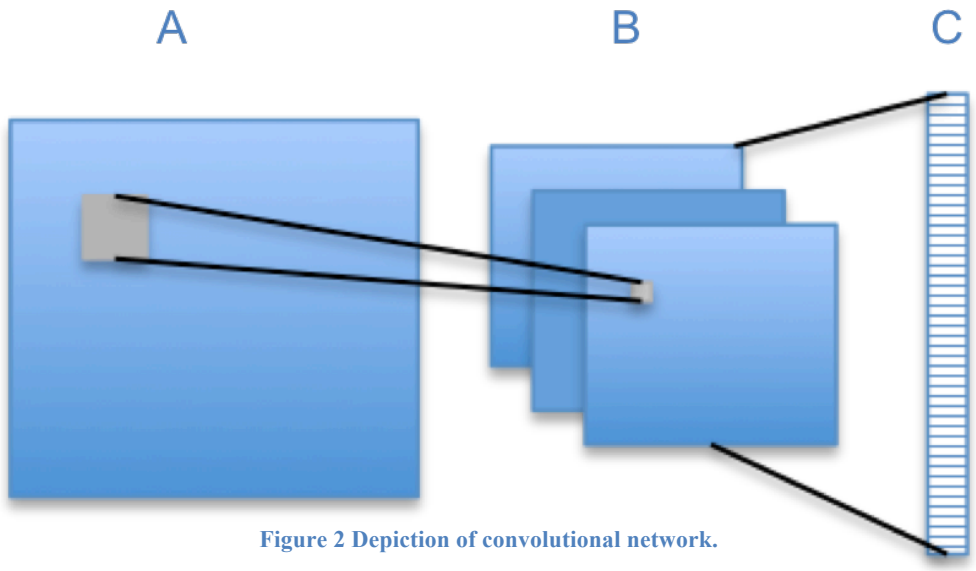

Figure 2 Depiction of convolutional network.

(A) is the original data with a convolutional filter being applied. (B) is the transformed data with each filter applied, showing where the operation in (A) mapped. (C) shows the data flattened for use in MLP or classification layers.

Recurrent Neural Networks (RNN), networks with connections within a layer or from a layer to a previous layer, have contributed to the success of deep learning in many fields, particularly in time series classification. Training an RNN is a difficult task. Backpropagation must be modified to function in RNNs, since there are cycles in the graph. This is frequently handled through a technique known as backpropagation through time, wherein the network is "unrolled" for a discrete number of steps. This process creates a network with only forward connections, allowing backpropagation to work as normal, at the cost of limiting the impact of the recurrent connections(Werbos, 1990). Because of this, only a few architectures have seen widespread use and success in classification tasks.

In particular, Long Short-Term Memory (LSTM) networks have proven successful in recent years, though they were introduced in the 1990s (Hochreiter & Schmidhuber, 1997). It was not viable to train an LSTM network on the hardware available in the 1990s due to the large number of computations necessitated by recurrent architectures. The specific architecture of an LSTM network is shown in Figure 3. LSTMs are made to function specifically with time series or sequential data. Each time point in the data is processed by its own LSTM unit, and the results are both passed to the next layer and to the processing of the next time point within the same layer. The

information passed through to the next layer is passed through an activation function, such as the *tanh* unit in Figure 3; however, the recurrent connection within the layer is not subjected to an activation function. The lack of an activation function in the recurrent layer is critical in avoiding the vanishing gradient problem, as it allows the gradient to be a constant value of one. Each LSTM unit has a number of gates which control the flow of information. A gate is a combination of a sigmoidal activation unit and pointwise multiplication. These gates control the amount of information that flows from one time point to the next, the amount of information that is the output of each unit, and other functions of the LSTM unit.

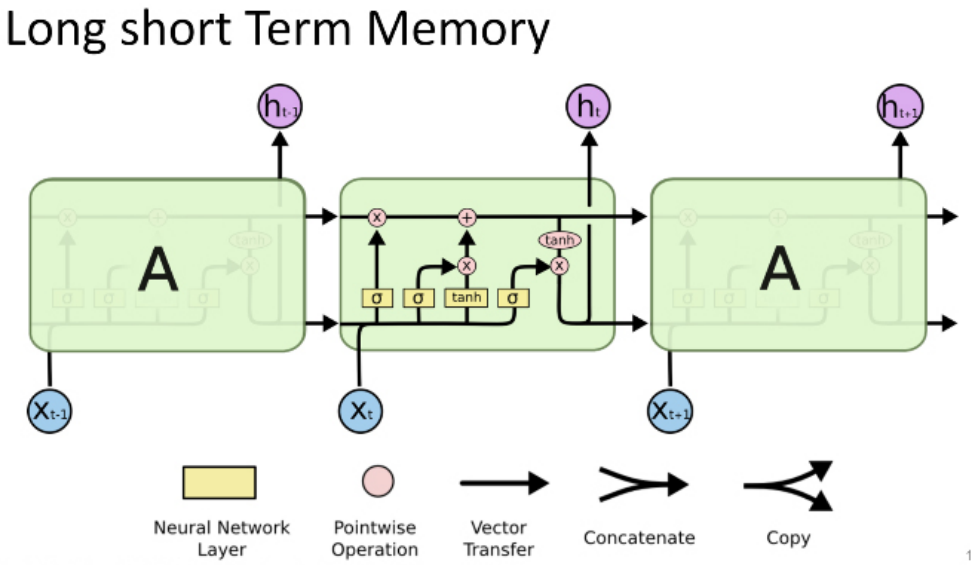

Figure 3 Architecture of an LSTM network

Proposed by Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

The advent of more powerful graphics cards and more convenient programming for those cards (Krizhevsky 2012). In 2007 NVIDIA debuted the Compute Unified Device Architecture (CUDA) API to that enables direct and easy programming for their graphical processing units (GPUs). GPUs are specialized processors designed particularly for graphical display. Since most graphical applications rely on large amounts of calculations that can be run in parallel, these processors have a large amount of cores. Over the next several years, CUDA gained popularity for use in highly parallelizable tasks able to take advantage of the large number of cores in GPUs. Deep learning is implemented as large matrix operations, which is one such task. Furthermore, the memory capacity and processing capabilities of GPUs increased dramatically from the launch of CUDA to 2012. The hardware and software advances in GPU programming were critical for Krizhevsky's 2012 ImageNet. Prior to these developments, training deep neural networks was essentially intractable- even early GPU implementations showed a 70X speed up over standard CPUs (Raina, Madhavan, & Ng, 2009).

Dropout and other regularizers (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Overfitting, the phenomenon of learning patterns that happen to be present in the training data by random chance and not present in general, is a constant threat in deep learning due to the immense numbers of parameters involved.

Regularization techniques prevent the model from becoming too complex and specific to the training data, thus reducing the tendency to overfit. L2 regularization is a standard technique used in many forms of optimization, in which the squared sum of the weights is applied as a penalty to the optimization function. In essence, this weighs the benefit of increased classification of the training data against model complexity. By preventing the model from becoming too complex, memorization of the training data is reduced, and a more generalizable model is developed.

Dropout is a technique in which a random set of neurons from each layer (generally between 30% and 60%) is excluded from both classification and updating during a training pass through the data. This effectively allows a single model to act as an ensemble, a group of classifiers that act in union to produce a classification. Furthermore, it prevents the direct memorization of training data, since different sets of neurons will be participating from one pass to the next.

Increasing rate of data collection: More data allows for larger, more expressive models to be developed. By having a larger sample size, more variance will exist in the data, and can be directly accounted for by the model. Similarly, variance in the training data that happens by chance will have less of an effect on the model since each sample will contribute less to the overall gradient during backpropagation, leading to better

generalization to new data. Deep learning shows its greatest strength in problems of very high complexity where traditional learning techniques have struggled.

2.2 EEG Classification

The classification of EEG signals presents several challenges that make it a uniquely difficult problem in machine learning. The EEG signal is high dimensional, with both spatial and temporal covariance. The high dimensionality is difficult to account for in traditional machine learning techniques, leading to a desire to extract features from the signal to reduce the dimensionality. However, many time series approaches for feature extraction face issues with the non-stationary nature of the EEG signal. A stationary signal is one in which the probability distribution does not change over the course of time, and, thus, features like mean and variance will not change. While data can be transformed to account for non-stationarity, it is an imperfect solution. The signals are also very noisy, being susceptible to factors such as physical movement, mood, posture, and external noise (Kevric & Subasi, 2017).

Another issue faced in the field is a lack of comparability between experiments. Unlike in image classification, there are no standard datasets used as performance benchmarks. Not only is data collected on machines with differing specifications, and individuals with differences in physiology, but entirely different tasks are also performed

during data collection leading to different target brain activities. Furthermore, some approaches use models for individuals, whereas others attempt to make a universal model, training and testing with samples from all individuals at one time. In general, the focus has been on the use of machine learning as a tool to draw conclusions in neuroscience, rather than improving the techniques of classification. Thus, while this section will explore techniques used, it will not focus on accuracy of the given techniques, since it is nearly impossible to compare them at the present.

2.2.1 Traditional Approaches

The majority of published techniques currently use some form of feature extraction followed by classification with a relatively simple model. These techniques require less implementation time than deep learning, and are less prone to overfitting. Support Vector Machines are one such model. They function by projecting the training data to a very high dimensional space, and then finding a hyperplane in that space that best separates the classes of data. Linear Discriminant Analysis (LDA) is another commonly used classifier which uses probabilistic modeling with assumptions of normality and homoscedasticity (both classes have the same variance) to create a linear decision surface. Further common classifiers include Naive Bayes, a simple probabilistic classifier which uses the concept of maximum likelihood to make decisions, and K-

Nearest Neighbors, which queries the training data using a distance metric to find the K samples closest to the one being classified to determine the label.

The most common feature extraction methods include: Fourier transforms and related methods, wavelet transforms, principal component analysis (PCA), independent component analysis (ICA), autoregressive methods, or combinations of those techniques.

The Fourier transform is likely the most common feature extraction method. It provides information on the frequency spectrum of the data, at the cost of losing temporal information. To combat this, short time Fourier transforms are more common in practice. The Fourier transform is used to calculate features, such as power spectra, which match nicely with EEG literature on brain activities that occur within different frequency bands (Wang, Nie, & Lu, 2014, Li & Lu, 2009). Al Zoubi, Calvo and Stevens (2009) used power spectral density values in 1 Hz bins from 2 Hz to 40 Hz to classify affect data from self-elicited emotions. There were 10 emotions used, with 180 trials each for 1800 samples per subject across three subjects. The study compared KNN, SVM, and Naïve Bayes. KNN was found to be the most effective, with up to a 66% classification rate compared to SVM at 54%, Naïve Bayes at 43%, and a baseline of 10% accuracy. The accuracies varied greatly among the subjects, however.

The wavelet transform is another common signal processing tool used in feature extraction. Wavelet transform methods also decompose the signal into frequency based information, but, unlike the Fourier transform, time and frequency trade-offs are built into the model (Unser & Aldroubi, 1996).

Another, less commonly applied transformation is the Hilbert-Huang Transform (HHT). Much like the wavelet transform, HHT is inherently a time frequency method. Unlike both the wavelet transform and FFT, HHT derives an adaptive basis rather than relying on an a priori basis. Additionally, the basis of the HHT is empirically determined, and thus not guaranteed to be complete (N.E. Huang & Wu, 2008). The benefits of this approach are that it does not assume stationarity or linearity of the signal – both of which are known to be false of EEG, but assumed in other frequency and time frequency methods. In a comparison with the wavelet transformation, HHT was found to produce more accurate results in the analysis of higher frequency brain signal information in certain BCI settings (M. Huang, Wu, Liu, Bi, & Chen, 2008).

2.2.4 Deep Learning Approaches

Presently, very few attempts have been made to apply deep learning to EEG classification. Bashivan et al. (2015) used a mixture of convolutional and recurrent networks on EEG "movies" wherein the channels of the EEG were projected onto a two

dimensional plane. After the projection, the Fourier analysis was applied across several time bins to reduce the data to 7 “frames”. The preprocessed data was then classified with a hybrid neural network containing a convolutional layer fed into a mixed long short-term memory (LSTM) and 1-D convolutional layer, which was fed into a fully connected layer for classification. This method obtained impressive results, lowering the error rates they had achieved with SVMs and random forests by up to 50%.

The use of fully convolutional neural networks (i.e., no fully connected layers) was explored by Lawhern et al. (2016). They applied 2D convolutions across data arranged in a channel by timepoint fashion. Fully convolutional networks have the advantage of having very few parameters (less than 2,200 parameters across 4 layers of convolutions in this case), thus requiring significantly less data to properly train. They achieved state-of-the-art performance in several paradigms, including: P300 event-related potential in an oddball task, error-related negativity in BCI, movement-related cortical potential in a finger movement task, and sensory motor rhythm in imagined movement.

Work by Stober, Sternin, Owen, & Grahn (2015) explored the use of convolutional autoencoders to “capture invariance between trials within and across subjects”. An autoencoder is a neural network where the input is also used as the label for the output. By having hidden layers with less parameters than the input (and thus output),

these neural networks learn a compressed version of the data. They also developed an architecture dubbed Hydra-nets, which featured multiple versions of certain layers of the network, allowing for custom tailoring the network based on features of the trial metadeta, such as which subject was being classified. They claimed the models had sufficient simplicity to allow for interpretation of learned features by domain experts, though no specific interpretations were provided.

2.3 Research Challenges

Currently, there is a lack of comparability in EEG classification literature. Most studies focus on using machine learning to answer a neuroscience question, rather than on improving the machine learning techniques themselves.

Generally only a single technique is applied to the classification of a dataset, so it is difficult to draw conclusions on the effectiveness classification system as compared to another. Individual datasets have highly varying characteristics, and some are simply harder to classify than others. Thus, there is a need for comparative studies.

Deep learning has largely been explored in a similar manner, with only a single type of architecture applied to a unique dataset in each study. There is a lack of comparative information between different architectures. Furthermore, training a deep learning classifier on EEG data is a difficult task. There are many ways to go about

training an individual architecture. Whether to consider subjects individually or as a group is important to explore. Other training techniques, such as transfer learning, have also remained largely unexplored, and have the potential to help with the issues of low data availability in EEG classification.

Chapter 3

Problem Definition and Approach

3.1 Problem Definition

Simply stated, the problem explored in this thesis is the classification of EEG data using deep learning techniques. While the classification of EEG signals can be useful in many areas, such as the detection of disease state or brain computer interfaces, this thesis focuses on the classification of which of three classes of visual stimuli a subject is viewing or thinking about based on the evoked brain activity. These sorts of stimuli paradigms are common in psychology and neuroscience for both basic research on perception and memory, and also as datasets for technique validation. The problem of EEG classification in these types of experiments has a number of challenges that must be considered, including:

1. There is high variability between subjects and within subjects,
2. There is limited availability of data, and
3. The data is composed of multiple channels of time series information

This thesis will address these challenges by exploring several variations of architecture selection, model search, regularization, and training paradigms within a deep learning

context, with the aim of harnessing the expressive power of deep learning while avoiding overfitting.

To state the problem formally: Given a training set of EEG signals, E = $\{s_1 \dots s_k\}$, where each signal s_i is composed of channels $c_1 \dots c_n$, with each c_j is a time series $t_{1,j} \dots t_{f,j}$. Let L be a true label function that maps each $s_i \in$ E to a set of classes C = $\{c_1 \dots c_n\}$. Let $\hat{E}$ by a novel set of EEG signals. The goal is to create a classifier K, with parameters θ and hyperparameters Ψ, such that K optimizes the expected value of the function $\Phi((\Gamma(K, L, \theta, E), \Psi, \hat{E}, L)$, where Φ and Γ may be any valuation metric, for example accuracy or categorical cross-entropy. In other words, we are searching for a classifier with high accuracy for the training set, the role of function Γ, which also generalizes well for the unknown test set, the role of function Φ.

3.1.1 High Variability

The highly variable nature of EEG data leads to difficulty in classification. Samples in the same class may be very different in nature from one another. There are multiple sources of variability both within subjects and between subjects.

Sources of within subject variability are briefly summarized below:

Level of attention. If the subject is not focused on the task, the patterns of activity produced by their brain will be quite different. This not only impacts data quality, but also reduces the amount of data that can be collected.

Multiple Sessions. Many experiments call for the subject to attend multiple sessions of data collection. The subject can be more or less focused, and may be in a different mood, both of which affect the data. Another common source of variability is the physical placement of the EEG electrodes. It can be very difficult to insure they are in the exact same location. Thus, channels may be collecting data from a slightly different source from one trial to the next.

Muscle movements. A constant issue in EEG data analysis is that the electrical activity created from muscle movement has a far higher magnitude than that produced by brain activity. This is particularly noticeable and problematic when the subject blinks. There are methods to reduce the impact of certain types of movements, including eye blinks, such as using independent component analysis to regress out the artifact. However, these methods are imperfect.

Machine noise. There is a certain amount of uncertainty inherent from the machine itself. Slow drifts in the data are common and are caused by either slight movements of the electrode or sweat interfering with the sensor. Movement in wires

connecting the electrodes can cause similar issues. These issues can often be mitigated through the use of band pass filters, however.

Sources of between subject variability are briefly described below:

Differing physiologies. Differences in skull shape can lead to electrodes monitoring different relative portions of the brain. Thus, the brain activity collected by an electrode on one subject may be from a slightly different region than the activity collected by the same sensor on a different subject.

Differing cognitive patterns. There are individual variations in how information is processed, and thus, the same stimulus may illicit differing responses in different subjects.

Differing behavior. Some subjects will be more focused on the task than others. Some will perform better than others on the experimental task. Differing behavior is associated with differences in brain activity patterns.

3.1.2 Limitations of Dataset

As previously discussed, deep learning requires a large amount of training samples to prevent over-fitting due to the immense number of parameters in the model. Most successful applications have tens of thousands to millions of samples. However,

collecting this amount of data for EEG tasks is at best difficult, and, frequently, intractable.

First and foremost, there is no way to automate the data collection. Proper EEG recording requires trained experts and takes considerable setup time. The electrodes most be properly connected and the subject most be observed to make sure they are participating in the experiment correctly. Thus, it places significant time demands on the expert during data collection, limiting the number of subjects for a study.

In addition, individual subject can only be expected to focus on a task for limited periods of time. As mentioned in the previous section, as attention wanes, brain activity changes significantly. Thus, data collection sessions are strongly constrained in their durations. Scheduling individual subjects for multiple sessions poses additional challenges. It may be difficult for some subjects to participate multiple times due to time constraints; it may be impossible for many subjects due to health reasons. It is also time consuming to set up the sensors multiple times. Finally, as mentioned previously, brain activity in a subject can differ from session to session, so collecting samples over multiple samples is at best an imperfect compromise.

3.1.3 Multiple Channels and Time Series

EEG data is recorded as a time series of floating point numbers at 250-1000 Hz, generally using 32 to 256 channels. Each sample is usually between half a second and five seconds long. These factors lead to several important implications that make classification task harder.

First and foremost is that the data is very high dimensional. In the case of an EEG with a small number of electrodes (e.g. 32) recorded for only a second, there are still 30,000 or more features in a single sample. Given that only a small number of overall samples that can be collected (hundreds per class per subject, generally), the curse of dimensionality is a real challenge.

Secondly, the data is a time series in nature. This means that the value at a specific point in the signal isn't as important as the pattern of values, in general. Furthermore, there can be small variations in the onset of the pattern. So, what happens at time point 10 in one sample may not occur until time point 15 in another sample. Many models have difficulty classifying data of this nature.

Finally, the information is distributed over multiple channels. The important information to distinguish between two classes may not only lie in the patterns over one

channel, but rather, the joint patterns of activity over multiple channels. This is again, very difficult for many types of models to efficiently incorporate.

3.2 Approach

In order to address the challenges presented in classifying EEG data, we take a three-part approach. First, we establish a baseline using traditional methods. Second, we perform a model search and hyper-parameter search among deep learning architectures. Finally, we look into the use of transfer learning, a technique of pre-training a network on one set of data and then fine-tuning on another.

3.2.1 Traditional Machine Learning Baseline

Given the highly diverse nature of EEG datasets, it is important to establish a baseline using traditional methods on the particular data we are classifying. A strong classification performance on one dataset, may only be a mediocre performance on another.

We establish this baseline using two common techniques in EEG classification, SVM and SMLR. SVM represents the most commonly used classifier in EEG research, while SMLR has been shown to be more successful on many datasets (e.g. Gu, Poesio, & Murphy 2014; Johnson, McCarthy, Muller, Brudner, & Johnson, 2015; Rebsamen, Kwok, & Penney, 2011).

We also explore the use of feature extraction in establishing a baseline. While wavelets and Fourier techniques are very common in literature, we consistently found them to underperform in similar datasets. In the initial publication of this data, the authors found no advantages in the use of Fourier transform (Johnson, 2017, personal communication). Average time-binning was used as the feature extraction method in the original paper, though the data without feature extraction was not explored. Average time binning is the process of reducing the dimensionality of a time series by taking the average over a time window. That is, given a time series T = $\{t_1 \dots t_n\}$, window *k*, and stride *s*, create a new time series, $\hat{T} = \{\hat{t}_l\}\ 1 \leq l\ \leq n/s$, where $\hat{t}_i$ = average($t_{sl} \dots t_{k+sl}$). For example, for a time series with n = 1000 and s = 10 and k= 10, after time binning, the time series would have a length of 100. In order to maintain comparability, we use time binning as the method of feature extraction. As deep learning is considered a form of representation learning, learning its own features in the lower layers of the network to classified in the terminal layers, we also explore not using any explicit method of feature extraction for comparability (LeCun, Bengio, & Hinton, 2015).

Finally, we examine two training paradigms: single-subject and universal. Much of the literature focuses on models trained on data from single subjects. While this allows the model to account for individual differences, it also reinforces the issues with the

dimensionality and low number of samples available per subject. Thus, universal models potentially have an advantage if the greater number of samples improves the models ability to generalize more than it is hurt by having to train on differing patterns between individuals.

3.2.2 Deep Learning Model Search

An immense number of diverse deep learning architectures exist. In particular, we examine four general classes of models. These models include the multi-layer perceptron (MLP), convolutional neural network (CNN), long short-term memory network (LSTM), and a hybrid CNN-LSTM. The MLP was chosen to provide a baseline, while the remaining architectures were chosen due to their high performance in other signal classification tasks.

Multi-Layer Perceptron: The first class of model is the multi-layer perceptron. This class of model is composed of entirely fully connected, feed-forward layers. This type of architecture does not account well for the spatial and temporal variance inherent in EEG data. Since each weight is directly associated with an individual time point in an individual channel, if the signal occurs slightly later or in a slightly different region of the brain, the model will have difficulty adapting. Thus, the MLP model is only considered as a baseline for the other deep learning models.

Convolutional Neural Network: The second type of model considered is the convolutional neural network (CNN). In particular, we focus on 2-dimensional convolutions, with one dimension over channels and the other over time. While flattening the channels to a single dimension loses some spatial information, this methodology is consistent with published work (Lawhern et. al., 2016) and allows the model to inherently account for spatial and temporal structures. Furthermore, since the model learns filters that are learned across the data, rather than weights on individual data points, it far more robust to spatial and temporal variance.

Convolutional Neural Network: While many convolutional architectures exist, we focus on fully convolutional networks, as presented in Lawhern et. al. 2016. These networks do not contain fully connected layers at the end of the model like those discussed in the background chapter. Rather, they contain only the convolutional layers, potentially max pooling layers, and the soft max classification layer. This reduces the number of parameters in the model to a few thousand, as compared to the hundreds-of-thousands or even millions of parameters in other deep learning architectures. This both aids in convergence for the model and acts as regularization. With fewer parameters, less data is needed for convergence and low data availability is one of the primary issues of

EEG classification. Furthermore, fewer parameters also helps to prevent overfitting, thus acting as a regularizer.

Long Short-Term Memory: The third type of model we explore are recurrent models, in particular LSTMs. LSTM layers are recurrent layers that allow the activation at one time point in a layer to directly affect the next time point. This allows the model to directly account for temporal structure in the data. Each LSTM layer takes input from multiple channels, allowing them to directly account for spatial effects. However, since there is only a single weight per channel per time point connecting to the LSTM layer, this is not likely to offer the same spatial invariance garnered from the convolutional models. Further, LSTM models are highly prone to overfitting (Zaremba, Sutskever, & Vinyals, 2014), which is generally the greatest difficulty in classifying EEG data successfully.

LSTM + CNN: Finally, we explore combination LSTM and convolutional networks. These models will first do the temporal processing with LSTM layers and then apply convolutional layers to allow for increased temporal and spatial invariance in the model. However, the combination of layers can lead to a very high model complexity.

Each of these basic architectures represents an infinite family of potential models. In order to ensure that the full value of the deep architectures is realized, hyperparameter

search is necessary. A hyperparameter is a value set before the learning algorithm begins. These include everything from the activation function chosen for a given neuron to the number of neurons in a layer to the learning rate and choice of optimization function. The weight of an edge between two neurons is not a hyperparameter, since that value is learned. Since the number of network configurations grows with the product of the choices per hyperparameter, the space is combinatorially large.

Grid search has been used in similar problems historically, however has proven to be ineffective in deep learning due to the size of the hyperparameter space and time required to evaluate a single configuration. Random search has proven to be more effective (Bergstra & Bengio, 2012), but is unappealing due to the loss of interpretability and the requirement to set bounds for the hyperparameters a priori. Thus, we use a manual hyperparameter search using several heuristics.

Figure 4 provides a flow chart of the basic logic used in the heuristic model search. After training the model with a given set hyperparameters, the training accuracy is examined first. If the training accuracy is low, then the number of parameters in the model was increased. Since classification performance is potentially limited by both the dataset itself and the basic type of model being explored (e.g., an MLP), the definition of low must be relative and determined separately for each type of model. Generally, to

increase the number of parameters in the model, either more neurons are added to layers or more layers are added to the model. Other actions are also explored at this step, including changing activation functions and adjusting the learning algorithm, including changing the learning rate or the algorithm used. Those actions are rare compared to simply increasing parameters, however.

If training accuracy is high, then the validation accuracy and test accuracies are considered. If the validation and test accuracies are also high, then the selection of hyperparameters can be accepted, or more tweaks can be made to the hyperparameters to determine if the working definition of high accuracy is sufficient. If the test and validation accuracies are low when the training accuracy is high, the model is most likely overfitting. There are two main ways of handling overfitting: increasing the amount of regularization in the model and decreasing the number of parameters in the model. Increasing the amount of regularization includes changes such as adding or increasing L2 penalties on weights, adding or increasing dropout, or adding batch normalization to layers. Reducing the number of parameters is achieved by either reducing the number of neurons in the layers or reducing the number of layers. Generally, reducing the parameters of the model is reserved for when increasing regularization fails to improve validation accuracy to avoid unnecessarily reducing the expressive power of the model.

However, if the model performs nearly perfectly on the training data while performing very poorly on the validation data, reducing the number of parameters can be the most effective change to the hyperparameters.

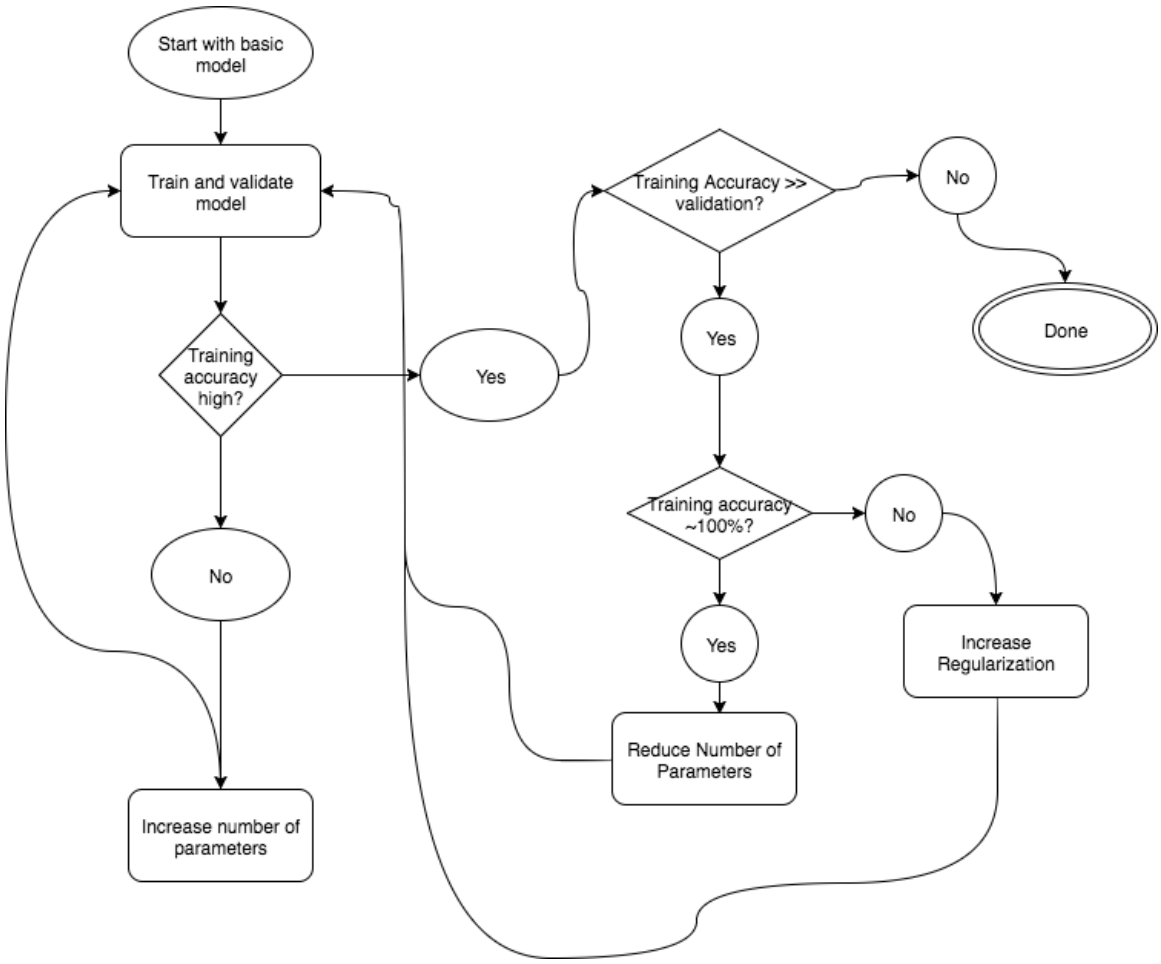

Figure 4 Flow chart of the heuristic hyperparameter search

3.2.3 Transfer Learning

Transfer learning is a technique in which a model trained on one dataset is used as the initialization for a model trained on another dataset. This technique is popular in other domains of deep learning, such as image classification. A common practice, for example, is to take a network trained on ImageNet, a dataset of over 14 million images, replace the classification layer, and retrain the network for a more specific task, such as plant leaf discrimination. This allows the early layers of the network to learn highly generalizable features from the larger dataset, and the later layers of the network to learn the specifics of the smaller dataset the model is being adapted for. While this technique is quite popular in image recognition, its use remains largely unexplored in EEG classification (Zhu et al., 2011; Oquab, Bottou, Laptev, & Sivic, 2014).

In our task, we will use the concept of transfer learning to take a model trained universally across all subjects and fine-tune it for use in the classification of an individual subject's data. More specifically we will run training in two phases. First, a universal model will be trained on all but one subject. Then, the weights obtained by the first phase will be used as an initialization for a second phase of training on the remaining subject. This provides many benefits. We have found that randomly initialized deep learning models do not reliably converge on an individual's EEG data when only a couple

hundred samples are available. Deep learning models can, however, successfully converge on several thousand samples split across multiple subjects. By first training a model over multiple subjects, we obtain an initialization that is then used for training on a single subject. This process is hypothesized to make convergence on an individual's data more likely. It should also provide stronger general filters in the early layers of the neural network. Figure 5 shows a conceptual depiction of transfer learning.

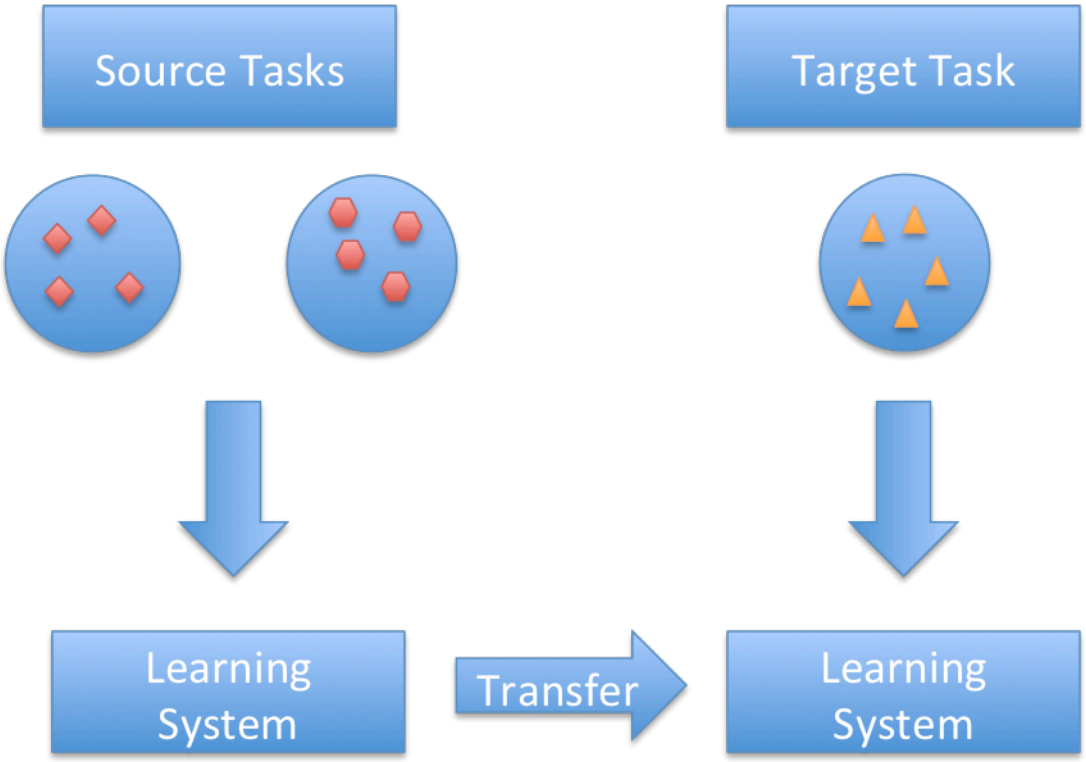

Figure 5 Transfer learning paradigm

Chapter 4

Dataset, Implementation, and Results

In this chapter we describe the dataset, the specific implementation details, and the results of the model search and transfer learning tasks.

4.1 Visual Presentation and Refresh Dataset

The dataset is composed of two parts. Initially, the subject is presented with a pair of stimuli for 1500ms: two written words, two images of faces, or two images of scenes. This stimulus period is referred to as the initial presentation. After a 500ms delay, the subject is then presented with a stimuli for 1500ms representing a refresh condition, a no act condition or an act condition. In the refresh condition, the subject is presented with an arrow pointing either up or down, to where one of the initial cues was located and the subject was instructed to think about the cue that was in that location. Both the initial presentation of the stimuli and the refresh condition are used for classification in this research.

The data was acquired on a 32 channel EEG at a 250 Hz sample rate. Signals were recorded with a bandpass filter of .01-100 Hz and a precision of 14 bits. 1.5 seconds

of seconds of recording during the viewing of the stimulus was used for classification. There were a total of 37 subjects with approximately 200 trials per subject, after artifact rejection for a total of slightly under 7000 trials. The data was labeled with three classes corresponding to whether the subject was viewing face, scene, or word data.

Figure 6 shows the response on channel 26 for a single trial of the task, corresponding the viewing of a face. Figure 7 shows the average response to the task across all trials on channel 26. This channel was chosen as it demonstrates a response consistent with the literature. While the pattern of activity in response to a visual stimulus is difficult to see in a single trial, it becomes visible after averaging many trials.

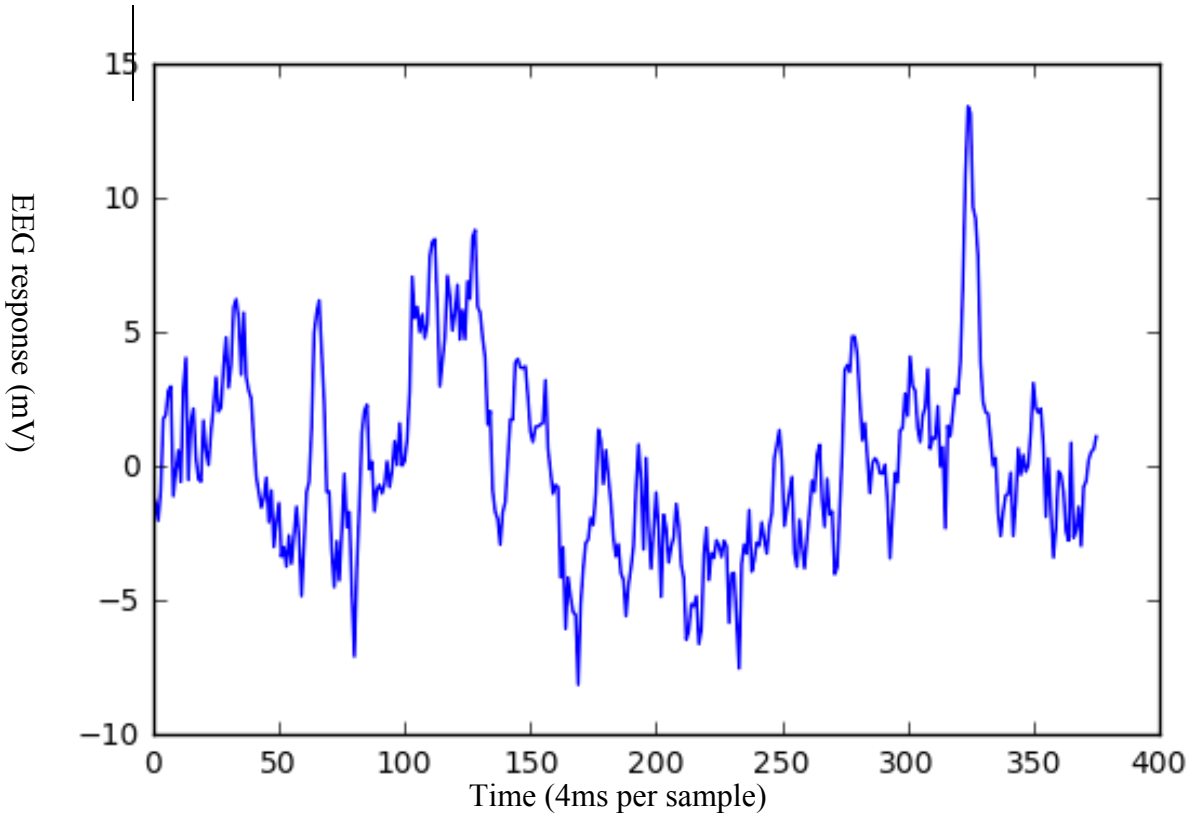

Figure 6 Graph of single trial response on Channel 26

4.2 Implementation Details

Experiments were run on servers with an Intel Xeon processor, 64 GB of RAM, and either an Nvidia 1070 or Nvidia Titan X GPU.

The primary language used was Python. In particular, the distribution used was the Anaconda scientific distribution of Python 3.5. Numpy and Scipy were used for the vast majority of numerical computations. Scikit learn was used for the machine learning algorithms, other than those related to deep learning. All deep learning related code was written using Keras, with a backend of Theano using CUDNN to allow for efficient GPU training.

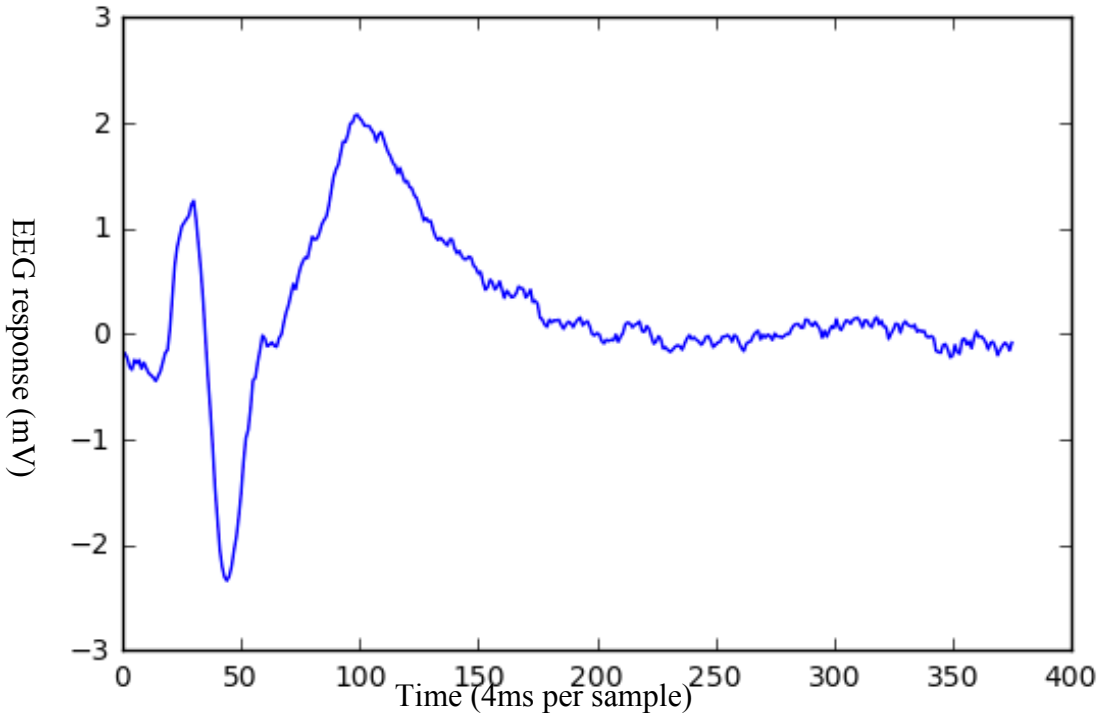

Figure 7 Average response across all trials for Channel 26

4.3 Results

The data was split in multiple ways to determine what is optimal for training. Training was performed either by training a single model per subject or training one model across all subject, corresponding to the single and universal subject training paradigms discussed in section 3.2.1. Separate models were developed for two different points in the experiments: during the initial presentation or during the refresh period. Finally, we compared no feature extraction to feature extraction with time binning using averages of 40 *ms* windows, which reduces the number of features by 90%. Time binning details

discussed in section 3.2.1. All experiments are three class classification, with chance being 33.33%.

4.3.1 Traditional Techniques

Table 1 summarizes the results of the traditional learning experiments. All values are the result of 500 rounds of random cross validation with an 80/20 train/test split.

SMLR and SVM performed very similarly across different breakdowns of the data. SMLR achieved the overall highest classification accuracy on both the initial presentation and refresh data, though only by a small margin. The highest overall initial presentation results were achieved by SMLR at 62.83%, with SVM achieving 62.02%. Similarly, the highest refresh result achieved was achieved by SMLR at 37.96%, with SVM achieving 37.59%. Thus, while SMLR comes out slightly ahead, the two methods are largely interchangeable for this data.

We found that universally trained classifiers perform significantly better for classification of the initial presentation period, but training separate classifiers for each individual led to higher performance in classifying the refresh period. This could be explained by the idea that basic physiological responses tend to generate much more similar brain activity between individuals than higher order processing. That is to say, the brain activity resulting from the initial presentation is more uniform between subjects

than the brain activity resulting from the refresh period. Thus, the increased amount of data available when training universal classifiers is able to outweigh the difficulty in classification occurred due to variance between subjects during classification of the initial presentation, but not the classification of the refresh period.

Table 1 Traditional MVPA results

Model	Subjects	Interval	Time Binned	Results
SMLR	Single	Presentation	Yes	58.69%
SMLR	Single	Presentation	No	59.02%
SMLR	Single	Refresh	Yes	**37.96%**
SMLR	Single	Refresh	No	36.36%
SMLR	Universal	Presentation	Yes	**62.83%**
SMLR	Universal	Presentation	No	57.54%
SMLR	Universal	Refresh	Yes	35.53%
SMLR	Universal	Refresh	No	34.5%
SVM	Single	Presentation	Yes	59.09%
SVM	Single	Presentation	No	57.45%
SVM	Single	Refresh	Yes	37.59%
SVM	Single	Refresh	No	37.04%
SVM	Universal	Presentation	Yes	62.02%
SVM	Universal	Presentation	No	59.51%
SVM	Universal	Refresh	Yes	35.66%
SVM	Universal	Refresh	No	35.63%

Time binning also increased accuracy across the board, with the only exception being SMLR trained per subject on the initial presentation data. This is an unsurprising

result, as SMLR and SVM methods both tend to struggle with high dimensionality, especially when the data is limited.

The initial presentation data was far easier to classify than the refresh data. There are several potential explanations. First, the brain activity caused when viewing an stimulus is more consistent than that when thinking back to a stimulus.

4.3.2 Deep Learning Model Search

In this section we first present the models determined by the model search and then the final classification results of these models in each task. The only preprocessing step was to scale the data by the 90th percentile value across all values in the training set. This scales the majority of the data to the -1 to 1 range at which deep learning methods function best, while preventing outliers from dominating the results.

The current go-to architectures in other domains, such as Inception-Net (Szegedy, Ioffe, & Vanhoucke, 2016), were quickly found to either over fit or simply not converge when used with our EEG dataset. The models were simply too complex given the small amount of data available.

There are several features that are consistent across the architectures that were found to be successful. First, the number of parameters is comparatively small compared to those used in other domains, such as image classification. The largest model used

consisted of approximately 300,000 parameters, as compared to the tens of millions that are common in published image classification architectures. Second, strong regularization is generally leads to better results. Applying L2 regularization of model weights with a large coefficient (0.1 as compared to the default coefficient in Keras of 0.01) was frequently used in the best performing models. Dropout applied to fully connected layers universally lead to an improvement in generalization.

The model search focused on performance in universal classification. The dataset only contained about 200 samples per subject, but had 27 subjects. Given the small sample size and the amount of noise in the data, even relatively small neural networks were unable to reliably converge when trained on a single subject.

The following sections will detail the final version of each class of model discussed in Section 3.2.2. The hyperparameter search heuristics detailed in Section 3.2.2 were followed for all models.

4.3.2.1 Multi-Layer Perceptron

A simple multilayer perceptron model was explored in order to set a baseline for other deep learning models. The final architecture is shown in Table 2. This model was trained using the ADAM optimizer with categorical cross-entropy as the loss function.

Variations on the architecture reported in Table 2 were explored. Increasing the number of neurons per layer to 64 or 128 resulted in very little difference in the test accuracy (<1%). Further increases in the neurons per layer led to overfitting. Similarly, increasing the number of fully connected layers showed little benefit or led to overfitting.

Table 2 Final MLP architecture

Layer	Activation Function	Parameters
Fully Connected	Leaky ReLU (.3)	32 neurons, Batch Normalization, Dropout (0.6)
Fully Connected	Leaky ReLU(.3)	32 neurons, Batch Normalization, Dropout (0.6)
Fully Connected	Leaky ReLU(.3)	32 neurons, Batch Normalization, Dropout (0.6)
Classification	Softmax	N/A

4.3.2.2 Convolutional Neural Network

Table 3 contains the final architecture chosen for the CNN based model. The convolutions were performed over the data organized as Channels by Time. A fully convolutional architecture with no fully connected layers was chosen to maintain comparability with Lawhern et al., 2016. Due to the extremely small number of parameters in a fully convolutional architecture, training is fast and overfitting is less likely.

Several features were found to lead to increased performance in the model search. Dropout is generally the most effective form of regularization in fully connected layers and recurrent layers, but was found to actually decrease accuracy when applied to convolutional layers. Since dropout was not viable for regularization in these architectuers, strong L2 regularization of the weights was found to be particularly beneficial. The model performed best with rectangular filters in which the dimension over channels was smaller than the dimension of time. Max pooling led to underfitting and was avoided for the final model.

Table 3 Final convolutional architecture

Layer	Activation Function	Parameters
2D Convolution	Leaky ReLU (.3)	12 Filters, 3x6 filter size, L2 Regularization (.1)
2D Convolution	Leaky ReLU(.3)	12 filters, 3x6 filter size, L2 Regularization (.1)
2D Convolution	Leaky ReLU(.3)	12 filters, 3x6 filter size, L2 Regularization (.1)
Classification	Softmax	

4.3.2.3 Long Short-Term Memory

Table 4 contains the final architecture for the LSTM model.

The final LSTM model had the greatest number of parameters of all the architectures selected for full testing at just under one million. Three layers with a

descending number of LSTM units followed by two fully connected layers was found to be the most effective model architecture. Reducing the number of features prior to the first fully connected layer was crucial to avoid having too many parameters and overfitting on the training data. While dropout was still beneficial in the LSTM layers, a smaller rate of 0.3 was found to be more effective, with higher values leading to poor convergence. Using the softsign activation in the LSTM layers provided better results than Leaky ReLU or tanh layers.

Table 4 Final LSTM architecture

Layer	Activation Function	Parameters
LSTM	Softsign	128 units, Dropout (0.3)
LSTM	Softsign	64 units, Dropout (0.3)
LSTM	Softsign	32 units, Dropout (0.3)
Fully Connected	Leaky ReLU(.3)	64 neurons, Dropout (0.6)
Fully Connected	Leaky ReLU(.3)	32 neurons, Dropout (0.6)
Classification	Softmax	

4.3.2.4 Integrated LSTM + CNN

Table 5 contains the final architecture for the LSTM + CNN model.

While the LSTM + CNN architecture has the most layers of any of the final model architectures, it has less parameters than the LSTM architecture. The model performed best with only a single LSTM layer, rather than using three layers as in the pure LSTM model. The structure of the convolutional and max pooling layers serves to

reduce the dimensionality of the network prior to the fully connected layer, thus leading to a drastic reduction in overall number of parameters.

The convolutional architecture is partially based on Szegedy, Ioffe, & Vanhoucke, (2016) wherein 2 dimensional convolutions were split into pairs of one dimensional convolutions. This leads to a lower number of parameters in the convolutional layers while simultaneously increasing the depth of the model. A 1x1 convolutional layer was used to separate the two convolutional passes. This is believed to provide benefits similar to a fully connected layer with a far lower increase in parameters, and led to an increase in performance over omitting the layer.

Table 5 Final LSTM + CNN architecture

Layer	Activation Function	Parameters
LSTM	Softsign	128 units, Dropout (0.3)
Convolutional	Leaky ReLU (0.3)	12 filters, 6×1 filter size, L2 Regularization (0.1)
Convolutional	Leaky ReLU (0.3)	12 filters, 1×4 filter size, L2 Regularization (0.1)
Max Pooling	N/A	3×2 window
Convolutional	Leaky ReLU(0.3)	6 filters, 1×1 filter size, L2 Regularization (0.1)
Convolutional	Leaky ReLU (0.3)	6 filters, 6×1 filter size, L2 Regularization (0.1)
Convolutional	Leaky ReLU (0.3)	6 filters, 1×4 filter size, L2 Regularization (0.1)
Max Pooling	N/A	3×2 window
Fully Connected	Leaky ReLU (0.3)	64 neurons, Dropout (0.6)
Classification	Softmax	

4.3.3 Deep Learning Classification Results

Table 6 summarizes the performance of the deep learning models selected during model search. Since none of the models were able to converge reliably on single subject data, those results are omitted. All results are calculated through 10 fold cross-validation with a 60/20/20 training/validation/test split.

- Time binning significantly decreased the performance of all deep learning models. In the most extreme case, the LSTM + CNN model, time binning lowered the accuracy from 64.36% to 46.89%. Only the LSTM model was able to perform above chance and on-par with the traditional methods on the refresh dataset.

- The MLP model performed poorly in all tasks – significantly lower than the traditional methods. As discussed in Chapter 3, MLP models are not invariant to temporal or spatial shifts in the data, and thus poor performance was expected when training across subjects.
- The CNN model performed on par with the most successful traditional methods, with an initial presentation accuracy of 62.09%. Of the deep learning methods, it was by far the quickest to train at around 45 minutes to converge due to the small parameter amount.
- The LSTM model performed slightly better than the traditional methods on the initial presentation data, with an accuracy of 63.61%. It was the slowest model to train, taking 4 to 5 hours.
- The LSTM + CNN model performed the best of any traditional or deep learning model, with an accuracy of 64.36%.

Table 6 Deep learning classification accuracy

Model	Subjects	Interval	Time Binned	Results
MLP	Universal	Presentation	Yes	54.29%
MLP	Universal	Presentation	No	55.69%
MLP	Universal	Refresh	Yes	34.33%
MLP	Universal	Refresh	No	35.06%
CNN	Universal	Presentation	Yes	60.04%
CNN	Universal	Presentation	No	62.09%
CNN	Universal	Refresh	Yes	34.75%
CNN	Universal	Refresh	No	35.63%
LSTM	Universal	Presentation	Yes	58.44%
LSTM	Universal	Presentation	No	63.61%
LSTM	Universal	Refresh	Yes	35.91%
LSTM	Universal	Refresh	No	**37.72%**
LSTM + CNN	Universal	Presentation	Yes	46.89%
LSTM + CNN	Universal	Presentation	No	**64.36%**
LSTM + CNN	Universal	Refresh	Yes	32.65%
LSTM + CNN	Universal	Refresh	No	34.30%

4.3.4 Transfer Learning

Table 7 summarizes the effects of transfer learning on classification of initial presentation data. The procedure for transfer learning was as follows:

1. Select all but one subject.

2. Train a model with the same procedure described in Section 4.3.3, with the alteration of using only training and validation datasets in an 80/20 split.
3. Use the learned weights to initialize a new model.
4. Train the new model on the subject originally held out, with a 60/20/20 training/validation/testing split.
5. Repeat 3 times per subject for cross-validation.

While transfer learning showed some promise in the classification of refresh data, it was highly volatile and requires further investigation before reporting. The number of folds was limited to 3 due to the large time required to train per transfer learned model.

The fine-tuning approach to transfer learning explored here led to an increase in accuracy of all deep learning models. After fine-tuning, even the MLP model performed slightly better than the traditional methods, with an increase in accuracy of 6.6%. The CNN model showed only a small increase of 3.3%, but that was enough to lead to clearly stronger performance than traditional methods. The LSTM model's accuracy increased by 5.8% to 69.4% with the application of fine-tuning. This is the highest accuracy of all models explored in this experiment. The LSTM+CNN model gained only 2.8% accuracy from the use of fine-tuning.

The models with convolutional layers showed only a small increase in accuracy from the use of fine-tuning. This is potentially because the convolutional layers are already highly regularized due to weight-sharing, so the benefit from training universally before moving to a single subject is minimized. Furthermore, they tended to have less parameters, and thus potentially less representative power.

A final important note is that all models showed a high within-subject variance across the three folds. This suggests that the fine-tuning phase may have led to over-fitting.

Table 7 Transfer learning results

Model	Universal Accuracy	Transfer Accuracy	Increase
MLP	55.7%	62.3%	6.6%
CNN	62.1%	64.4%	3.3%
LSTM	63.6%	69.4%	5.8%
LSTM + CNN	64.4%	67.2%	2.8%

Chapter 5

Summary and Future Work

5.1 Summary

Traditional methods perform best after the application of dimensionality reduction, and, at least during visual perception tasks, when trained across subjects. LSTM based deep learning models are able to out perform traditional methods. With the application of transfer learning, all deep learning models tested were superior to traditional techniques, and the gap between LSTM methods and traditional techniques widened considerably.

It is currently difficult to exceed the performance of traditional machine learning techniques with deep learning techniques. It took many iterations of model architectures and hyperparameters, along with weeks of computing time to find architectures that outperformed the traditional results initially. However, with the continued exploration of deep learning in the classification of EEG and better guidelines, this time could be drastically reduced. Furthermore, the benefits provided by transfer learning may be significant enough to allow classification of problems traditional techniques fail on.

5.2 Future Work

There are three main avenues for the extension of this research: application to more datasets, expanded model search, and other forms of transfer learning.

5.2.1 More Datasets

Currently, these techniques were only applied to a single dataset. In order to show generalizability, it is desirable to replicate this research on further datasets with different properties.

The dataset in this thesis involved the classification of visual stimuli. While this is a fairly common task in EEG classification (e.g., Stewart, Nuthmann, & Sanguinetti, 2014; M. Johnson & Johnson, 2010), there are several other common tasks that should also be explored. Motor imagery is perhaps the most common task in EEG classification, and is used in several forms of brain computer interface (e.g., Tabar & Halici, 2017; Battapady, Lin, Fei, Huang, & Bai, 2009; Miranda et al., 2015). Due to the ease of classification and potential upside in this task, it is a valuable extension. Similarly, experiments where the subject thinks about words from a set vocabulary have immediate potential use in communication based BCIs, and would also show a valuable extension of this research. Finally, extending this work to datasets of any sort of stimuli that have been difficult to classify using traditional techniques is important in establishing the benefit of these techniques.

The dataset explored in this thesis was collected on a 32-channel EEG. Many modern EEGS have 128 or even 256 channels, leading to several times the number of

features. The negative impact of feature reduction on deep learning found in our results suggests that having a higher number of recorded features may improve the classifiability of the signal, despite the issues with the curse of dimensionality. Thus, it is important to examine the performance of these techniques on data collected with different hardware.

5.2.2 Expanded Model Search

While the work presented in this thesis covered many permutations of four basic types of models, there are other architectures that are promising.

The first category of models that show promise are those based on a different representation of the data. While strictly recurrent architectures showed the highest classification accuracy of the models currently explored, this may be due to the information lost by compressing the channels to a single dimension. By using a 3D projection of the data in *X by Y by time* format, that spatial information may be largely retained. Many architectures used in the classification of video may be valuable in these situations. This includes 3D convolutional models, which aggregate information of a localized area over time. 2D convolutional models over space fed into recurrent architectures over time are also promising(Karpathy et al., 2014). Furthermore, representing the data in this format may allow for the use of data augmentation

techniques used in image classification and audio classification, such as shearing and stretching (Schlüter & Grill, 2015).

There are also other layer types that should be considered. For example, Gated Recurrent Units (GRUs) have been gaining favor over LSTM models in other time series data, such as natural language processing (Cho et al., 2014).

5.2.3 Transfer Learning Techniques

Transfer learning is a family of techniques involving the use of one set of data to create an initialization for the classification of another set of data. Currently, we have only explored the training across all subjects in an experiment, then fine-tuning the entire network on an individual in the network. There are many other transfer learning paradigms that may show strong performance.

The most straightforward change that could be made in the transfer learning paradigm is simply freezing the bottom layers of the network during fine-tuning. That is to say, only allow the classification layer and perhaps the last fully connected layer to update during the fine-tuning phase. The benefit of this approach is that it may reduce overfitting during the fine-tuning phase, which is likely a problem with our current approach given the high variance of the classification accuracies within subjects.

Another approach to transfer learning that is worth exploring is transferring between datasets with similar stimuli. There are numerous known common features in EEG data, such as the P300 response in visual stimuli(Polich, 2007). Thus, it is reasonable to believe that applying transfer learning over several datasets to learn universal basic features for the early filters in the network could be beneficial. It may also allow networks with more parameters, and thus greater representative power to be trained than could be achieved with a single dataset. There are some barriers that would need to be considered, however, especially if the data is collected on different machines or has a different time span.

5.3 Conclusion

The use of deep learning in EEG classification is still in its early stages. Currently, it is difficult to find architectures and training paradigms that result in improvements over traditional methods. However, as research continues into the use of deep learning for classification of EEG signals, best practices will become well known. The results of our experiments show a strong case for the heavy use of regularization through both direct L2 methods and dropout, the strength of recurrent models, and the powerful benefits of universal to single subject transfer learning.

Bibliography

AlZoubi, O., Calvo, R. a., & Stevens, R. H. (2009). Classification of EEG for affect recognition: An adaptive approach. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *5866 LNAI*, 52–61. http://doi.org/10.1007/978-3-642-10439-8_6

Bashivan, P., Rish, I., Yeasin, M., & Codella, N. (2015). Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks, 1–15. Retrieved from http://arxiv.org/abs/1511.06448

Battapady, H., Lin, P., Fei, D.-Y., Huang, D., & Bai, O. (2009). Single trial detection of human movement intentions from SAM-filtered MEG signals for a high performance two-dimensional BCI. *Conference Proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, *2009*, 524–7. http://doi.org/10.1109/IEMBS.2009.5333632

Bottou, L. (2010). Large-Scale Machine Learning with Stochastic Gradient Descent. *Proceedings of COMPSTAT'2010*, 177–186. http://doi.org/10.1007/978-3-7908-2604-3_16

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734. http://doi.org/10.3115/v1/D14-1179

Correia, J. M., Jansma, B., Hausfeld, L., Kikkert, S., & Bonte, M. (2015). EEG decoding of spoken words in bilingual listeners: From words to language invariant semantic-conceptual representations. *Frontiers in Psychology*, *6*(FEB), 1–10. http://doi.org/10.3389/fpsyg.2015.00071

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297. http://doi.org/10.1023/A:1022627411411

Fair, D. A., Bathula, D., Nikolas, M. A., & Nigg, J. T. (2012). Distinct neuropsychological subgroups in typically developing youth inform heterogeneity in children with ADHD. *Proceedings of the National Academy of Sciences*, *109*(17), 6769–6774. http://doi.org/10.1073/pnas.1115365109

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological*

Cybernetics, *36*(4), 193–202. http://doi.org/10.1007/BF00344251

Gabor, A. J., Leach, R. R., & Dowla, F. U. (1996). Automated seizure detection using a self-organizing neural network. *Electroencephalography and Clinical Neurophysiology*, *99*(3), 257–266. http://doi.org/10.1016/S0921-884X(96)96001-4

Gao, J., Wang, Z., Yang, Y., Zhang, W., Tao, C., Guan, J., & Rao, N. (2013). A Novel Approach for Lie Detection Based on F-Score and Extreme Learning Machine. *PLoS ONE*, *8*(6), e64704. http://doi.org/10.1371/journal.pone.0064704

Gu, Y., Poesio, M., & Murphy, B. (2014). EEG study of the cortical representation and classification of the emotional connotations in words. *BMC Neuroscience*, *15*(Suppl 1), P81. http://doi.org/10.1186/1471-2202-15-S1-P81

Guan, C. G. C., Thulasidas, M., & Wu, J. W. J. (2004). High performance P300 speller for brain-computer interface. *IEEE International Workshop on Biomedical Circuits and Systems, 2004.*, 13–16. http://doi.org/10.1109/BIOCAS.2004.1454155

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, *18*(7), 1527–1554. http://doi.org/10.1162/neco.2006.18.7.1527

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. http://doi.org/10.1162/neco.1997.9.8.1735

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*(8), 2554–2558. http://doi.org/10.1073/pnas.79.8.2554

Huang, M., Wu, P., Liu, Y., Bi, L., & Chen, H. (2008). Application and contrast in brain-computer interface Between hilbert-huang transform and wavelet transform. *Proceedings of the 9th International Conference for Young Computer Scientists, ICYCS 2008*, (1), 1706–1710. http://doi.org/10.1109/ICYCS.2008.537

Huang, N. E., & Wu, Z. (2008). a Review on Hilbert-Huang Transform : Method and Its Applications. *October*, *46*(2007), 1–23. http://doi.org/10.1029/2007RG000228.1.INTRODUCTION

Jarrett, K., Kavukcuoglu, K., Ranzato, M., & LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2146–2153). http://doi.org/10.1109/ICCV.2009.5459469

Johnson, M., & Johnson, M. (2010). Decoding individual natural scene representations during perception and imagery. *Journal of Vision*, *10*(7), 780–

780. http://doi.org/10.1167/10.7.780

Johnson, M. R., McCarthy, G., Muller, K. A., Brudner, S. N., & Johnson, M. K. (2015). Electrophysiological Correlates of Refreshing: Event-related Potentials Associated with Directing Reflective Attention to Face, Scene, or Word Representations. *Journal of Cognitive Neuroscience*, *27*(9), 1823–1839. http://doi.org/10.1162/jocn

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Li, F. F. (2014). Large-scale video classification with convolutional neural networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1725–1732. http://doi.org/10.1109/CVPR.2014.223

Kevric, J., & Subasi, A. (2017). Comparison of signal decomposition methods in classification of EEG signals for motor-imagery BCI system. *Biomedical Signal Processing and Control*, *31*, 398–406. http://doi.org/10.1016/j.bspc.2016.09.007

Kingma, D. P., & Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations 2015*, 1–15.

Koutsouleris, N., Meisenzahl, E. M., Borgwardt, S., Riecher-Rössler, A., Frodl, T., Kambeitz, J., … Davatzikos, C. (2015). Individualized differential diagnosis of schizophrenia and mood disorders using neuroanatomical biomarkers. *Brain*, *138*(7), 2059–2073. http://doi.org/10.1093/brain/awv111

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, 1–9. http://doi.org/http://dx.doi.org/10.1016/j.protcy.2014.09.007

Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P., & Lance, B. J. (2016). EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces, 1–20.

Li, M., & Lu, B. L. (2009). Emotion classification based on gamma-band EEG. *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society: Engineering the Future of Biomedicine, EMBC 2009*, 1323–1326. http://doi.org/10.1109/IEMBS.2009.5334139

Linnainmaa, S. (1970). Alogritmin kumulatiivinen pyöristysvirhe yksittäisten pyöristysvirheiden Taylor-kehitelmänä.

Lotte, F., Congedo, M., Lécuyer, a, Lamarche, F., & Arnaldi, B. (2007). A review of classification algorithms for EEG-based brain-computer interfaces. *Journal of Neural Engineering*, *4*(2), R1–R13. http://doi.org/10.1088/1741-

2560/4/2/R01

McCulloch, W. S., & Pitts, W. (1943). A Logical Calculus of the Idea Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, *5*, 115–133. http://doi.org/10.1007/BF02478259

McFarland, D. J., & Wolpaw, J. R. (2008). Brain-computer interface operation of robotic and prosthetic devices. *Computer*, *41*(10), 52–56. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4640662

Minsky, M., & Papert, S. (1969). *Perceptron*. Oxford, England: M.I.T. Press.

Miranda, R. A., Casebeer, W. D., Hein, A. M., Judy, J. W., Krotkov, E. P., Laabs, T. L., … Ling, G. S. F. (2015). DARPA-funded efforts in the development of novel brain–computer interface technologies. *Journal of Neuroscience Methods*, *244*, 52–67. http://doi.org/10.1016/j.jneumeth.2014.07.019

Molina, G. G., Nijholt, A., & Twente, U. (2009). Emotional Brain-Computer Interfaces, 138–146.

Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1717–1724. http://doi.org/10.1109/CVPR.2014.222

Pierce, J. R. (1969). Whither Speech Recognition? *The Journal of the Acoustical Society of America*, *46*, 1049. http://doi.org/10.1121/1.1911801

Polich, J. (2007). Updating P300: An Integrative Theory of P3a and P3b. *Clinical Neurophysiology*, *118*(10), 2128–2148. http://doi.org/10.1016/j.clinph.2007.04.019.Updating

Prasad, S. C., & Prasad, P. (2014). Deep Recurrent Neural Networks for Time Series Prediction, *95070*, 1–54. Retrieved from http://arxiv.org/abs/1407.5949

Raina, R., Madhavan, A., & Ng, A. Y. (2009). Large-scale deep unsupervised learning using graphics processors. *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, 1–8. http://doi.org/10.1145/1553374.1553486

Ramgopal, S., Thome-Souza, S., Jackson, M., Kadish, N. E., S??nchez Fern??ndez, I., Klehm, J., … Loddenkemper, T. (2014). Seizure detection, seizure prediction, and closed-loop warning systems in epilepsy. *Epilepsy and Behavior*, *37*, 291–307. http://doi.org/10.1016/j.yebeh.2014.06.023

Rebsamen, B., Kwok, K., & Penney, T. B. (2011). Evaluation Of Cognitive Workload From EEG During A Mental Arithmetic Task. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *55*, 1342–1345.

http://doi.org/10.1177/1071181311551279

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. http://doi.org/10.1037/h0042519

Schlüter, J., & Grill, T. (2015). Exploring data augmentation for improved singing voice detection with neural networks. *Proceedings of the 16th International Society for Music Information Retrieval Conference*, 121–126. Retrieved from http://www.ofai.at/~jan.schlueter/pubs/2015_ismir.pdf

Schulz, E., Zherdin, A., Tiemann, L., Plant, C., & Ploner, M. (2012). Decoding an individual's sensitivity to pain from the multivariate analysis of EEG data. *Cerebral Cortex*, *22*(5), 1118–1123. http://doi.org/10.1093/cercor/bhr186

Shirer, W. R., Ryali, S., Rykhlevskaia, E., Menon, V., & Greicius, M. D. (2012). Decoding subject-driven cognitive states with whole-brain connectivity patterns. *Cerebral Cortex*, *22*(1), 158–165. http://doi.org/10.1093/cercor/bhr099

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ImageNet Challenge*, 1–10. http://doi.org/10.1016/j.infsof.2008.09.005

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*, 1929–1958. http://doi.org/10.1214/12-AOS1000

Stewart, A. X., Nuthmann, A., & Sanguinetti, G. (2014). Single-trial classification of EEG in a visual object task using ICA and machine learning. *Journal of Neuroscience Methods*, *228*, 1–14. http://doi.org/10.1016/j.jneumeth.2014.02.014

Stober, S., Sternin, A., Owen, A. M., & Grahn, J. A. (2015). Deep Feature Learning for EEG Recordings. *Arxiv*, 1–24. Retrieved from http://arxiv.org/abs/1511.04306

Sutskever, I., Martens, J., Dahl, G. E., & Hinton, G. E. (2013). On the importance of initialization and momentum in deep learning. *Jmlr W&Cp*, *28*(2010), 1139–1147. http://doi.org/10.1109/ICASSP.2013.6639346

Szegedy, C., Ioffe, S., & Vanhoucke, V. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Arxiv*, 12.

Tabar, Y. R., & Halici, U. (2017). A novel deep learning approach for classification of EEG motor imagery signals. *Journal of Neural Engineering*, *14*(1), 16003. http://doi.org/10.1088/1741-2560/14/1/016003

Trambaiolli, L. R., Lorena, A. C., Fraga, F. J., Kanda, P. a M., Anghinah, R., & Nitrini, R.

(2011). Improving Alzheimer's disease diagnosis with machine learning techniques. *Clinical EEG and Neuroscience : Official Journal of the EEG and Clinical Neuroscience Society (ENCS)*, *42*(3), 160–165. http://doi.org/10.1177/155005941104200304

Tripp, S., & Grueber, M. (2011). Economic Impact of the Human Genome Project. *Battelle Memorial Institute*, 58. http://doi.org/10.1017/CBO9781107415324.004

Unser, M., & Aldroubi, W. (1996). of Wavelets i Biomedical Ap. *Review Literature And Arts Of The Americas*, *84*(4).

Wang, X.-W., Nie, D., & Lu, B.-L. (2014). Emotional state classification from EEG data using machine learning approach. *Neurocomputing*, *129*, 94–106. http://doi.org/10.1016/j.neucom.2013.06.046

Wang, X., Zhang, T., Chaim, T. M., Zanetti, M. V, & Davatzikos, C. (2015). Classification of MRI under the Presence of Disease Heterogeneity using Multi-Task Learning: Application to Bipolar Disorder. In N. Navab, J. Hornegger, W. M. Wells, & A. Frangi (Eds.), *Medical Image Computing and Computer-Assisted Intervention -- MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part I* (pp. 125–132). Cham: Springer International Publishing. http://doi.org/10.1007/978-3-319-24553-9_16

Zaremba, W., Sutskever, I., & Vinyals, O. (2014). Recurrent Neural Network Regularization. *arXiv:1409.2329 [Cs]*, (2013), 1–8. Retrieved from http://arxiv.org/abs/1409.2329%5Cnhttp://www.arxiv.org/pdf/1409.2329.pdf

Zhang, L., Samaras, D., Tomasi, D., Volkow, N., & Goldstein, R. (2005). Machine learning for clinical diagnosis from functional magnetic resonance imaging. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, *1*, 1211–1217 vol. 1. http://doi.org/10.1109/CVPR.2005.219

Zhu, Y., Chen, Y., Lu, Z., Pan, S. J., Xue, G.-R., Yu, Y., & Yang, Q. (2011). Heterogeneous Transfer Learning for Image Classification. *Proceedings of the 25th AAAI Conference on Artificial Intelligence, AAAI*, (January 2014), 1304–1309. http://doi.org/10.1007/s00247-003-1118-z