MULTI-SUBJECT BRAIN DECODING USING DEEP LEARNING TECHNIQUES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BURAK VELIOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

AUGUST 2016

Approval of the thesis:

MULTI-SUBJECT BRAIN DECODING USING DEEP LEARNING TECHNIQUES

submitted by **BURAK VELIOĞLU** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Adnan Yazıcı
Head of Department, **Computer Engineering** _____

Prof. Dr. Fatoş T. Yarman Vural
Supervisor, **Computer Engineering Department, METU** _____

Assist. Prof. Dr. Şeyda Ertekin Bolelli
Co-supervisor, **Computer Engineering Department, METU** _____

Examining Committee Members:

Prof. Dr. Göktürk Üçoluk
Computer Engineering Department, METU _____

Prof. Dr. Fatoş T. Yarman Vural
Computer Engineering Department, METU _____

Assoc. Prof. Dr. Sinan Kalkan
Computer Engineering Department, METU _____

Assist. Prof. Dr. Yusuf Sahillioğlu
Computer Engineering Department, METU _____

Assoc. Prof. Dr. Pınar Duygulu Şahin
Computer Engineering Department, Hacettepe University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: BURAK VELIOĞLU

Signature :

ABSTRACT

MULTI-SUBJECT BRAIN DECODING USING DEEP LEARNING TECHNIQUES

Velioğlu, Burak
M.S., Department of Computer Engineering
Supervisor : Prof. Dr. Fatoş T. Yarman Vural
Co-Supervisor : Assist. Prof. Dr. Şeyda Ertekin Bolelli

August 2016, 87 pages

In this study, a new method is proposed for analyzing and classifying images obtained by functional magnetic resonance imaging (fMRI) from multiple subjects. Considering the multi-level structure of the brain and success of deep learning architectures on extracting hierarchical representations from raw data, these architectures are used in this thesis. Initially, the S500 data set collected in the scope of Human Connectome Project (HCP) is used to train formed deep neural networks in an unsupervised fashion. Then, pre-trained networks are utilized for two different multi-subject brain decoding tasks. Goal of these tasks is discriminating the cognitive state of a subject, using the fMRI data of other subjects. In the first task, brain decoding is performed by fine-tuning the pre-trained neural networks with the label information included in the S500 data set. In the second task, pre-trained networks are used to obtain hierarchical representation for object recognition data set to transfer the information between fMRI experiments. Obtained results show us that deep neural networks are more successful than traditional machine learning algorithms on multi-subject brain decoding tasks and experiment-independent representations can be obtained with deep neural networks better than factor models used in the literature. Besides, regions activated for different cognitive states of S500 data set are visualized by implementing a saliency analysis over trained deep neural networks.

Keywords: Multi-subject brain decoding, fMRI, deep neural networks, transfer learning, neuroscience

ÖZ

DERİN ÖĞRENME YÖNTEMLERİNİ KULLANARAK ÇOK DENEKLİ BEYİN OKUMA

Velioğlu, Burak
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Fatoş T. Yarman Vural
Ortak Tez Yöneticisi : Yrd. Doç. Dr. Şeyda Ertekin Bolelli

Ağustos 2016 , 87 sayfa

Bu çalışmada çok-denekli fonksiyonel manyetik rezonans görüntüleme (fMRG) yöntemi ile elde edilen görüntüler için yeni bir analiz ve sınıflandırma metodu önerilmiştir. Beynin çok katmanlı yapısı ve derin öğrenme mimarilerinin ham veriden çok katmanlı gösterim çıkarma konusundaki başarısı göz önünde bulundurularak, tez dahilinde derin öğrenme mimarileri kullanılmıştır. İlk adım olarak Human Connectome Project (HCP) projesi kapsamında toplanan S500 veri seti kullanılarak oluşturulan derin sinir ağları gözetimsiz bir şekilde eğitilmiştir. Ön-eğitimden geçen bu ağlar tez dahilinde iki farklı çok-denekli beyin okuma deneyinde kullanılmıştır. İki deneyin de amacı kişilerin bilişsel durumlarını diğer deneklerin fMRG verilerini kullanarak sınıflandırmaktır. Birinci deneyde, ön eğitimden geçen derin sinir ağları S500 veri setinde bulunan sınıf bilgileri kullanılarak gözetimli şekilde eğitilmiş ve çok denekli beyin okuma tesi gerçekleştirilmiştir. İkinci deneyde, ön eğitimden geçen sinir ağlarından elde edilen çok katmanlı yapı kullanılarak bir başka fMRG deneyi olan obje tanıma deneyi için gösterimler elde edilmiştir. Elde edilen sonuçlar derin sinir ağlarının geleneksel makine öğrenme yöntemlerine göre çok-denekli beyin okuma deneyinde ve deneyden bağımsız gösterim elde etme konusunda daha başarılı olduğunu göstermektedir. Elde edilen bu sonuçların yanısıra, S500 deneyindeki farklı bilişsel süreçler esnasında aktif olan beyin bölgeleri derin sinir ağları ile belirlenerek görselleştirilmiştir.

Anahtar Kelimeler: Çok-denekli beyin okuma, fMRG, derin sinir ağları, transfer öğrenme, sinirbilimw

To my family...

ACKNOWLEDGMENTS

I would like to express my best appreciations to my supervisor Prof. Dr. Fatoş Tünay Yarman Vural. She does not only guide me on my academical journey but also reformed my character like an artist. This work can not be completed without her wisdom and vision. It is both an honor and chance for me being a student of her.

I would like to thank my thesis committee members, Prof. Dr. Göktürk Üçoluk, Assoc. Prof. Dr. Sinan Kalkan, Assoc. Prof. Dr. Pınar Duygulu Şahin and Asst. Prof. Dr. Yusuf Sahillioğlu for their valuable feedback on this thesis.

I would like to thank my family for their warm support. They are always thoughtful towards me. All the efforts they have made are priceless. My mother, Şükran and my father, Ali always understand my distresses and get me out of a lots of jams. My brother, Mehmet Ali is not only my brother but also one of my best friends. Being a younger brother of him makes the life nicer and safer. Without him I could not find my way in the life. I also thank Melike for all her supports.

Many ideas in this thesis are inspired from the ideas of Dr. Mete Özay and Orhan Fırat. They have taught me the ethical and practical details of the academic life. Without the help of Itır, the secret director of our lab, I would not have solved the theoretical problems I have faced with. Working with her has improved me a lot.

I would like to thank my friends Ali Mert, Arman, Barış, Batuhan, Emre, Güneş, Hazal, Ozan and Sarper from the ImageLab for such a warm environment. It was nice working with you.

I would like to thank my friends of many years Batu, Murat "The Presidente" and Yılmaz. You are the guys whom I call when I feel bad. Wherever I will live, I know you are with me.

I would like to thank Mert, Çağkan and Oğuz especially for the crazy Friday nights. Being a friend of you is really funny guys.

Last but not least, my dearest appreciations go to my honey Gülsüm. It's the greatest chance of my life to meet you. I always feel calm and happy with you. All the challenges, efforts and successes are more meaningful with you. I hope to add more and more notes about you to my records. Without your warm supports I can't finish this thesis.

I acknowledge the support of TÜBİTAK (The Scientific and Technological Research

Council of Turkey). This thesis is supported on the scope of the projects 112E315 and 114E045.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xii

LIST OF TABLES xvi

LIST OF FIGURES xviii

LIST OF ALGORITHMS xxi

LIST OF ABBREVIATIONS xxii

CHAPTERS

1 INTRODUCTION 1

1.1 Problem Definition 1

1.2 Proposed Method 2

1.3 Contribution 3

1.4 Thesis Outline 4

2 MACHINE LEARNING FOR BRAIN DECODING BY FMRI DATA 5

2.1 fMRI Data and Brain Decoding 5

2.2 fMRI Experiment Design and Pre-processing 9
2.3 Machine Learning for Brain Decoding 10
2.3.1 Dimensionality Reduction Techniques for fMRI data 11
2.3.2 Feature Extraction for fMRI 13
2.3.3 Classifiers Applied on fMRI Data 14
2.4 Chapter Summary 16
3 BRAIN DECODING BY DEEP LEARNING 19
3.1 Biological Basis of Neural Networks 19
3.2 How Does Artificial Neural Network Work ? 20
3.2.1 Forward Propagation 22
3.2.2 Backward Propagation 23
3.3 Deep Learning 26
3.3.1 Why deep architectures ? 26
3.3.2 Unsupervised Networks 27
3.3.2.1 Restricted Boltzmann Machines 27
3.3.2.2 Single Layer Autoencoders 28
3.4 Deep Neural Decoding 28
3.4.1 Stacked Autoencoders : Layer-by-layer training and fine-tuning 29
3.4.2 Constraints applied over unsupervised networks . . 31
3.4.3 Details of Activation Functions 32
3.4.4 Regularization Methods 33

3.4.4.1 Parameter Norm Penalties 34
3.4.4.2 Dropout 35
3.4.4.3 Early Stopping 36
3.4.4.4 Batch Normalization 37
3.4.5 The Optimization Techniques for SDAE 38
3.4.6 Visualizing the obtained parameters 40
3.4.7 Input Representations for Deep Learning 41
3.4.7.1 Average ROI Intensity Values 41
3.4.7.2 Pearson Correlation Values 43
3.4.7.3 Temporal Mesh Model 44
3.4.8 Two Major Goals of the Thesis 45
3.5 Chapter Summary 47
4 EXPERIMENTAL RESULTS 49
4.1 Experiments and Data Sets 49
4.1.1 Object Recognition Data Set 50
4.1.2 Human Connectome Project HCP500 Data Set . . 50
4.2 Brain Decoding on HCP500 Data Set 52
4.2.1 Decoding Raw Data 52
4.3 Including Connectivity Information 53
4.3.1 Decoding Correlation Data 53
4.3.2 Decoding the Mesh Model Weights 59

4.4 Transfer Learning with HCP500 and Object Recognition Data Set 63

4.4.1 Chapter Summary 69

5 CONCLUSION AND FUTURE WORK 77

5.1 A Brief Summary 77

5.2 Future Work 79

REFERENCES 81

LIST OF TABLES

TABLES

Table 4.1 Number of samples for each type of stimulus. 51

Table 4.2 Multi-Subject Brain Decoding Using 3D Images of HCP S500 data set. [1] 52

Table 4.3 Decoding the Cognitive task of S500 Data Set by Using Correlation Data With and Without Dropout. 57

Table 4.4 Decoding the Cognitive Tasks Using Pearson Correlation Input Vector With Varying Subject Counts and Window Sizes on HCP500 Data Set. 59

Table 4.5 Decoding Temporal Mesh Weights with number of architectures and classifiers. 60

Table 4.6 SVM classification results of Pearson correlation inputs with different window sizes of object recognition data set. 66

Table 4.7 SVM classification results using the hierarchical features for the Pearson correlation inputs with $U = 4$ on object recognition data set. . . . 66

Table 4.8 SVM classification results using the hierarchical features for the Pearson correlation inputs with $U = 5$ on object recognition data set. . . . 66

Table 4.9 SVM classification results using the hierarchical features for the Pearson correlation inputs with $U = 6$ on object recognition data set. . . . 67

Table 4.10 SVM classification results using the representation obtained with PCA when window size $U = 4$. 67

Table 4.11 SVM classification results using the representation obtained with PCA when window size $U = 5$. 67

Table 4.12 SVM classification results using the representation obtained with PCA when window size $U = 6$. 68

Table 4.13 SVM classification results using the representation obtained with ICA when window size $U = 4$. 68

Table 4.14 SVM classification results using the representation obtained with ICA when window size $U = 5$. 68

Table 4.15 SVM classification results using the representation obtained with ICA when window size $U = 6$. 68

Table 4.16 Name of Important ROIs for the emotion task 70

Table 4.17 Name of Important ROIs for the gambling task 71

Table 4.18 Name of Important ROIs for the language task 72

Table 4.19 Name of Important ROIs for the motor task 73

Table 4.20 Name of Important ROIs for the relational task 74

Table 4.21 Name of Important ROIs for the social task 75

Table 4.22 Name of Important ROIs for the working memory task 76

LIST OF FIGURES

FIGURES

Figure 2.1 Sample BOLD contrast. [1] 6

Figure 2.2 Figure a and b represent block and event-related designs, respectively. First signals of both a and b represent the stimuli function. As explained in the Figure 2.1 signal represented with green color is HDR function. Theoretical experiment signal is obtained by convolving HDR function with stimuli function. [2] 7

Figure 2.3 Obtained signal of a single voxel, obtained signals of all voxels and stimuli indices for each stimuli are given as a, b, and , respectively. 8

Figure 2.4 Swiss-roll data set. 12

Figure 2.5 LOF 16

Figure 3.1 First drawing of Purkinje Cell by Cajal in 1952. 19

Figure 3.2 Perceptron classifier. 21

Figure 3.3 Two-layered neural network. 22

Figure 3.4 Architecture of 3-layered autoencoder. 29

Figure 3.5 Multi-layer neural network classifier. 30

Figure 3.6 Behavior of a sample unit exposed to dropout at training and testing phase[3]. 36

Figure 3.7 Sliding window representation with window size U 43

Figure 3.8 Block Diagram of the overall brain decoding framework suggested in this thesis. First, voxel intensity values are mapped into three different representations. Then, by using these representations as an input to deep neural networks, multi-subject brain decoding results are obtained. Besides, pre-trained deep neural network with S500 data set is also used for transfer learning. To implement this, pearson correlation values extracted from object recognition data set fed into the pre-trained neural network. Then cognitive states of object recognition task are classified using hierarchical representations obtained from this network. Details of the S500 and object recognition data sets are explained in Chapter 4. 46

Figure 4.1 Histogram of activation function of 3^{rd} layer hidden units with sigmoid activation function. Input is the 4753 correlation values obtained between the pairs of 98 ROIs of HCP S500 data set. 55

Figure 4.2 Histogram of activation function of 2^{nd} layer hidden units with PReLU activation function. Input is the 4753 correlation values obtained between the pairs of 98 ROIs of HCP S500 data set. 56

Figure 4.3 Histogram of activation function of 2^{nd} layer hidden units with PReLU activation function and Batch Normalization. Input is the 4753 correlation values obtained between the pairs of 98 ROIs of HCP S500 data set. 57

Figure 4.4 3-Layered Deep Neural Network Architecture. 58

Figure 4.5 50 mesh arc weights with highest absolute values for WM stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for WM stimulus are found by doing a saliency analysis over pre-trained neural network. 61

Figure 4.6 50 mesh arc weights with highest absolute values for GB stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for GB stimulus are found by doing a saliency analysis over pre-trained neural network. 61

Figure 4.7 50 mesh arc weights with highest absolute values for MT stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for MT stimulus are found by doing a saliency analysis over pre-trained neural network. 62

Figure 4.8 50 mesh arc weights with highest absolute values for LG stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for LG stimulus are found by doing a saliency analysis over pre-trained neural network. 62

Figure 4.9 50 mesh arc weights with highest absolute values for SC stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for SC stimulus are found by doing a saliency analysis over pre-trained neural network. 63

Figure 4.10 50 mesh arc weights with highest absolute values for RP stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for RP stimulus are found by doing a saliency analysis over pre-trained neural network. 63

Figure 4.11 50 mesh arc weights with highest absolute values for EP stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for EP stimulus are found by doing a saliency analysis over pre-trained neural network. 64

Figure 4.12 The block diagram of the suggested deep architecture. Note that this architecture is trained in an unsupervised fashion. 64

LIST OF ALGORITHMS

ALGORITHMS

Algorithm 1 Stochastic gradient descent applied on neural network 25

LIST OF ABBREVIATIONS

fNIRS	Functional Near-infrared Spectroscopy
EEG	Electroencephalography
PET	Positron Emission Tomography
fMRI	Functional Magnetic Resonance Imaging
MR	Magnetic Resonance
HDR	Hemodynamic Response
BOLD	Blood-oxygen-level Dependent
HCI	Human-computer Interface
MNI	Montreal Neurological Institute
MVPA	Multi Voxel Pattern Analysis
PCA	Principal Component Analysis
ICA	Independent Component Analysis
MDS	Multi Dimensional Scaling
LLE	Local Linear Embedding
ROI	Region of Interest
LDA	Linear Discriminant Analysis
SVM	Support Vector Machine
RBM	Restricted Boltzmann Machine
MSE	Minimum Squared Error
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
GLM	General Linear Model
AAL	Automated Anatomical Labeling
FWHM	Full Width at Half Maximum
HCP	Human Connectome Project
DNN	Deep Neural Network
SDAE	Stacked Denoising Autoencoder

CHAPTER 1

INTRODUCTION

1.1 Problem Definition

Understanding the human brain has been one of the essential objectives of various scientific fields. Although early cognitive scientists and neurologists employed diagnostics or invasive methods for the analysis, recently invented non-invasive imaging technologies allow data-oriented researchers, such as computer scientists and statisticians, to involve in brain studies. Decoding the mental states from the neural measurements, which is known as brain decoding, is one of the important fields for understanding the brain in the neuroscience literature.

Many functional imaging technologies, such as functional near-infrared spectroscopy (fNIRS), electroencephalography (EEG), positron emission tomography (PET) and functional magnetic resonance imaging (fMRI) provide information about the working mechanism of the human brain with different characteristics. Because of it's high spatial and acceptable temporal resolutions, fMRI is the primary technique for brain decoding studies [2, 3]. In this thesis all experiments are implemented using fMRI images. Although, fMRI provides a detailed representation of the brain compared to other functional imaging technologies, high-dimensional data obtained as the output of fMRI machine makes these studies practically very difficult. Researches attempt to overcome this problem by modifying the representation space using various heuristics such as [4, 5, 6]. However, these approaches rely heavily upon the expert knowledge from neuroscience community and can not be shared among a wide range of cognitive tasks.

A major problem in brain decoding task is defining a valid representation which can be used to discriminate the cognitive states. Common experimental data include approximately 300 samples for one subject and each of these samples consist of approximately 40000 voxels. Practically, it is very difficult to design a machine learning system with such a feature-sample ratio. To overcome this problem, variants of factor models [7, 8] , non-linear dimensionality reduction techniques[9, 10, 11, 12] and feature extraction methods [13, 14, 15, 16] are proposed. Yet, none of them utilize the common and discriminate patterns shared among subjects.

Recent studies show that shared representation exists in the fMRI images of different subjects [17, 1]. In this study we take this proposal one step further and suggest that common representation among both multi-subject and multi-experiment data set can be detected by deep learning techniques. This representation is expected to overcome curse of dimensionality problem. We suggest to transfer the hierarchical information between different experiments we expected to make multi-subject brain decoding better than state-of-the-art studies. Brain decoding performance of hand-crafted and hierarchically learned representations are also compared to analyze whether involvement of expert knowledge can be discarded from brain decoding systems using deep learning techniques suggested in this thesis.

1.2 Proposed Method

Extracting relevant features is the problem of many domain enjoying learning systems. Deep learning architectures bring a state-of-the-art solutions for learning representation in many fields such as computer vision and natural language processing [18, 19, 20]. The essential reason for this success is the replacement of manual feature extraction processes by an automatic learning process. A great amount of data fed into deep architectures and multiple non-linear processing layers exist in these architectures, statistically common characteristics of samples can be hierarchically learned through the raw data.

As it is known from neuroscience studies, hierarchical processing layers also exist in the human brain [21]. To reveal this hierarchical and latent representation auto-

matically, multi-layer neural networks i.e. deep neural networks are employed in this thesis. As a first step, multi-layer neural networks are trained in an unsupervised fashion. fMRI data of 97 subjects of Human Connectome Project S500 data set are used [22] to test the performance of the suggested model for brain decoding task. Then, pre-trained networks are fine tuned with the labels of the same data set to implement multi-subject brain decoding. Besides, these networks are also used as a base network to obtain experiment-independent hidden representation for another brain decoding experiment. This approach makes it possible transferring the statistically common information between different cognitive tasks. Both temporal and spatial features extracted from the raw data are fed into deep neural networks, separately.

1.3 Contribution

Although few studies exist [17, 1, 23, 24] in the literature about applying deep architectures on fMRI data, these studies generally discriminate the healthy subjects from patients using resting state fMRI data which includes much more samples than brain decoding tasks. Besides, none of them transfer the learned representations among different cognitive tasks. The following items are the main novelties of this thesis :

- We suggest a deep neural network architecture which is capable of multi subject brain decoding.
- We provide a route map about applying deep neural networks on brain decoding problem for neuroscience researchers comparing various deep architectures and parameter values.
- We show that there is a common representation among different fMRI tasks and common representations can be learned by deep neural networks.
- We show that the learned representations can reflect temporal and spatial properties of fMRI data and can be used to discriminate cognitive tasks across subjects.

1.4 Thesis Outline

Starting with the definition of the problem of brain decoding, proposed method and contributions, thesis includes the following topics.

Chapter 2 will elaborates the details of fMRI data, experiment and pre-processing techniques. Since it is the major motivation of this thesis, curse of dimensionality problem for fMRI brain decoding will be elaborated. Then, a brief literature of data-oriented fMRI analysis studies will be provided, emphasizing the details of brain decoding studies.

Chapter 3 is devoted to the suggested deep brain decoding methods with the explanation of neural networks and deep neural networks. Theoretical details of the suggested deep architectures are also explained in that chapter. The characteristics of different activation functions, regularization techniques and optimization methods are studied.

Results of experiments performed in two different fMRI data set are provided in Chapter 4. This chapter is divided into two parts. Supervised mental state decoding results are provided in the first part. Effect of different characteristic of deep neural networks on brain decoding tasks are analyzed in this part. Second part of chapter 4 is devoted to examine whether or not it is possible to transfer statistical information among different cognitive tasks and across the subjects.

In Chapter 5, the suggested deep brain decoder is discussed and possible future directions of this work are suggested.

CHAPTER 2

MACHINE LEARNING FOR BRAIN DECODING BY FMRI DATA

Recently, number of studies applying machine learning techniques on fMRI data grows explosively [3, 5, 13, 25]. This chapter is devoted to overview how machine learning techniques are applied for brain decoding task. First, the problem of brain decoding considering the characteristics of fMRI data are explained. Then, experimental design and pre-processing steps of fMRI data are elaborated. Note that, different machine learning strategies are followed depending on the types of experiments and the pre-processing techniques affect the results significantly. Feature selection and extraction methods utilized in the fMRI literature are also elaborated to indicate the importance of representation on brain decoding problems. Finally, traditional machine learning approaches applied on fMRI data are discussed to analyze the advantages and drawbacks of these approaches, compared to the deep architectures.

2.1 fMRI Data and Brain Decoding

Magnetic resonance imaging (MR) is a non-invasive brain imaging technique invented by Lauterbur in 1973 [26]. Subjects are placed into an electromagnetic field to collect the MR data from them. Hydrogen atoms in the brain are aligned with the effect of magnetization level inside the MR scanner. Then, radio frequency pulses are used to transfer these atoms to higher magnetization level. Removal of this pulse causes hydrogen atoms to return their original position back, by generating different amount of currents in a receiver coil, which provides the MR signal. Gradient

magnetic field is also used to locate the different nuclei.

Although the current changes among different locations provides the static image of the brain, which is also known as the structural magnetic resonance image, temporal variations of the data is required to analyze the characteristics of the brain under different cognitive states. Changes in brain hemodynamics, which means dynamics affected by the blood, also change the intensity value of the MR signal locally. Functional magnetic resonance imaging (fMRI) is a type of MRI. fMRI generates sequence of brain images by collecting MR measurements quickly. The temporal resolution of collected data is related to the time gap between consecutive images, which generally changes between $[1-2]$ seconds. Lately, developed fMRI machines decrease this gap time to ~ 0.7 seconds.

The change of magnetic field in consequence of neural activity is called hemodynamic response (HDR). The essential form of fMRI uses blood-oxygenated-level-dependent (BOLD) contrast to map the neural activity in the brain. It is discovered by Ogawa et. al. [27] that deoxyhemoglobin concentration in the brain locally changes the magnetic field. Since the activated regions in the brain needs oxygenated blood to supply ample amount of energy to neurons, active and inactive regions of brain under a cognitive state can be detected by the fMRI technique. Sample BOLD contrast can be seen in Figure 2.1. Parametric definition of BOLD contrast can be found in [27].

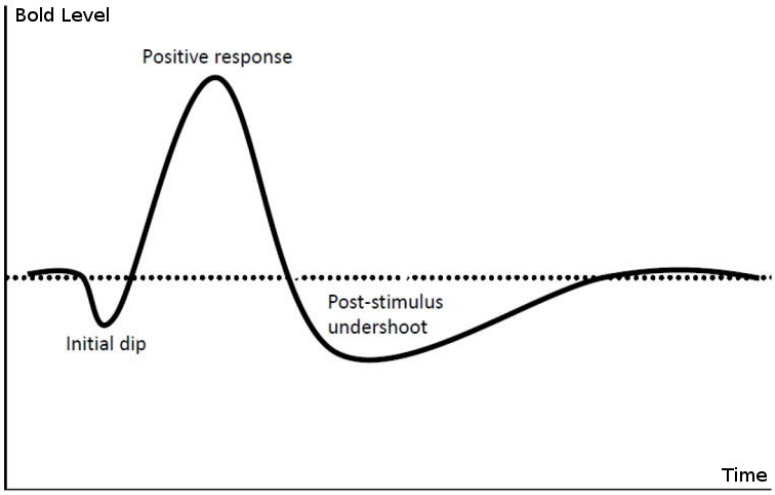


Figure 2.1: Sample BOLD contrast. [1]

[1] Image courtesy of [28]

As it can be seen from the sample BOLD contrast, when the neuronal tissue is activated, BOLD signal magnitude decreases in the first stage. Time gap between the neuronal activation and blood flow mechanism is the cause of that drop. Afterwards the BOLD signal reaches the maximum value in approximately 5 seconds, due to oxygenated blood is flowed to this area. Because of the assumption that the activated area is more likely to be activated again, just like the memory access assumptions used by operating systems, more than sufficient amount of blood is pumped to the activated area. When no more activation is occurred in this area, BOLD level falls back to the homeostatic level after ~ 15 seconds. In this thesis, approximate times of reaching the maximum BOLD signal value and falling back to the homeostatic level are used to determine the parameters of developed brain decoding models.

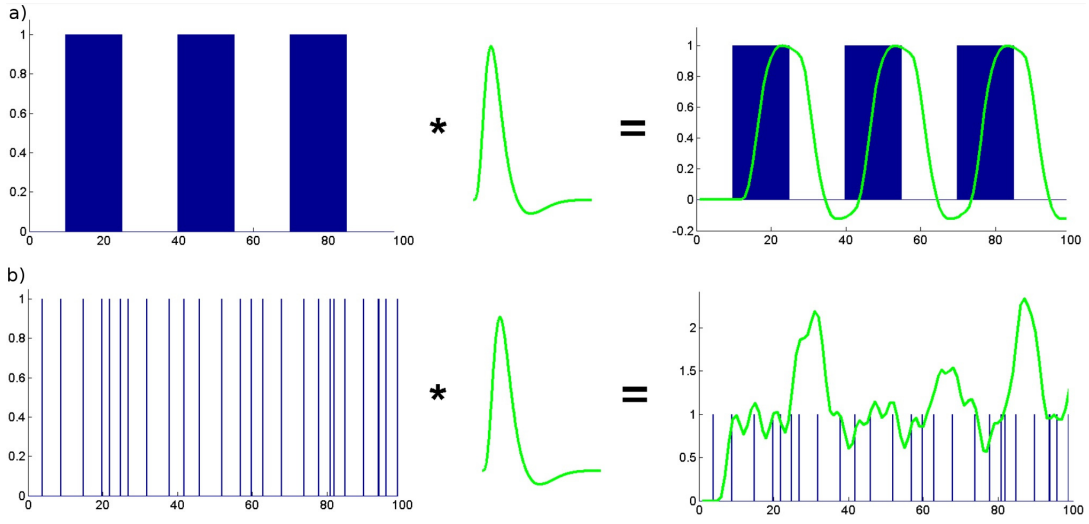


Figure 2.2: Figure a and b represent block and event-related designs, respectively. First signals of both a and b represent the stimuli function. As explained in the Figure 2.1 signal represented with green color is HDR function. Theoretical experiment signal is obtained by convolving HDR function with stimuli function. [3]

The BOLD signal characteristics mentioned above explains the fMRI signal acquisition process for one voxel, which is the smallest addressable element in the volumetric brain data. Depending on the spatial resolution of the experiment and the fMRI machine, there may exist 40.000 to 200.000 voxels in a single scan.

[3] Image courtesy of [29]

Smallest time interval that can be successfully separated out by fMRI is the temporal resolution. This period is determined by neuroscientists designing the experiment by taking different cognitive functions needs different amount of times to be done into consideration. Note that, BOLD signal is assumed to be a linear signal. In other words, HDR of two overlapping cognitive stimuli can be calculated by adding two singular HDRs. Linearity of HDR is first studied by Boynton [30] and it is the fundamental assumption behind each experiments conducted in this thesis. In Figure 2.2 stimulus function of the experiment, HDR and the modeled response function is given from left to right. The first row indicates an experiment where a stimulus of 20 seconds is shown to the subject, whereas second row shows a sample of event-related stimuli function and theoretical response function.

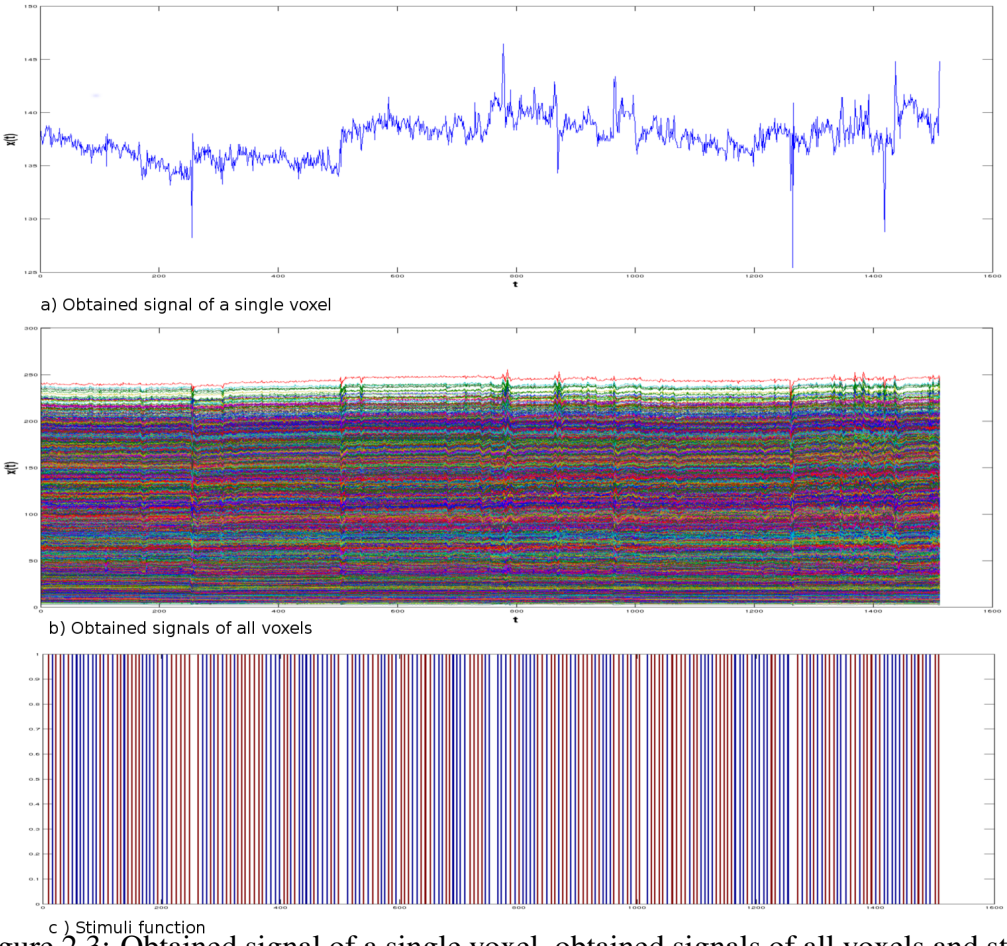


Figure 2.3: Obtained signal of a single voxel, obtained signals of all voxels and stimuli indices for each stimuli are given as a, b, and , respectively.

Sample signal extracted from one voxel and an image obtained by putting each voxel's

signal per row can be observed from a,b and c of Figure 2.3. In the undermost figure, red ones represents the time indexes of first class and blue ones represents the time indexes of second class. Note that these signals are not hypothetical but the real signals obtained from the conducted object detection experiment.

As shown in the Figure 2.3, different types of stimuli can be given to a subject inside the fMRI machine. Brain decoding or neural decoding is a technique that try to discriminate the type of stimulus using only functional images obtained from the subject. It can be considered as reverse engineering. Brain decoding researchers work on both single-subject and multi-subject brain decoding tasks. In this thesis, fMRI images are used to decode cognitive states. In single-subject brain decoding task cognitive state of the subject is estimated using the fMRI data of this subject. Yet, in the multi-subject brain decoding task, fMRI data of other subjects are used to estimate cognitive state of the selected subject.

2.2 fMRI Experiment Design and Pre-processing

Designing an fMRI experiment is a challenging process. Because it involves both data acquisition and design of the cognitive experiment, necessity of neuroscience expertize is inevitable. Both statistical power and neuroscientific validity must be maximized in experimental design [29].

There are two types of experimental designs in the fMRI literature, namely block design and event-related design. One can also use a mixed-design technique by combining block and event-related designs to utilize advantages of both of them.

In a block design, subject is exposed to a stimulus within the whole time of condition as a block. Stimulus changes only when a different condition is presented. This alternation of stimulus is known as "AB block design" [31]. That design approach had dominated the former fMRI experiments due to its robustness, greater statistical power and high HDR change. Yet, block design approach does not utilize the temporal resolution capabilities of fMRI over PET [32].

In an event-related design, subjects are exposed to a set of stimuli at discrete time

instants. Flashing light or showing an image for few hundreds of milliseconds are most commonly used events. Time gap between consecutive events are generally randomized to minimize the effect of systematic patterns of thought unrelated to the task. Event-related design provides researchers to discriminate different conditions and leads to real-time human computer interaction (HCI) systems. Note that, detecting an HDR activation pattern with event-related design is more difficult than block design.

Due to the noise in data acquisition process and physiological artifacts, fMRI data undergoes a number of pre-processing steps before the fMRI studies. Slice timing correction, motion correction, co-registration & normalization and spatial smoothing are the 4 major pre-processing steps. fMRI analyses assume that each voxel signal is temporally aligned. In reality, voxels located in different slices are measured sequentially. Slice time correction shifts each voxel's signal to align with each other. Since signal intensity values of the same voxel at different time points are used as the same feature values for different samples, fMRI scans must be aligned spatially. Motion correction pre-processing handle this alignment with transforming each fMRI image to the first image using rigid body transformation. For transforming the fMRI scans of each subject to more detailed and normalized space, co-registration and normalization pre-processes are applied. In this thesis fMRI scans of each subject are transformed to Montreal Neurological Institute (MNI) brain template. Although normalization add more noise to fMRI data, it is inevitable for group fMRI studies. As a final pre-process spatial smoothing is applied by convolving the fMRI images with a Gaussian kernel to improve inter-subject registration and decrease the effect of normalization. Detailed explanation of pre-processing steps can be found in [29].

2.3 Machine Learning for Brain Decoding

Initial fMRI studies concentrate on the analyses of raw fMRI signals. One of the important goals was detecting location and magnitude of experiment related HDR signals. These studies use single voxel analyses techniques. Seminal paper of Mitchell et. al. [33] draws the attention of machine learning researchers to fMRI domain. After this paper, researchers use multi-voxel pattern analysis (MVPA) approach to

apply machine learning algorithms over fMRI data. This enables us to characterize the fMRI signals better than the statistical approach and makes it possible to classify them [3]. That is achieved by using the intensity values of voxels as features of a learning systems. Yet, it is known that having about 40.000 features is problematic for machine learning algorithms.

As it is explained above, brain decoding with fMRI data is a sample of $p >> N$ problem, where p represents the number of features and N represents the number of samples. The curse of dimensionality phenomena arises for fMRI analysis which causes both high variance and overfitting problems for the learning algorithms [34]. In an effort to handle high dimensional samples, various dimensionality reduction techniques have been applied, which will be overviewed in the next subsection. Also, note that curse of dimensionality is not only a problem of dimension of the data, but also the algorithm being applied [35].

2.3.1 Dimensionality Reduction Techniques for fMRI data

As mentioned in the section 2.1, classifying the cognitive states from fMRI data is known as brain decoding. A classifier in fMRI brain decoding task is a function $f(y; x, \theta)$ where y represents the cognitive class label and x represents the high dimensional fMRI scan data and θ represents the parameters of the function f. The goal of machine learning approaches is to find the optimal parameters for the chosen function f. Yet, with the increasing number of feature dimension, the volume of the space where the samples can reside exponentially growth. Therefore, finding the optimal θ parameters become harder, which is known as "Hughes Effect" [36]. To overcome this effect, three common types of dimensionality reduction techniques are applied within fMRI analysis studies.

First type of dimensionality reduction technique applied on fMRI data is variants of factor models. Well known examples of this type are Principal Component Analysis (PCA) and Independent Component Analysis (ICA) [7, 8]. The purpose of both approaches is to find a new set of basis vectors to project the sample vectors. Then, projected samples $\widetilde{x}$ can be represented with the number of basis vectors less than the original dimension of the data. While PCA projects samples to lower dimensional

space with linearly uncorrelated basis, ICA finds basis which are statistically independent from each other. Since ICA works well when the raw signal have statistically independent source signals, it is preferred over PCA for fMRI analysis studies. Although both of these approaches are linear, by using kernel functions both of them can gain the ability to reduce dimension non-linearly. Yet, determining the kernel function requires domain knowledge.

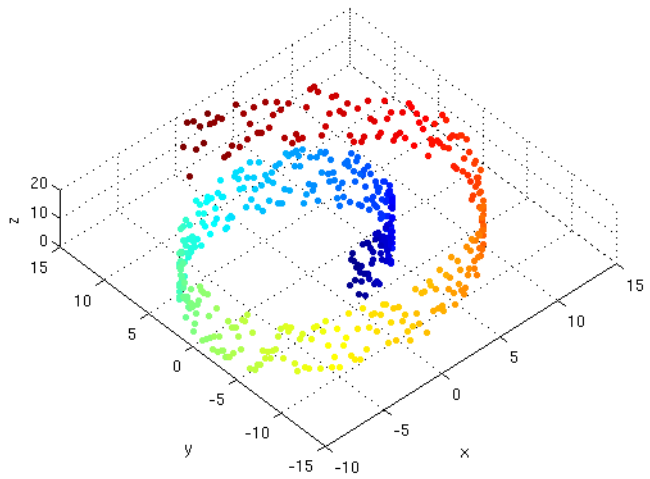


Figure 2.4: Swiss-roll data set.

Global and local non-linear techniques [37] are the second type of approaches, which maps the samples from high-dimensional space to low-dimensional space by preserving the global or local distances between samples as much as possible. These techniques are based on the intuition that high-dimensional raw data lies on a manifold, which is a complex low-dimensional space. Swiss roll data set is a famous example of 2D manifold inside 3D coordinate systems, which can be seen in figure 2.4

Mapping from high-dimensional to low-dimensional space is achieved by minimizing the stress function,

$$\phi(\widetilde{X}) = \sum_{ij} (dist(x_i, x_j) - dist(\widetilde{x_i}, \widetilde{x_j}))^2. \tag{2.1}$$

Function defined above minimizes the total distance between each pair of samples, which is known as the raw stress function of multidimensional scaling (MDS) [38]

family. There are various modified version of raw stress functions, namely, Sammon cost function [39] puts more emphasis on preserving the distance between samples which are close to each other in the original space. Local Linear Embedding (LLE) [40] calculates the stress function locally to preserve only local properties of the original data. Various studies utilize these methods to overcome curse of dimensionality problem by adding different regularities to stress function. Researches on non-linear techniques and application on fMRI data can be found in [9, 10, 11, 12].

Third approach includes supervised approaches, that uses class labels explicitly to find a transformation. Using the class labels provides these approaches to find new set of features with high task relevance [41]. That type of dimensionality reduction approach is generally used as a pre-process step to cognitive state decoding models.

2.3.2 Feature Extraction for fMRI

Although dimensionality reduction methods can eliminate the redundant voxels obtained in fMRI recordings, many researchers prefer to obtain representations from raw data using their domain knowledge. As in the field of computer vision, natural language processing or speech recognition, researchers analyzing fMRI data also aim to improve the discriminative quality of fMRI data using feature extraction techniques particular to fMRI.

One of the earliest feature extraction techniques applied to fMRI analysis is the *Searchlight* [42]. The basic assumption behind the *Searchlight* technique is that some regions of the brain are not active for a specific cognitive task. Furthermore, activated regions affect their neighbors. In order to both detect the activate parts of the brain and to gather the local effects from nearby voxels, 3D multivariate *Searchlight* is moved through each voxel. Activation degree of a voxel is founded by calculating a multivariate effect statistic between voxel intensity values inside that 3D *Searchlight* and stimulus function. Note that, gathering the local effect from nearby voxels also decrease the noise.

Representing the brain as a network is another feature extraction technique used in the fMRI data analysis studies. Nodes of the network is identified as a set of functional

nodes, which are generally spatial Region of Interests (ROI) defined biologically by one of the anatomical atlases or ICA maps obtained with group-fMRI studies. After the nodes of the network are determined, edges between these nodes are founded by conducting multi-variate statistical analysis between fMRI time series associated with them. There are many multi-variate statistical analysis applied in fMRI to obtain edge weights. Calculating the zero order pearson correlation between two nodes is the most commonly used technique [13, 14]. Graphical model based [15, 16] techniques are also used to consider all nodes simultaneously. These feature extraction techniques are used for group-fMRI studies, since stating each subject's brain as a network of common nodes handles the spatial differences among subjects fMRI data. Disease detection studies also use network representation to discriminate healthy and diseased subjects, since it is known that functional relation between anatomical ROIs are degenerated with brain diseases [43, 44].

Another important feature extraction technique used in this thesis is the mesh model [45]. As it can be seen from the figure 2.3 signal intensity value of particular voxel does not change drastically to discriminate it between different stimuli types. Ozay et. al. [45] proposed that representing the each voxel as a linear combination of it's neighbor voxels increases the discriminative power. Results of the mesh model can also be interpreted as a network-based model, where the intensity values of voxels represents the node labels and edges of the network are represented with the coefficients of the linear system.

One of the most important goal of this thesis is comparing the discriminative qualities of hand-crafted and hierarchically learned features. Then, discussing whether manual feature extraction processes can be replaced with deep hierarchical feature representation models.

2.3.3 Classifiers Applied on fMRI Data

As it is explained in [35], regularizing classifiers gives better results for high-dimensional classification tasks. Therefore, linear classifiers have convex cost function and less number of parameters compared to nonlinear ones and heavily used in brain decoding literature to discriminate samples obtained during into different cognitive states.

The goal of a linear classifiers is to find an hyperplane $\mathbf{w}^t\mathbf{x}+\mathbf{b}$ which ensures that $y = sgn(\mathbf{w}^t\mathbf{x}+\mathbf{b})$ returns $+1$ for the positive samples and -1 for the negative samples. In order to find optimal values for **w** and **b**, different approaches and cost functions are utilized. Two most commonly used ones in the fMRI domain, are Linear Discriminant Analysis (LDA) and Support Vector Machines which are explained below.

In LDA the samples from two classes are assumed to be conditionally normally distributed. The same full rank covariance matrix Σ and two different mean values μ_1 and μ_2 are used to define these normal distributions. Assuming that all Σ, μ_1 and μ_2 are known, optimal normal vector of separating hyperplane can be found as $\mathbf{w} = \Sigma^{-1}(\mu_1 - \mu_2)$. The greatest problem of LDA on high-dimensional problem is estimating the entries of Σ. Shrinking the full covariance matrix into a diagonal matrix can solve this problem by assuming that features are independent from each other [35]. Comparison of different LDA methods on decoding brain states with fMRI data can be found in [46].

Support Vector Machine (SVM) is the another popular approach for fMRI studies. Since it is robust to feature and sample size of the classification problem, it fits well to fMRI studies. Although both LDA and SVM, without kernel trick, are linear classifiers, SVM classifier find the hyperplane which maximizes the margin between separating hyperplane and sample from both classes which are closest to hyperplane among other samples from the same class. These samples are called as support vectors. As it can be seen from 2.5, the total margin between separating hyperplane and support vectors is $\frac{2}{||\mathbf{w}||}$. Then computing the parameters of SVM becomes equivalent to the problem of optimizing the following cost function

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(w \cdot x_i + b)\right)\right] + \lambda||w||^2. \tag{2.2}$$

By representing the cost function as a constrained optimization problem and solving the Lagrangian dual of it, one can convert 2.2 to an optimization problem which can be efficiently solved by a quadratic programming algorithm. Details of this optimization can be found in [47].

[5] Image courtesy of https://en.wikipedia.org/wiki/Support_vector_machine

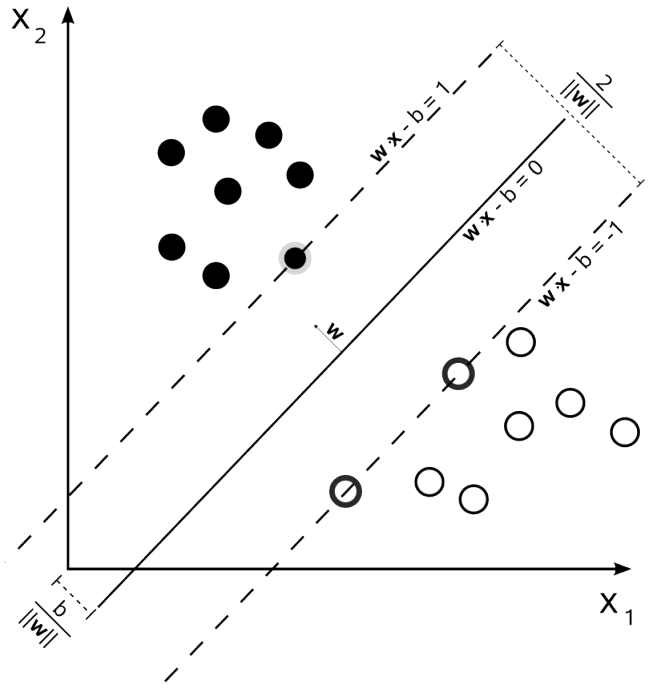


Figure 2.5: Hyperplane and maximum margins obtained with SVM. Samples on the margins are support vectors.[5]

Although these classifiers give better results on small datasets compared to the non-linear ones such as neural networks, they are not capable to generate discriminative features for the classification process. In order to handle both feature extraction and classification tasks simultaneously, deep learning techniques are utilized in this thesis. Details of the deep learning algorithms suggested in this thesis are explained in Chapter 4.

2.4 Chapter Summary

In this chapter constituent parts of the characteristics of the fMRI data, brain decoding with fMRI data, designing an fMRI experiment and multivariate fMRI analysis are explained. The most important ideas that should be summarized in this chapter can be listed as follows;

- Temporal variations detected by fMRI makes brain decoding possible using fMRI data.
- fMRI data obtained with event-related and block design can be theoretically

estimated with linearity assumption of fMRI signal.

- Since there exist about 40.000 voxels per each time instant, dimensionality reduction and feature extraction methods are necessary for brain decoding tasks.
- Since they are affected less from high dimension compared to other classifiers, LDA and SVM are most commonly used classifiers in the brain decoding literature.

Due to the curse of dimensionality problem, feature extraction or elimination step is inevitable for brain decoding task. However, we believe that there exist a common representation for different subjects and fMRI experiments. Utilizing this representation we can decode cognitive state of subject using others' fMRI data, transfer common representation among different experiments and attack curse of dimensionality problem at the same time. In order to obtain this common representation we use deep learning techniques. Next chapter is devoted to the theoretical background and implementation details of techniques applied in this thesis.

CHAPTER 3

BRAIN DECODING BY DEEP LEARNING

Artificial neural network (ANN) is a non-parametric approach for classification and regression problems inspired from human brain. The techniques applied in this thesis are developed from neural network family. Therefore, this chapter is devoted to theoretical background and implementation details of neural networks. Chapter starts with the biological basis of neural networks. Then, effect of depth, activation function and regularization methods are explained. Optimization and visualization techniques applied on the suggested neural networks are also described. Implementation details of deep neural networks used in this thesis are also explained in this chapter.

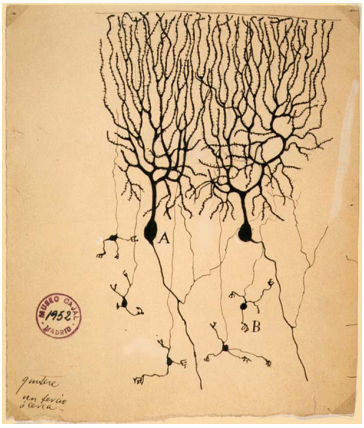


Figure 3.1: First drawing of Purkinje Cell by Cajal in 1952.

3.1 Biological Basis of Neural Networks

The human brain is one of the most complex network observed in the nature purportedly containing 100 billion neurons [48]. Neuron is a cell that manage the informa-

tion exist in the human brain. Electricity coding the information in the brain, just like computer, can be transmitted over these neurons. Neuron cells also include chemicals which can be released into the communication gap i.e. synapses between neurons by the effect of electricity. Being an electrically excitable cell is the most important functionality of neurons. They have the capability of generating and propagating action potentials. The first schematic representation of neuron cells, located in the pigeon cerebellum is depicted by Cajal in 1952, as it is shown in 3.1.

Although working mechanism of artificial neural networks and biological neural networks are completely different from each other, anatomical structure of biological neuron cell is the source of inspiration for the units of the artificial neural network utilized in the machine learning literature [49]. Each neuron cell consists of the cell body, dendrites and an axon. There is only one axon and cell body exist for each one of the neuron, but many dendrites can be stemmed from cell body of the neuron to collect different inputs from many other neurons to the given one. Each biological neuron propagates an action potential if the sum of input signals exceed some pre-defined threshold [50]. Responsibility of axon is propagating the output action potential to other neurons. Although only one axon stems from the neuron's cell body, it branches over the way to provide the same action potential to different neurons by contacting with their dendrites. Note that, feedback mechanism, plasticity and fuzzy nature of neurons are not explained here for clarity[51].

Representing the information through a hierarchical system is very effective in the meaning of both cost and space. Yet, it is not feasible to mimic network of whole brain with current computational resources. Therefore, artificial neural networks just use the very basic input-output structure of biological neural networks.

3.2 How Does Artificial Neural Network Work ?

As it is mentioned above, neurons of neural networks propagates electricity to other neurons according to the magnitude of input electricity. In [49] Frank Rosenblatt developed an artificial neuron, namely *perceptron* inspired by biological neurons. Although perceptron is rarely used in neural network literature, to understand the logic

behind the modern neural networks it is important to understand *perceptron*.

Perceptron is a linear binary classifier. It makes a decision about the class of the input $\mathbf{x}$ which is represented by a vector of features. A decision is taken according to the parameters $\mathbf{W}, b$. If the weighted sum of the input feature vector exceeds the threshold b, perceptron fires i.e. returns an output of 1, otherwise it returns 0. Mathematically,

$$f(\mathbf{X}; \mathbf{W}, b) = \begin{cases} 0, \text{if } \mathbf{x} \cdot \mathbf{W} \leq b \\ 1, \text{if } \mathbf{x} \cdot \mathbf{W} > b \end{cases} \tag{3.1}$$

Structure of the perceptron with an input vector of size 3 can be seen from the figure 3.2. Outputs of other neurons, which is shown by $\mathbf{x} = \{x_1, x_2, x_3\}$, are connected to the body of the neuron scaling with different weights $\mathbf{W} = \{w_1, w_2, w_3\}$. Note that, by adjusting the weight vector $\mathbf{W}$ of the perceptron properly, one can implement a NAND gate. Since NAND gates are known to be universal, any logical function can be designed using perceptrons.

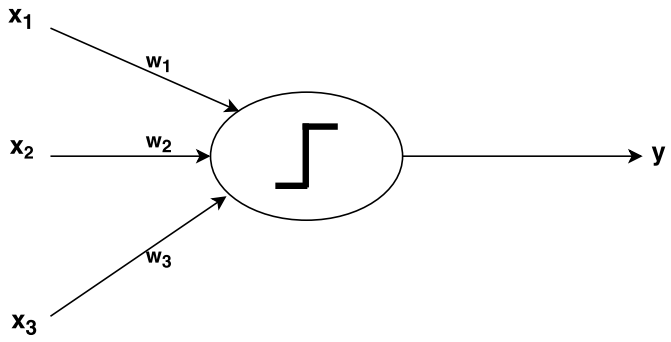


Figure 3.2: Perceptron classifier.

In order to learn a parametric function from the data, overall system must improve it's performance by deriving lessons from its mistakes. That can be implemented by changing the weights $\mathbf{W}$ slightly to change the discrimination function slightly using optimization techniques. Yet, all-or-nothing nature of the perceptron unit makes it inconvenient for such kind of learning mechanism. Small changes in $\mathbf{W}$ either does not change the output value of the perceptron or can fire it which is not fired before the adjustment because of zero and infinite gradient at the non-zero and zero inputs,

respectively. In order to assure that small change of weights results in small changes in the discrimination function, at least within the range of the input domain, non-linear activation functions are used. The details of how these neurons generate an output function and how one can adjust the parameters $\mathbf{W}$ programmatically are elaborated in the following subsections.

3.2.1 Forward Propagation

Artificial neural networks consist of a number of neurons organized in multiple layers, including input, output and hidden units. The neurons are very similar to perceptron consisting of multiple input units and single output unit. However, neurons of neural networks have non-linear activation functions that have positive or negative derivation in the input domain. In other words, to provide that small changes on parameters results with small changes on the output, activation functions are changed from step function to a continuous function.

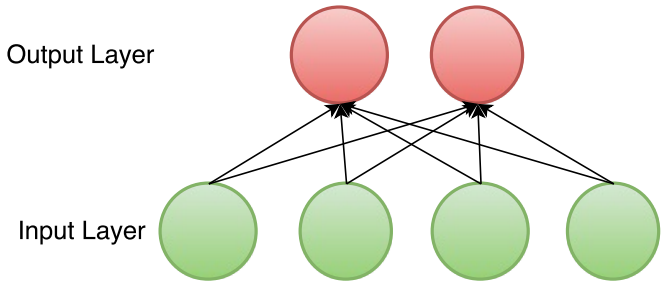


Figure 3.3: Two-layered neural network.

Let us define the activation function of each neuron by σ, then a sample architecture for 4 input units and 2 output units is shown in the figure 3.3. Note that, each level of a neural network is named as layer. First layer of the architecture is known as *input layer* and the last layer is known as the *output layer*. These two layers are given in figure 3.3 as the set of green and red nodes, respectively. As the names imply, goal of the input and output layer is to fed the architecture with the sensor data and to generate the vector output of the architecture, respectively. In order to obtain a hierarchical architecture, multiple hidden layers can also be used. These layers can reside between input and output layers.

Output $\mathbf{y}$ of the neural network with input $\mathbf{x}$ can be calculated layer-by-layer. In this thesis, parameters of each layer are represented with $\mathbf{W}^l$, where l is the index of the layer. Input z_i^l of the each neuron with $l \geq 1$ is equal to

$$z_i^l = \sum_{j=1}^{n^{l-1}} a_j^{l-1} w_{ij}^l + b_i^l, \tag{3.2}$$

where a_j^{l-1} is the output of the unit j at the layer $l-1$ and b_i^l is the bias for the unit i at the layer l of which input is calculated. As it can be seen from the figure 3.3, output of a unit is calculated by passing that value as an input to the non-linear σ function. Therefore, the activation value for the unit i of layer l can be calculated as $a_i^l = \sigma(z_i^l)$. Note that $\mathbf{a}^0 = \mathbf{x}$, because input of the initial layer are came from the sensors.

Utilizing the linear algebra and vector valued functions, output vector for each layer can be calculated as

$$\begin{aligned} \mathbf{z}^l &= \mathbf{W}^l \mathbf{a}^{l-1} + \mathbf{b}^\mathbf{l}, \\ \mathbf{a}^l &= \sigma(\mathbf{z}^l). \end{aligned} \tag{3.3}$$

Output of the neural network without any loop can be calculated using equations 3.2 and 3.3. Yet, obtaining a desired output from neural networks requires the weight parameters of the network to be adjusted. In the next subsection, we explain the methods to adjust the parameters of a neural network automatically.

3.2.2 Backward Propagation

Using equation 3.3, output of the neural network with input $\mathbf{x}$ can be calculated. However, in order to optimize W over all the levels one must define an error function and train the network. Let us assume that the error function, or more commonly known as cost function, is represented by $C(\mathbf{W}, \mathbf{b})$.

Then, the $C(\mathbf{W}, \mathbf{b})$ must be optimized with respect to $\mathbf{W}, \mathbf{b}$. Gradient based methods are used for optimizing $\mathbf{W}, \mathbf{b}$. In order to apply gradient based optimization on the

neural network, partial derivative $\frac{\partial C}{\partial w}$ and $\frac{\partial C}{\partial b}$ is calculated for each w and b. In order to make this calculation efficient, a well-known algorithm known as backpropagation is used [52]. The main idea behind this algorithm is propagating the error from upper layers to lower layers and calculating $\frac{\partial C}{\partial w}$ and $\frac{\partial C}{\partial b}$ based on these errors.

Error for the unit i at layer l is represented as δ_i^l. It is used to obtain partial derivatives for parameters. The error value represents the responsibility of given unit for the overall error.The error at the output units can easily be calculated using the overall error. Therefore, error value for the unit i at layer L, which is the penultimate layer, can be found as

$$\delta_i^L = \frac{\partial C}{\partial a_i^L}\sigma'(z_i^L). \tag{3.4}$$

In order to find the responsibility for units of hidden layers, error value is back-propagated. Because each unit at the layer $l-1$ affect the input of the layer l according to the parameters $\mathbf{W}^l$, error is also back-propagated in proportion to $\mathbf{W}^l$. Therefore, the error value δ_i^l can be calculated as

$$\delta_i^l = \sum_{j=1}^{n^{l+1}} w_{ji}^{l+1}\delta_j^{l+1}\sigma'(z_i^l). \tag{3.5}$$

After calculating the responsibility for each unit of the network, partial derivative according to each parameter can be calculated using these errors. Desired partial derivatives for $\mathbf{W}, \mathbf{b}$ are computed as

$$\begin{aligned} \frac{\partial C}{\partial w_{jk}^l} &= a_k^{l-1}\delta_j^l, \\ \frac{\partial C}{\partial b_j^l} &= \delta_j^l. \end{aligned} \tag{3.6}$$

Note that, similar to forward propagation, partial derivatives for each parameter can be obtained using linear algebra and vector valued functions. That makes it possible to obtain them in parallel. Many epochs, which means passing through all samples, are required for obtaining optimized values of parameters. Therefore, implement-

ing parallel version of optimization has became necessary for multi-layered neural networks.

Obtaining partial derivative for each parameter one can use any gradient based optimization method to obtain the optimal parameters for the given neural network. In subsections 3.4.4 and 3.4.5 several optimization and regularization method applied in this thesis will be explained.

To provide a complete overview of backpropagation method, Algorithm 1 presents an epoch of stochastic gradient descent algorithm for optimizing the neural network. Note that λ is the learning rate.

Initialize $\mathbf{W}$ and $\mathbf{b}$ randomly.
while *batchCounter < # of batches* **do**
  $\Delta_{\mathbf{W}} = \mathbf{0}$, $\Delta_{\mathbf{b}} = \mathbf{0}$
  while *sampleCounter < # of samples per batch* **do**
    Find the output for the given input, (3.3)
    Find $\frac{\partial C}{\partial \mathbf{W}}$, $\frac{\partial C}{\partial \mathbf{b}}$, (3.6)
    $\Delta_{\mathbf{W}} = \Delta_{\mathbf{W}} + \frac{\partial C}{\partial \mathbf{W}}$
    $\Delta_{\mathbf{b}} = \Delta_{\mathbf{b}} + \frac{\partial C}{\partial \mathbf{b}}$
  end
  $\mathbf{W} = \mathbf{W} - \frac{1}{sampleCounter}\lambda\Delta_{\mathbf{W}}$
  $\mathbf{b} = \mathbf{b} - \frac{1}{sampleCounter}\lambda\Delta_{\mathbf{b}}$
end

Algorithm 1: Stochastic gradient descent applied on neural network

Using a set of training samples $\{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$ and defining a cost function based on these values, neural network with arbitrary architecture can be trained. Yet, training a neural network with more parameters, in our case it is about 1 million. With the increasing number of layers, neural networks not only gain more capacity, but also, become more susceptible to overfitting problem. Due to this handicap of multi-layer neural networks, they had omitted by machine learning researchers until 2006. Recently, two different research groups [53, 54] show that training a deep neural networks layer-wise improves the generalization performance of them by providing better representations from lower layer to upper layers. In section 3.3, details of training a deep neural network is explained.

3.3 Deep Learning

Starting with two seminal papers [53, 54], a new branch of machine learning studies known as deep learning is emerged. Contrary to feature extraction studies in the literature of traditional machine learning, the goal of deep learning studies is understanding the latent relations between features in high-dimensional space. Deep learning algorithms require less engineering and utilize more from data compared to traditional methods [19]. Beside, since hierarchical representations can be obtained with these algorithms, transfer learning algorithms also can be implemented with them.

3.3.1 Why deep architectures ?

Success of machine learning algorithms is mostly based on the representation power of samples. Because different kind of representations can hide or disclose different factors behind the data, numerous methods are suggested to find an optimal representation for image recognition [55, 56], speech recognition [57] and also brain decoding [6, 45]. Yet, relying heavily upon the feature extraction methods is the weakness of traditional machine learning methods. Although knowledge of domain experts can easily be injected into learning systems while generating these features, that extraction step becomes a bottleneck when the domain information is limited, as in the brain decoding problem.

Representation learning methods have been used to automatically represent raw data as a feature vector [58] utilizing the statistical regularity of the sensor data. Goal of these methods are extracting information from raw data to provide discriminative features for prediction tasks. In general, feature vectors which can be separated linearly into different classes are admitted as more discriminative. There are many approaches exist for the representation learning, yet taking the hierarchical nature of the brain into consideration deep learning methods are used to learn complex feature vectors in this thesis. These methods generate representations in multiple levels. High-level ones are extracted from lower-level ones by non-linearly composing them.

Various architecture types, activation functions, regularization methods and optimization techniques are utilized to form a deep learning method. The following subsec-

tions are devoted to explain the details of components used in the experiments of this thesis. Additionally, in order to observe what kind of representations are obtained through the suggested deep learning methods, learned architectures are visualized. Visualization procedure applied in this thesis is also explained.

3.3.2 Unsupervised Networks

As in the general machine learning literature, two main types of networks exist in the deep learning literature, namely supervised and unsupervised networks. They can also be used together to create a hybrid system.

Goal of unsupervised deep neural networks is to find a hierarchical representation using input vectors $\mathbf{D} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_3\}$, where the samples do not have any label information. There are two main approaches in the deep learning literature to find hierarchical representations from unlabeled data set, namely Restricted Boltzmann Machines and autoencoders. The following sections will briefly overview these approaches.

3.3.2.1 Restricted Boltzmann Machines

A very popular unsupervised deep learning approach is known as Restricted Boltzmann Machines (RBM) [59]. It is implemented as a graphical model [60]. By minimizing the energy of graphical model with input vectors and maximize it with non-input vectors, RBM learns the optimal parameters for the input vectors. Formally, if the energy function of the graphical model is defined as $E(\mathbf{x})$ and likelihood of input vector $\mathbf{x}$ is defined as $p(\mathbf{x}) = \frac{e^{-E(\mathbf{x})}}{Z}$, where $Z = \sum_{\forall_{\mathbf{x}}} e^{-E(\mathbf{x})}$ goal of the RBM is maximizing the likelihood function $p(\mathbf{x})$ for input vectors and minimizing it for non-input values. Although the parameters of the system can be learned by stochastic gradient descent with negative log-likelihood function, it requires to calculate the energy of the model for each possible configuration.

In order to handle this optimization problem, in [61] contrastive divergence method is proposed. This method computes both positive and negative gradient for the gradient

based optimization method. In order to compute positive gradient, outer product of given training input representation and sampled hidden activation representation is used. Negative gradient is obtained with the outer product of reconstructed input and re-sampled hidden activation representations. Then weight update is obtained by adding the positive gradient and subtracting the negative gradient. To improve the expressive power of the network hidden units are added over input units. [53] used RBM to initialize the parameters of the deep neural network architecture.

3.3.2.2 Single Layer Autoencoders

Second approach for learning the discriminative representation from unlabeled data is autoencoders. Although the main idea of autoencoder is similar to the RBM, it is a discriminative model. Also it is easier to regularize the autoencoder by constraining the activation values, parameters or the architecture of it. Therefore, autoencoders are utilized in this study to both learn a hierarchical representation and initialize formed deep neural network.

The goal of the autoencoder is to find a function $H_{wb}(\mathbf{x}) = \mathbf{y} \approx \mathbf{x}$. It is indeed a neural network with the goal of mapping an input vector to itself. Because there is no label information exist in the H function, unsupervised cost function must be defined to implement backpropagation over the architecture. The most commonly used cost function is the mean squared error function which is defined as $J(\mathbf{W}, \mathbf{b}) = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y}_i - \mathbf{x}_i)^2$, where $\mathbf{N}$ is the number of samples. Trivially, identity function can map the input $\mathbf{x}$ to the output $\mathbf{y}$ as they will be identical. However, finding such mapping can not capture the statistical regulation within the data.

3.4 Deep Neural Decoding

As mentioned in the section 3.3, unsupervised deep neural networks are capable to extract representations from the unlabeled data. It is known that brain works in a hierarchical manner. We believe that these differences between different subjects emerge at the upper layers of this hierarchy. In order to find the similarities in the lower layers of this hierarchy we employed deep neural networks. Founding hierarchical similari-

ties between different subjects let us represent them with the common patterns. In the following subsections, we explain how we find these common patterns.

3.4.1 Stacked Autoencoders : Layer-by-layer training and fine-tuning

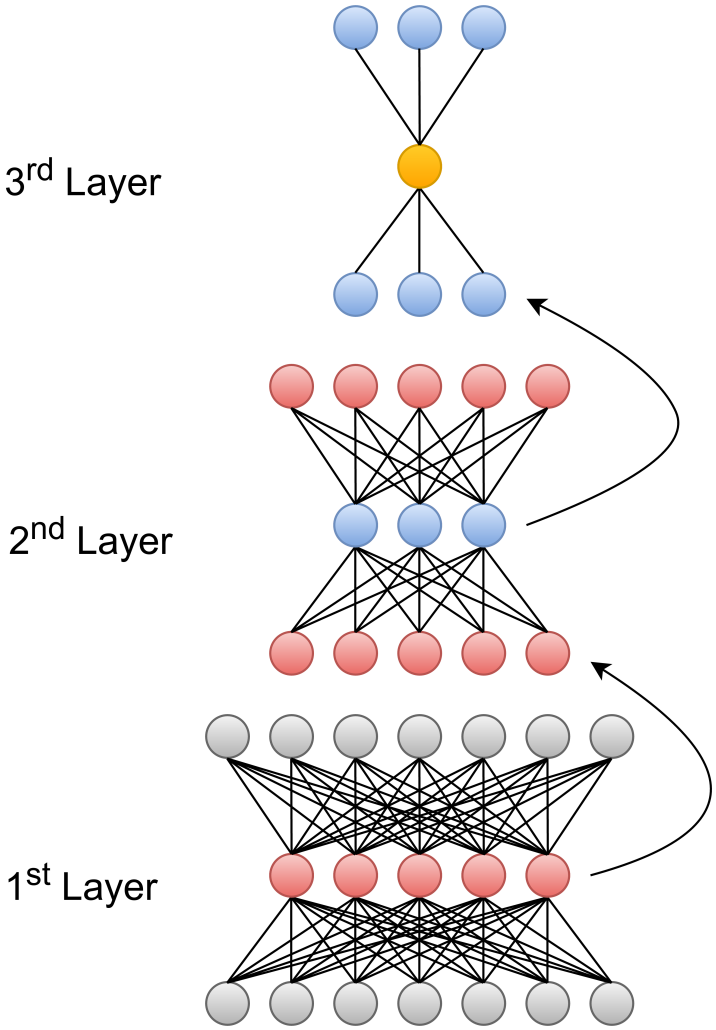


Figure 3.4: Architecture of 3-layered autoencoder.

In order to find hierarchically common patterns from the unlabeled fMRI data, number of autoencoders are stacked. While stacking autoencoders over each other, each layer are trained separately. Goal of unsupervised learning in our approach is to find experiment-independent common patterns from the fMRI recordings.

Although the goal of the unsupervised networks is specified above as forming a hierarchical representation using unlabeled data set, neural networks with only single hidden layer is explained up to now. In order to extend that explanation to networks

with more than one hidden layer, output vector of each hidden layer must be discriminated from each other. Let the output vector of the first hidden layer for the input vector $\mathbf{x}_i$ is shown as $\mathbf{e}_i{}^1$. To obtain the representation from upper layers, greedy layer-wise optimization is applied as it is shown in the figure 3.4. Grey nodes in the figure represents the input vector given to the autoencoder. Nodes of the first hidden layer are shown as red nodes in the lowermost network. Input and output nodes of 2^{nd} layer are also pictured as red nodes, since the same vector obtained from the 1^{st} hidden layer is used there. Similarly, the output of 2^{nd} hidden layer is used as an input and output for the 3^{rd} layer. Number of levels used to obtain hierarchical representation with autoencoder is also a hyper-parameter. Therefore, the hidden layer count of neural network is also found with cross-validation technique in this thesis.

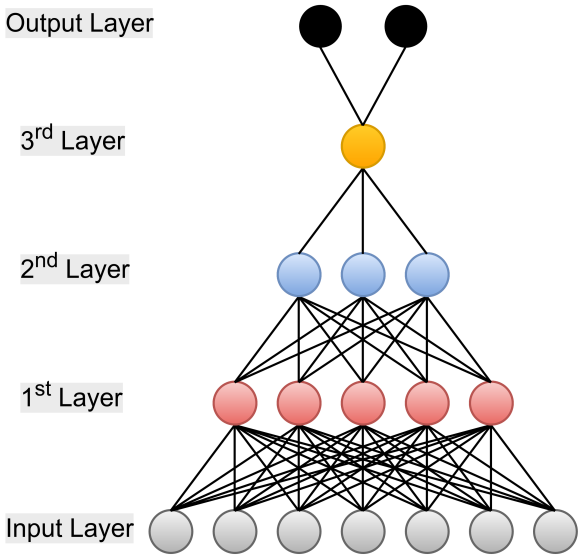


Figure 3.5: Multi-layer neural network classifier.

Let us describe the layer-wise training procedure. After the representation $\mathbf{e}^1$ is obtained from the first hidden layer, 2^{nd} hidden layer's representation is computed by using $\mathbf{e}^1$ as the input of the second layer. Then, $\mathbf{e}^2$ vector is used as an input to the third layer to obtain 3^{rd} layer representation. This process can be applied layer-wise for each hidden layer of the network. Note that, the optimization process on layer l^h doesn't affect the parameters of the layers different than l^h.

Although unsupervised networks provide an abstract and hierarchical representation of an unlabeled data set, they do not force the system to learn representations suitable

for classification tasks. The learned parameters are further adjusted to get a suitable representation after the unsupervised learning steps. Hidden layers trained with autoencoder are stacked over each other to form a new neural network as given in the figure 3.5. Note that, learning the units of a hidden layer and learning the values of parameters $\mathbf{W}, \mathbf{b}$ connecting output vector of former layer to current layer is used interchangeably in this thesis. In order to fine-tune the parameters favorable to classification task, they are optimized in supervised fashion using techniques explained in the section 3.2.

3.4.2 Constraints applied over unsupervised networks

As mentioned above, deep neural networks are pre-trained in an unsupervised manner to obtain a hierarchically common patterns embedded in fMRI data. Yet, to learn abstract representations from the data itself some constraints must be applied. Otherwise these networks can easily learn the identity function. In order to avoid this degenerate solution, two different kind of constraints consistent with the nature of the fMRI data is applied over unsupervised networks, which will be explained next.

It is known that different parts of the brain is responsible for different type of cognitive tasks. Therefore, the first constraint applied on the hidden layer of the autoencoder is to keep the number of units in hidden layer fewer than the input and the output layer. This constraint forces architecture to reconstruct vector given in the input layer using a compressed representation of itself. The basic idea behind this architecture is similar to PCA [8] in the sense that both try to project data to the space with fewer dimensions, yet using a non-linear activation function on each unit of the hidden layer makes this architecture non-linear extension of the PCA. The relationship between PCA and compressed autoencoder is studied in the [58]. Although this constraint approximately estimates the function of the manifold space where the samples lie on, other constraints exist in the literature to improve discriminative power and noise robustness of the autoencoder [58]. Since it generates both sparse and robust representation, denoising autoencoder is also applied in this study.

The constraint applied in this study is adding noise to the input vector, and reconstructing the raw version of the input vector using noisy version of it. It is known that

obtaining the HRF via fMRI is a noisy process. The noise embedded in fMRI signal is tried to be handled by denoising the encoder function. Formally speaking, for a given sample $\mathbf{x}_i$, output of the encoder function is calculated as,

$$\begin{aligned} o &= Bernoulli(u), \\ \overline{\mathbf{x}}_i &= o \cdot \mathbf{x}_i, \\ \mathbf{k}_i &= \mathbf{W}\,\overline{\mathbf{x}}_i + \mathbf{b}, \\ \mathbf{e}_i &= \sigma(\mathbf{k}_i), \end{aligned} \tag{3.7}$$

where u represents the corruption level, $\cdot$ is the dot production operation and f is the non-linear activation function of the hidden layer units. Then the decoder function minimizes mean squared error (MSE) between $\mathbf{e}_i$ and $\mathbf{x}_i$ for each i.

Note that hyper-parameter u of the 3.7 must be found with cross-validation technique, because it can not be optimized with gradient-based optimization techniques.

Theoretical explanations given above in this section do not include the effect of activation function, regularization methods and optimization techniques on the performance of the deep neural networks with fMRI data. Following subsections elaborate these details.

3.4.3 Details of Activation Functions

As explained in section 3.2, it is necessary to map input of neurons to output with a smooth function to avoid degenerate solutions. Because it closely mimics the characteristics of the perceptron and can be interpreted as probability of neuron's firing, sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$ is used in the former studies [58]. Range values of it changes between [0,1]. It saturates negatively at about -6 and positively at about +6. In order to improve the generalization performance of neural networks, $tanh$ function $tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$, which is the shifted version of the sigmoid function with the relation $tanh(x) = 2\sigma(2 \cdot x) - 1$, is favored over sigmoid. $tanh$ results in more generalized performances compared to σ function, because it has stronger gradients than sigmoid function and removes a bias over gradient located beyond the first layer

of the network by returning an output values between [-1,1] [62].

Although $tanh$ activation function solves two important problems of the sigmoid activation function, it can not handle the *vanishing gradient problem*. Basically *vanishing gradient problem* is that gradient of activation function becomes increasingly small with the increase of the absolute value of the input. To keep the gradient of the function constant for the varying inputs, rectified linear unit ($ReLU$) activation function is proposed as $ReLU(x) = max(0, x)$. $ReLU$ takes the value of zero for negative inputs and it linearly increases with the increasing value of positive input. Beside decreasing the effect of the *vanishing gradient problem*, $ReLU$ activation function also provides sparser representation compared to σ and $tanh$ activation functions. Since it is known that neurons of the brain uses a sparse representation [63, 64], choosing the $ReLU$ as an activation is more proper to mimic the network of biological neurons.

As it is mentioned above, derivation of $ReLU$ activation function for input values smaller than zero is equal to zero. In order to avoid zero gradient, parametric version of $ReLU$ function ($PReLU$) is proposed in the [65]. Formally, $PReLU$ activation function can be defined as

$$f(x) = \begin{cases} x, \text{if } x > 0 \\ \alpha\, x, \text{if } x \leq 0 \end{cases} . \tag{3.8}$$

α parameter controls the slope of the negative part of the activation function. Using $PReLU$ activation function introduces a new parameter to the system per each layer. So, no extra risk of overfitting is expected. Furthermore, optimizing the value of α simultaneously with $\mathbf{W}, \mathbf{b}$ values using backpropagation adds negligible computational complexity to the system.

3.4.4 Regularization Methods

Up until this point, the methods are discussed to improve the classification or regression performance of the inputs given to deep neural network, which are known as *training data set*. However, the main goal of the deep architectures is to perform equally well on unseen inputs, which are known as *test data set*. In order to reduce

the error on the *test data set* by possibly increasing the error on the *training data set* various regularization techniques are applied within the experiments of this thesis.

3.4.4.1 Parameter Norm Penalties

Having a great complexity, deep neural networks can easily overfit to the *training data set*. Overfitting occurs when the model has much more parameters than the number of training samples [34]. In order to limit the capacity of the architecture, parameter norm penalty can be applied over $\mathbf{W}$ parameters of the network. If parameter norm penalty is shown with $\Omega(\mathbf{W})$, regularized objective function of the overall network can be represented with

$$J_{reg}(\mathbf{W}, \mathbf{b}) = J(\mathbf{W}, \mathbf{b}) + \gamma\Omega(\mathbf{W}), \tag{3.9}$$

where γ weights the effect of regularization term for the overall cost $J_{reg}(\mathbf{W}, \mathbf{b})$ and $\gamma \in [0, \inf)$. Setting γ to the 0 means there is no parameter norm penalty is applied over cost function and larger γ values result with more regularized cost function. Note that parameter norm penalty does not contain $\mathbf{b}$ values, because having higher values of $\mathbf{b}$ only shifts the obtained function. Note that, $\mathbf{b}$ parameters can be optimized with less amount of samples compared to $\mathbf{W}$, because each $\mathbf{W}$ value deals with interaction of two variables but each $\mathbf{b}$ variable controls only one variable [66].

Both L_1 and L_2 parameter norm penalties are applied over networks in this thesis. The penalty term L_n is defined as $L_n = \frac{1}{2}||w||_n^n$. Therefore, $L_1 \propto ||w||_1^1 = \sum_{\forall w_i} |w_i|$ and $L_2 = \frac{1}{2}||w||_2^2$. Adding L_1 and L_2 penalties over the cost function J affect the characteristics of estimated $\mathbf{W}$ differently. While L_2 penalty function diminishes the large $\mathbf{W}$ values more, it doesn't force the system to have a sparse representation as L_1 penalty function does. On the other hand, L_1 can select at most K variables before the saturation on the problems having K samples with the dimension of I, where $I >> K$ [67]. It just omits the remaining features. Note that, brain decoding problem is good example of high-dimensional samples with few examples case. Therefore L_2 norm penalty is added over L_1 norm penalty term to obtain a hybrid regularization cost functions as follows.

$$
\begin{aligned}
J_{reg_2}(\mathbf{W}, \mathbf{b}) &= J(\mathbf{W}, \mathbf{b}) + \frac{\gamma}{2}\mathbf{W}^T\mathbf{W}, \\
\Delta_w J_{reg_2}(\mathbf{W}, \mathbf{b}) &= \gamma w + \Delta_w J(\mathbf{W}, \mathbf{b}), \\
w &\leftarrow (1 - \lambda\gamma)w - \lambda J(\mathbf{W}, \mathbf{b}),
\end{aligned}
\tag{3.10}
$$

and

$$
\begin{aligned}
J_{reg_1}(\mathbf{W}, \mathbf{b}) &= J(\mathbf{W}, \mathbf{b}) + \gamma||\mathbf{W}||_1^1, \\
\Delta_{\mathbf{W}} J_{reg_1}(\mathbf{W}, \mathbf{b}) &= \gamma sign(\mathbf{W}) + \Delta_{\mathbf{W}} J(\mathbf{W}, \mathbf{b}), \\
\mathbf{W} &\leftarrow \mathbf{W} - \lambda(\gamma sign(\mathbf{W}) + \Delta_{\mathbf{W}} J(\mathbf{W}, \mathbf{b})).
\end{aligned}
\tag{3.11}
$$

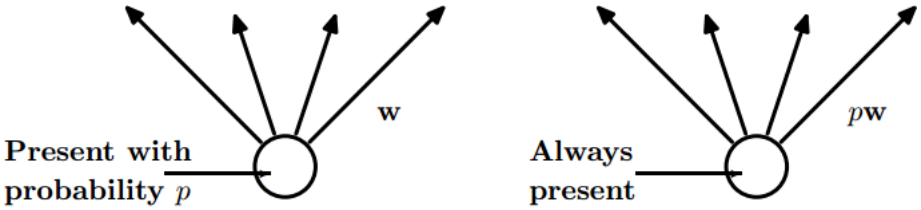
Parameter updates with L_1 and L_2 norm penalties using gradient descent approach is given in 3.10 and 3.11. As it can be seen from the update step, derivative of L_1 term penalty is stable for positive and negative values and equals to -1 and 1, respectively. It forces parameters to become zero constantly at each input. On the other hand, L_2 norm penalty diminishes the greater values of $\mathbf{W}$ more than smaller values of $\mathbf{W}$. Applying both L_1 and L_2 provide the system to force both greater and smaller values of $\mathbf{W}$ to become 0. Effect of norm penalty regularization on the obtained results can be observed in the experiment of chapter 4.

3.4.4.2 Dropout

Although L_1 and L_2 regularization methods increase the generalization performance of the system, these penalties are not enough to make the system more robust against the noisy inputs in the *training data set*. Since fMRI machine records the signals from voxels according to indirect measurements of the blood flow, the data consists of noise which is generated by multiple sources. Srivastava et. al. [68] proposed the *dropout* method to overcome this problem.

Dropout method uses the *bagging* idea to increase the robustness of deep neural network against noisy inputs [68]. Essence of the *bagging* idea is to train the model by different subsets of the training data set. This approach enables to obtain a set of

classification performances for the each data set. Then, the final result is obtained by weighting the classification scores of these models. By adjusting the weights of model scores, the overall system can maximally utilize the advantages of different models. However, training deep neural network from scratch is tedious and expensive process. Therefore, implementing the traditional bagging is not feasible. In the *dropout* method, approximation of bagging is implemented by dropping out different visible and hidden units for each step of the batch gradient descent. Units are dropped randomly with the probability of p within the training phase. Within the test phase each unit is always present, but each element of $\mathbf{W}$ is multiplied by p. Behavior of a sample unit exposed to dropout at training and testing phases is shown in the figure 3.6.



(a) At training time (b) At test time

Figure 3.6: Behavior of a sample unit exposed to dropout at training and testing phase[2].

Because activation or deactivation of a unit is independent from other units, different p values can be used for different units. The effect of the amount of dropout will be discussed in Chapter 4. As it can be seen from the results in chapter 4, deactivating %90 of features gives the best results for brain decoding task, which shows us the percentage of redundant features in the recorded fMRI data.

3.4.4.3 Early Stopping

As it is known generalization error is calculated by feeding the unseen data to the model trained with *training data set*. Although forcing the system to learn parameters for a general representation with norm penalties and dropout improves the generalization performance of the trained model, taking unseen data into consideration gives

[2]Image courtesy of [68]

better results. In order to utilize unseen data for training the model without mixing *training data set* and *test data set*, part of the data is reserved as *validation data set*.

After the part of the data set is reserved as *validation data set*, result of the trained model on this data set is used to determine best hyper-parameters of the system which are mentioned above. For assisting the optimization procedure by determining the values of hyper-parameters, *early stopping* is also applied in this thesis. By assuming that both *validation data set* and *test data set* are generated from the same distribution, training procedure is terminated when the classification error on the *validation data set* is started to increase while the classification error on the *training data set* still decreases. By identifying the iteration when the overfitting starts in the optimization process, more generalized model is obtained using *early stopping* in this thesis.

3.4.4.4 Batch Normalization

Up to this point each regularization method mentioned above assumes that all samples in the *training data set* is generated from the same distribution. Although making such an assumption is valid for single subject brain decoding task, fMRI data of different subjects have varying statistical characteristics.

It is known that training phase of the deep neural network converges faster if its inputs are whitened [69]. Although whitening the inputs to the deep neural network handles the statistical variations to a certain extent for the input layer, small changes in lower layers are amplified through the upper layers of the deep neural network . The network model is partially suffered from the covariate shift [69]. Hidden layers of the neural network suffers from this problem due to the change in the distributions of these layers results in a continuous effort for adaptation to the new distribution. Ensuring the distribution of inputs of hidden layers remains more stable helps optimizer to get stuck in the saturated regime [69]. As it will be represented in the experiments chapter, intra-class distance between fMRI recordings of different subjects cause substantial changes in the distribution of internal input nodes. Adjusting the distribution via fixing the mean and variance attacks this regularization problem.

In order to fix the distribution of inputs for internal layers, normalization is applied

to each activation independently using the *batch normalization*. Let the k^{th} particular activation is represented as a^k and we have m values of this activation in the mini-batch. Note that, training of network models are implemented by separating training samples into batches. Algorithm 1 is followed for each of them separately. Each of these subsets of training samples are named as mini-batch. For each sample the output $\mathbf{y}_i$ is calculated as,

$$\begin{aligned}
\mu_\beta &\triangleq \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i, \\
\sigma_\beta^2 &\triangleq \frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}_i-\mu_\beta)^2, \\
\hat{\mathbf{x}_i} &\triangleq \frac{\mathbf{x}_i-\mu_\beta}{\sqrt{\sigma_\beta^2+\epsilon}}, \\
\mathbf{y}_i &\triangleq \theta\hat{\mathbf{x}_i}+\beta \triangleq \tau_{\gamma,\beta}(\mathbf{x}_i).
\end{aligned} \tag{3.12}$$

Because *batch normalization* is applied as a part of the training procedure of the deep neural network, parameters θ and β are found with back-propagating the gradient of loss through the transformation given in the 3.12. Details of the chain rule including *batch normalization* layer can be found in the [69].

As it can be seen in the experiments chapter, applying regularization techniques on the fMRI data improves the brain decoding performance. Without the regularization technique mentioned above, classifier of multi-subject brain decoding task can easily overfit to the training samples which are the fMRI data of the part of the subjects. Effect of regularization techniques mentioned in the subsection 3.4.4 on multi-subject brain decoding task can be observed in the experiments chapter.

3.4.5 The Optimization Techniques for SDAE

As mention in the section 3.2.1, Renaissance of the neural networks begins with the invention of back-propagation technique [52]. This technique is built on the non-convex optimization theory. The greatest hurdle the deep neural network and generally all machine learning researches is to find the global minimum of the given

non-convex function.

Optimization of a deep neural network is tedious process compared to that of traditional models such as SVM and LDA. Probability of getting stucked at a local minimum increase with increasing number of parameters, which is unavoidable in the fMRI literature. Although the widely-used convolution technique [66] decreases the number of parameters drastically, the fMRI images are not equivariance, which means that the location of activation is unimportant . Therefore, obtaining a representation using convolutional neural networks decreases the brain decoding performance. Therefore, fully connected deep neural networks are used as an architecture in this thesis.

In the neural network literature many optimization techniques are developed. Most important ones are stochastic gradient method (SGD), momentum-based methods, second-order methods and first-order per-dimension methods [66]. Due to large amount of parameters in our architectures, implementing second-order methods are too expensive to implement. Therefore the first three optimization techniques mention above is utilized in our experiments.

Although both SGD and momentum-based methods achieve good performances in various domains [54, 58], adjusting the learning rate parameter is very difficult issue and both techniques are very sensitive to initialization of the weights. The effect of initialization and learning rate parameters are explained in chapter 4.

On the other hand, per-feature first-order methods introduce dynamic learning rate parameter which is computed per-feature basis using only results of first derivation [70]. Per-feature first-order methods don't need manual setting of learning rate and there are no robust methods in the deep neural network literature with great amount of parameters for selecting a good learning rates. These techniques provide us to get rid of learning rate selection problem.

Within our experiments ADADELTA optimization [71] technique results with best results. As it can be seen from the results, ADADELTA converges in dramatically less epochs when it is compared with other optimization techniques, SGD and momentum-based methods. Therefore, the generalization performance of classifier increases and

system overfit less on the samples of training subjects.

3.4.6 Visualizing the obtained parameters

In the above sections of this chapter we have approached the brain decoding problem from the perspective of the machine learning literature. Yet, the neuroscience research is very much involved in identifying the responsible brain ROIs for the given experiment.

There are variety of methods in the neuroscience literature to detect the active ROIs of brain under a cognitive stimulus. There are two commonly used methods in the neuroscience literature, one of these approaches is most commonly used for single subject studies and the other is used generally for multi-subject studies.

Among many others, the single subject studies used the general linear model (GLM) very frequently [29]. In the GLM, intensity signals of each voxel, or each ROI, is reconstructed using the experimental signals. Generally, stimulus function convoluted with HRF and random motion signals are used as a signal within the experimental signals. Mathematically speaking, intensity signal y_v of each voxel is obtained as $y_v = \beta \cdot \mathbf{x_e} + \eta_e$ where the β is the coefficients of experimental signals $\mathbf{x_e}$ and η_e is the error term. After obtaining different β parameters for each voxel, t-test or f-test is used to detect regions activated more given the experimental stimulus.

For multi-subject brain decoding tasks, the fMRI signals obtained from each subject is simultaneously used to obtain independent basis signals. Signal of a voxel obtained from all subjects is used as a channel. Then, signal of each voxel is used as a separate channel. Finally ICA is applied over these channels to obtain independent components from these channels. Ranking the basis vectors according to significance value, researchers obtain most important basis for the experiment. Commonly activated regions for each subject are observed by neuroscientists by visualizing these ICA basis.

In this thesis, visualization of active brain regions is achieved by using the deep neural networks. Active regions of the brain given a type of stimulus is found by looking for an input pattern that maximizes the activation of the output unit responsible for

that type of stimulus. This pattern is identified by applying a simple gradient ascent method in the input space. Mathematically, the goal of the gradient ascent optimization is to find

$$\mathbf{x_i}^* = \underset{\mathbf{x}\, s.t. ||\mathbf{x}||=1}{\arg\max}\ f_i(\mathbf{W}, \mathbf{b}, \mathbf{x}), \tag{3.13}$$

where $f_i(\mathbf{W}, \mathbf{X})$ is i^{th} element of the output vector, in other words the activation value for the stimulus type i. In order to find the representation $\mathbf{x_i}^*$, which is the optimal input for the i^{th} class in this case, gradient of the $f_i(\mathbf{W}, \mathbf{x}, \mathbf{b})$ is calculated for $\mathbf{X}$ and moving direction within the optimization process is chosen according to this gradient. Rather than gradient descent, gradient ascent is applied because it is an activation maximization problem. In the experiments chapter, optimal brain representations for each type of stimuli is represented.

3.4.7 Input Representations for Deep Learning

As mentioned in chapter 2, the goal of brain decoding with fMRI data is understanding the mind of a subject using her/his fMRI recordings. The major assumption of this thesis is that a common representation exists among the fMRI data of different subjects and different experiments. In order to identify these common representations across subjects, deep neural networks are used in this thesis. To analyze the effect of initial representation over the brain decoding performance, three different input representations are fed into the deep architectures. In this section, we will explain the details of three different representations. Obtained results are shared in the chapter 4.

3.4.7.1 Average ROI Intensity Values

The first representation used at the input of deep architecture is the average voxel intensity values over anatomical region of interests defined by Automated Anatomical Labeling (AAL) segmentation. AAL is a digital atlas of human brain developed in [72]. In order to train a deep neural networks over average intensity value of each anatomical region, average intensity value of each region for each 3D image obtained

with fMRI are concatenated and used as an input. Rather than voxel intensity values average ROI intensity values are used to classify cognitive states, because of the high feature-sample ratio of fMRI experiments. This ratio in the experiment used on the scope of this thesis is $\sim$60.

Mathematically, suppose that we are given a large number of brain volumes of fMRI data, recorded during a sequence of cognitive stimulus. Let us denote the intensity values measured at each voxel i at a time instant t of stimulus type s as $x_{is}(t)$, where $i \in \{1, 2, 3, ..., I\}$ represents the voxel index and $t \in \{1, 2, 3, ..., T_s\}$ represents the time index for the given stimulus s. Let us denote an atomic region as $\mathbf{r_j} \in \{\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, ..., \mathbf{r_J}\}$ with a set of voxels spatially included in this region. Note that, regions are mutually exclusive and collectively exhaustive sub-volumes of the brain. Then, for the region $\mathbf{r_j}$ and the stimulus s, average intensity value at time $t = t_c$ is calculated as ,

$$m_{js}(t_c) = \frac{\sum_{i \in \mathbf{r_j}} x_{is}(t_c)}{|\mathbf{r_j}|}. \tag{3.14}$$

As mentioned above, each region of AAL segmentation is used as an element of region set $\mathbf{r_j} \in \{\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, ..., \mathbf{r_J}\}$ in this study. In order to form feature vector $M_s(t_c)$for the time point t_c of the stimulus s, average intensity value $m_{js}(t_c)$ for each region j is concatenated as $[m_{1s}(t_c), m_{2s}(t_c), m_{3s}(t_c)...m_{Js}(t_c)]$. The set of feature vectors $M_s = \{M_s(1), M_s(2), M_s(3), ..., M_s(T_s)\}$ is calculated for each type of stimulus, separately.

Although averaging voxel intensity values over AAL regions summarizes the spatial information for each region, the lack of relational information among the regions is the major handicap of this representation. In order to add the connectivity information into deep neural networks, two types of connectivity features are extracted over the raw data. The first type of connectivity feature is the zero lag Pearson correlations, computed for each pair of anatomic regions. The second type is the mesh model suggested in [73]. By obtaining zero-lag correlation and fully-connected mesh model [73] and feeding them as input to the deep neural network, we aim to observe the effect of pair-wise and global connectivity over deep neural networks, respectively.

3.4.7.2 Pearson Correlation Values

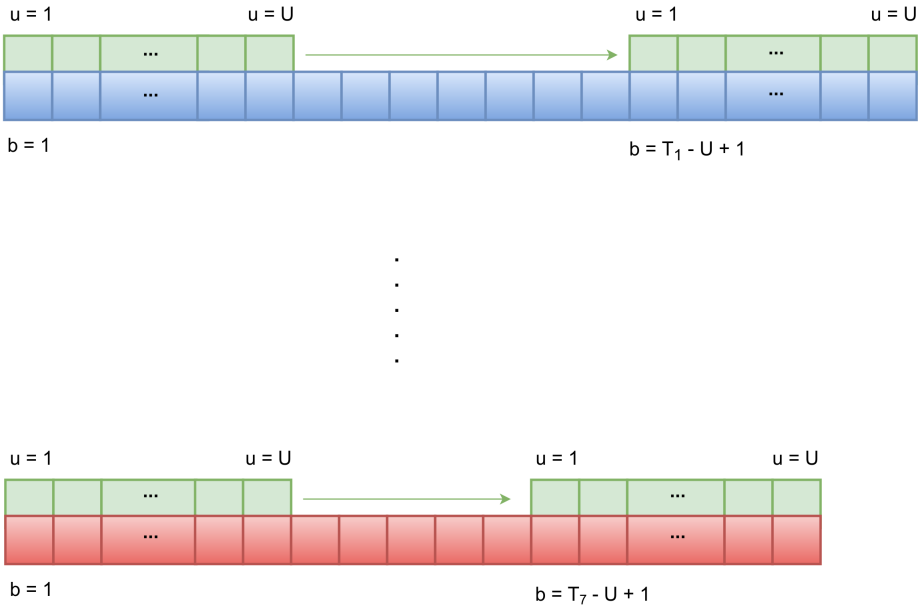


Figure 3.7: Sliding window representation with window size U

Recently, it is shown that relations among voxels or voxel groups are more representative than the raw representation of voxel time series [74]. In order to experimentally observe the effect of connectivity on the performance of deep neural networks, *Pearson* correlation values among the selected AAL regions are employed. *Pearson* correlation between each pair of regions $\mathbf{r}_y$ and $\mathbf{r}_z$ are calculated as

$$p(c_{ybU}, c_{zbU}) = \frac{cov(c_{ybU}, c_{zbU})}{\sigma_{c_{ybU}} \sigma_{c_{zbU}}}, \tag{3.15}$$

where b represents the initial time point of the correlation window and U is the width of the correlation window. c_{ybU} represents the set of average intensity values of region r_y between time points b and $b + U$. As shown in the figure 3.7, correlation values between each pair of regions are calculated by sliding a correlation window with size U for data of each stimulus separately. By combining correlation values between each pair of regions, functional connectivity matrix can be formed. Note that equation3.15 provide us a symmetric functional connectivity matrix, therefore only the elements in the upper-triangle of the functional connectivity matrix are concatenated to form a feature vector. These feature vectors are randomly separated into *training*, *test* and *validation* data set and fed into deep neural networks to classify them. In order to

decode the cognitive state of a subject using samples from other subjects, samples from the same subject can only reside in one of the *training*, *test* and *validation* data set.

3.4.7.3 Temporal Mesh Model

Taking the Pearson correlation between each ROI provide pairwise connectivity information to deep neural networks. To provide a connectivity information extracted among all regions, temporal mesh model is used [73]. To represent each ROI as a linear combination of other ROIs, a mesh connected to all other ROIs are formed for each ROI separately. Center ROI of this mesh is named as the seed ROI. As it is shown in the [45] voxel intensity values change slightly for the same time point t_c. Therefore the, arc weights connecting neighboring ROIs to each seed ROI, $\mathbf{r_s}$ are estimated by a linear model formed among c_{sbU} and each c_{jbU} where $j \neq s$. Note that, these arc weights are found for each sliding time intervals [b, b + U) and for each stimulus s separately. Linear model among the seed ROI and neighboring ROIs are formed as,

$$c_{sbU} = \sum_{j \in J} c_{jbU} \cdot a_{sjb} + \epsilon_{sb}. \tag{3.16}$$

The arc weights, a_{sjb}, are estimated by minimizing the expected square error

$$E((\epsilon_{sb})^2) = E((c_{sbU} - \sum_{j \in J} c_{jbU} \cdot a_{sjb})^2). \tag{3.17}$$

Note that, expectation is taken over the time interval [b, b + U) of the ROI intensity values. The set of arc vectors for each neighboring regions which can be defined as $\mathbf{a_{sb}} = [a_{s1b}, a_{s2b}, ...a_{sJb}]$ is computed using the following closed form ridge regression equation

$$\mathbf{a_{sb}} = (R_{sb}^T R_{sb} + \lambda I)^{-1} R_{sb}^T c_{sbU}, \tag{3.18}$$

where λ is the regularization parameter and founded by cross validation in the training

phase. Finding a_{sb} for each s and concatenating them, we obtain the feature vector for the starting time point b to the input of the deep learning architecture. Getting feature vectors for each b, we obtain the feature vectors for the related stimulus. Note that, this procedure is followed for each stimulus separately. Details of temporal mesh model can be found in [73].

3.4.8 Two Major Goals of the Thesis

After obtaining the above initial input representations, to learn a more compact and general hierarchical representation the unsupervised deep neural network is used in this thesis. Two main goals are reached by using deep neural networks.

The first goal is to achieve multi-subject brain decoding using deep neural networks. It is well-known that, deep neural networks perform much better than the traditional machine learning algorithms provided that there is statistically sufficient amount of data which is received from multiple subjects [17]. This fact is empirically shown in our experiments where we compare deep neural networks to the popular machine learning algorithms in the neuroscience literature, such as, K-NN and SVM. In order to perform multi-subject brain decoding, formed deep neural network is initially trained without any label information in an unsupervised fashion. Then, pre-trained deep neural networks are fine-tuned with the label information included in the fMRI experiment. A variety of architectures are employed to classify cognitive states using fMRI data. Results are compared in the Chapter 4.

Second goal of this thesis is to show that a common representations among different fMRI experiments can be detected by deep neural networks. In order to transfer hierarchical representation among fMRI experiments an unsupervised deep neural network, called stacked denoising autoencoder is employed. Representations learned from an fMRI experiment in an unsupervised fashion is used to obtain hierarchical representations for another fMRI experiment. This goal is achieved by the features extracted for the second fMRI experiment is fed into the deep neural network pre-trained with the unlabeled samples of first fMRI experiment. Pearson correlation input vectors are employed for this goal. Then, hierarchical representations are obtained for the second fMRI experiment. Samples are represented with these features

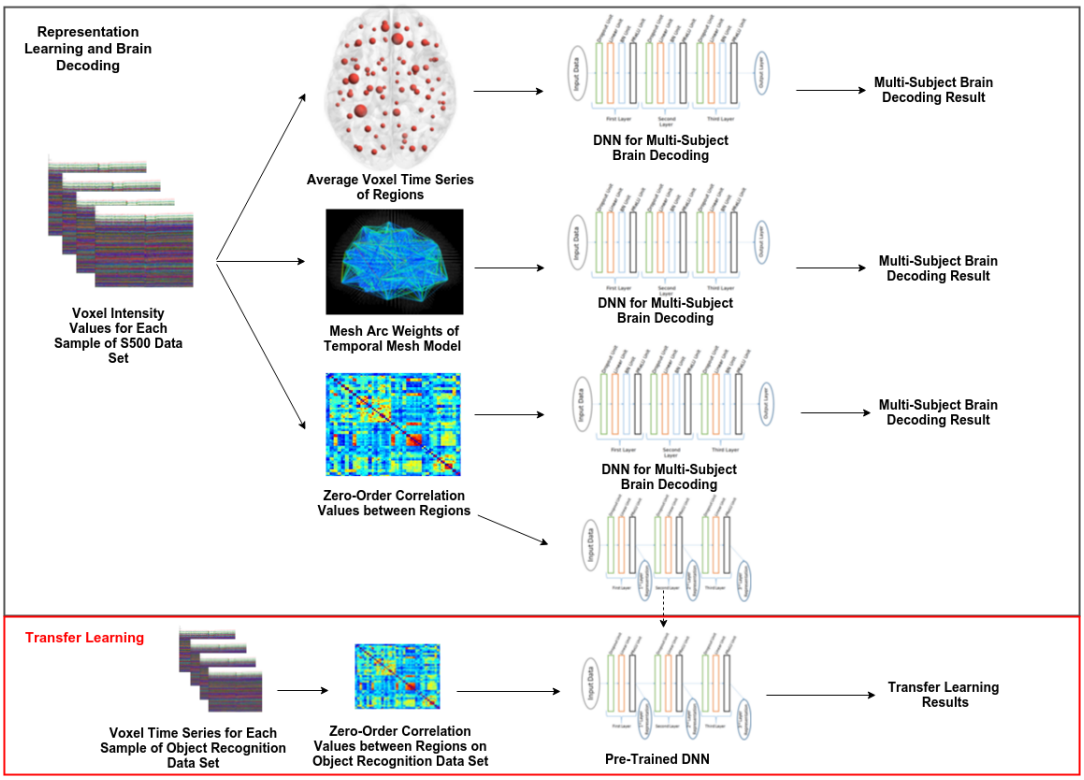


Figure 3.8: Block Diagram of the overall brain decoding framework suggested in this thesis. First, voxel intensity values are mapped into three different representations. Then, by using these representations as an input to deep neural networks, multi-subject brain decoding results are obtained. Besides, pre-trained deep neural network with S500 data set is also used for transfer learning. To implement this, pearson correlation values extracted from object recognition data set fed into the pre-trained neural network. Then cognitive states of object recognition task are classified using hierarchical representations obtained from this network. Details of the S500 and object recognition data sets are explained in Chapter 4.

to classify them. Results show us that, statistically common patterns can be transferred among different fMRI experiments. Results are given in the Chapter 4.

A block diagram of the overall brain decoding system, suggested in this thesis is given in the Figure 3.8. As represented in this block diagram, multi-subject brain decoding is implemented using two different data set. As a pre-processing step, three different representations are extracted from the raw voxel intensity values of these two data set. Then, multi-subject brain decoding results are obtained by using these features as an

input for deep neural networks.

3.5 Chapter Summary

In this chapter, both biological and computational foundations of deep neural networks are studied in brief. We explain the essence of deep neural networks utilized in this thesis and the implementation of them on brain decoding problem. Because recent deep learning techniques are based on the forward propagation and backpropagation techniques, we discussed about the basic formalism of these algorithms. Then we discuss the initialization, regularization, optimization and visualization techniques applied over deep neural networks for brain decoding. Finally input representations and two major goal of the thesis are explained.

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter, experimental results of brain decoding obtained by the deep learning techniques suggested in the chapter 3 are provided. In order to examine the performance of these techniques comprehensively, two different dataset are utilized. In this chapter, first these data sets will be explained. In order to examine the effect of input type over the performance of deep neural networks on brain decoding, three different initial representation are employed. Therefore, comparison of different types of representations given as an input to the formed deep neural networks are analyzed. Then, performance of supervised networks on a single data set will be examined using different types of regularization and optimization techniques which were explained in the chapter 3. Afterwards, effect of transfer learning approach is investigated on the brain decoding domain by transferring information between two different data sets using unsupervised deep learning techniques given in the section 3.3.2. Finally, the deep neural networks are visualized for different types of stimuli to provide a probabilistic activation map for human brain under different cognitive states.

4.1 Experiments and Data Sets

In the fMRI data collection phase, the experimental design performed by the neuroscience community is the vital part of fMRI studies. Each stimulus must be chosen carefully to observe its effect over the BOLD signal. In this thesis, we employed two different fMRI data sets, namely the object recognition data set and Human Connectome Project S500 data set [22]. In the following subsections, the experimental

set-ups of these data sets will be provided.

4.1.1 Object Recognition Data Set

The first dataset used in this study is obtained by our study group [1]working cooperatively with psychology researchers from Koç University. 6 different subjects are exposed to a memory experiment inside the fMRI machine. While representing different types of stimuli inside the fMRI machine, subject's attention must be kept high. In order to keep the attention, n-back paradigm is chosen as an experimental paradigm. In this paradigm subjects are asked to remember each stimulus represented within the last n stimuli and give a response, if the current stimulus is included among the n stimuli. n can be adjusted to change the difficulty of the experiment.We need to avoid a pure working memory experiment. Since our goal is to decode the stimulus type according to the obtained fMRI data, n is chosen as 1. In the object recognition task, the stimuli consist of gray-scale images. Subjects are asked to discriminate images from two different categories, namely flowers and birds. Subjects are shown each stimulus for 4 seconds. Then, there is a fixation period of 8, 10 or 12 seconds between each consecutive stimuli to minimize the effect of previous stimulus on the BOLD signal of the represented stimulus. Experiment is performed in 6 runs. Total of 36 measurements were recorded in each of 6 runs to obtain 216 fMRI images i.e. samples. At each run, first 12 samples per each class are used for training and the 6 remaining samples are used for testing in the transfer learning experiments. SPM8 toolbox [75] is used for pre-processing recorded fMRI images. Realignment of images is followed by co-registration to an anatomical image which is also acquired as a part of the experiment. Then, images are spatially smoothed with a 10-mm FWHM isotropic Gaussian kernel.

4.1.2 Human Connectome Project HCP500 Data Set

It is known that increasing the layer count of deep neural network increases the number parameters exponentially. Because of the large amount of parameters in the deep

[1] http://neuro.ceng.metu.edu.tr

Table 4.1: Number of samples for each type of stimulus.

	Images per Run	**Run Duration (sec)**
Working Memory	405	301
Gambling	253	192
Motor	284	214
Language	316	237
Social Cognition	274	207
Relational Processing	232	176
Emotion Processing	176	136

neural networks, ample amount of samples is necessary for training them. Otherwise such a learning model can easily overfit to the training samples. HCP500 dataset gathered as a part of the Human Connectome Project [22], is used to train the deep neural networks formed in this thesis to overcome overfitting problem. Although structural MRI and resting state fMRI data are also provided in the dataset, only task-evoked fMRI (tfMRI) data is utilized. tfMRI data consists of 500 healthy adult subjects collected in the scope of HCP. In this thesis, pre-processed images of 97 subjects are used to train the suggested deep neural networks due to our limited computational power. tfMRI images of each sample is passed over pre-processing steps explained in the section 2.2. Each subject is exposed to seven different types of stimulus namely; working memory (WM), gambling (GB), motor (MT), language (LG), social cognition (SC), relational processing (RP) and emotion processing (EP). 1940 fMRI images are employed from each subject under these 7 types of stimuli. For each type of stimulus data is collected within only one run. In table 4.1, number of samples for each type of stimulus is given. Note that, the data is collected from each of the 97 subjects. Therefore, totally 188180 fMRI images are enjoyed in classification experiments. Although such amount of samples are collected within HCP500 data set, having ~ 150.000 voxel per sample forces research to make supervoxel-based analysis. Experimental paradigm of each experiment can be studied from [76]. In this thesis super-voxels are selected as the ROIs of AAL regions.

4.2 Brain Decoding on HCP500 Data Set

As mentioned in section 4.1.2, HCP500 data set consists of samples which belong to seven different cognitive task categories. Goal of experiments applied on the HCP500 data set is decoding the cognitive state of test subjects using labeled tfMRI images of training subjects. When the 3D sequential brain volumes of each subject is separated into training and testing samples, a simple classifier, such as linear support vector machine explained in section 2.3.3 gives classification result of 100%. However, combining the subjects in both data sets decreases classification performance significantly as explained in section 4.2.1. These results show us that non-linear mapping exist between the fMRI images of different subjects.

4.2.1 Decoding Raw Data

Classification performances results of [1] given in table 4.2 show that the samples from different subjects do not share the same feature space with each other. As it can be seen from table 4.2, training a linear classifier; logistic regression or support vector machine with linear kernel over multi-subject fMRI samples results with the classification performance of the chance level. Note that, samples from WM task comprise $\sim 20.87\%$ of the data. Because of the failure of linear classifier on multi-subject brain decoding, non-linear classifiers; radial basis function kernel - SVM (RBFSVM) and deep neural network are applied over the same data. In order to apply these architectures over the multi-subject data, average voxel time series of each region for each subject are obtained with fMRI are used in a vector form as a feature vector. This vector is given as the input to these classifiers as explained in subsection 3.4.1.

Table 4.2: Multi-Subject Brain Decoding Using 3D Images of HCP S500 data set. [1]

Type of Classifier	**Architecture**	**Mean Accuracy**
Logistic Regression	116-7	20.81
Support Vector Machine	116-7	20.87
RBF Kernel - Support Vector Machine	116-7	47.97
Deep Neural Network	116-500-7	48.94
Deep Neural Network	116-500-500-7	**50.74**
Deep Neural Network	116-500-500-500-7	50.57

In the table, first parameter of each classifier's architecture represents the number of features included in the input. Since average voxel intensity value of each ROI is used as a feature, first parameter of each architecture is 116. Last parameter represents the number of output classes, which is 7 for HCP S500 data set. For deep neural networks, numbers at the middle of these two numbers are the number of hidden units located in the hidden layers. Architectural detail of classifiers are given in the [1]. Although in reference [1] classification experiment is performed using all 116 AAL regions, during our experiments J is taken as 98. In other words, 98 regions of 116 AAL regions are used. Regions spatially included in the *Cerebellum* are omitted due to their irrelevance in the brain decoding tasks.

4.3 Including Connectivity Information

As mentioned in subsection 3.4.1, decoding cognitive states using voxel intensity values doesn't provide connectivity information among the regions. Recall that in Chapter 3, we form two connectivity features. First one was Pearson correlation between the anatomic region pairs, and the second one was the mesh network in the neighborhood of each anatomic region. Feature vectors are obtained as explained in the subsection 3.4.7. Implementation details of brain decoding with connectivity information is explained below.

4.3.1 Decoding Correlation Data

First, model selection process is explained in order to provide a complete analysis of applying deep neural networks on brain decoding tasks with connectivity information. As it is known from the chapter 3, vanishing gradient is one of the most important problems of the deep neural networks. In order to solve this problem, empirical analyses are performed with a variety of activation functions, optimization techniques and regularization methods. As mentioned in section 3.2.2, unsupervised learning is also used to initialize weights of deep neural networks.

In this thesis, each deep neural network is pre-trained with stacked denoising autoencoder before using it as a classifier. Because correlation values are real-valued, least

squares method is used as a cost function of unsupervised neural networks. Training a deep neural network refers to two stage process in this thesis. In the first stage, weights between the layers are initialized using unsupervised learning. At the second stage, weights are fine-tuned with a supervised cost function. Although pre-training supervised deep neural network in an unsupervised fashion improves the generalization performance, the type of activation functions, optimization methods and regularization methods also play critical roles on the quality of obtained representations. In this thesis, those hyper-parameters are analyzed and validated empirically.

Since the early deep neural network studies use sigmoid (or tanh) function as non-linear activation function of hidden units [54], primarily tanh is used as an activation function. Although various initialization and regularization methods are applied during the training phase of deep neural networks, the high dimension of input feature space causes hidden units of upper layers to saturate, which results in vanishing gradient problem. Sample of the saturation on the 3^{rd} layer of the deep neural network is shown in the figure 4.1, which represents the histogram of activation values generated from a random batch at 6^{th} epoch of training. As it is given in the figure, most of the units are saturated in the 3^{rd} layer. Since most units of the uppermost hidden layer are saturated, they can not propagate the gradient back to lower layers. In this figure, the architecture of the network consists of an input layer with 4753 inputs, 3 hidden layers with 3000,2000 and 1000 units, respectively. Finally, 3^{rd} hidden layer is connected to output layer with 7 category label units. Also note that, optimization techniques with global learning rate always results with worse performances than per-parameter learning rate techniques. Therefore, to avoid the learning rate selection problem, ADADELTA is used as an optimization technique for the experiments.

In order to solve the saturation problem, rectified linear units explained in Chapter 3 are used as an activation function of hidden units of the network. PReLU function is selected as the activation function. This function is parametric version of the ReLU function as shown in the chapter 3. It adds the capability of having a gradient within the domain smaller than zero and doesn't break the rule of making the representation sparse, which is a known characteristic of the brain. Although weights are initialized to very small values by scaling the Glorot initialization [77] with 0.1 and 0.01, great amount of parameters and varying characteristics of samples force the system to have

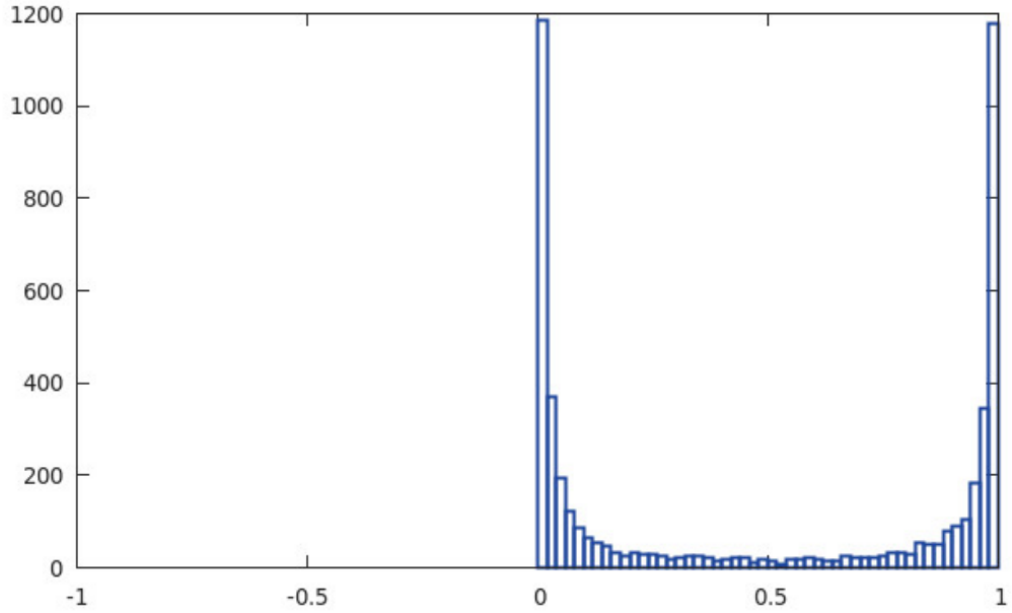


Figure 4.1: Histogram of activation function of 3^{rd} layer hidden units with sigmoid activation function. Input is the 4753 correlation values obtained between the pairs of 98 ROIs of HCP S500 data set.

the representation non-sparse. As it can be seen from the figure 4.2, a large number of activation units have greater or smaller values than zero, that causes the system to overfit fast. As it is shown in the [17] extracting sparser representations increase the brain decoding performance.

Although large output values at the hidden layer cause system to overfit to training samples, histogram of output of activation function of 4.2 shows that PReLU activation function saturates less than tanh (or sigmoid) activation function with pairwise pearson correlation values of 98 regions.

In order to solve the overfitting problem, a recently developed technique called Batch Normalization [69] is used. As explained in chapter 3, the goal of batch normalization is to decrease the internal covariate shift to improve the convergence speed. It also prevents the architecture from overfitting when the learning rate is high. When the same data set is trained by adding a batch normalization layer, representation becomes more sparse. Histogram of the output of activation function of 2^{nd} layer when the batch normalization is applied is shown in the figure 4.3. Note that histograms given in the both figure 4.2 and 4.3 are obtained from a random batch in the 6^{th} epoch.

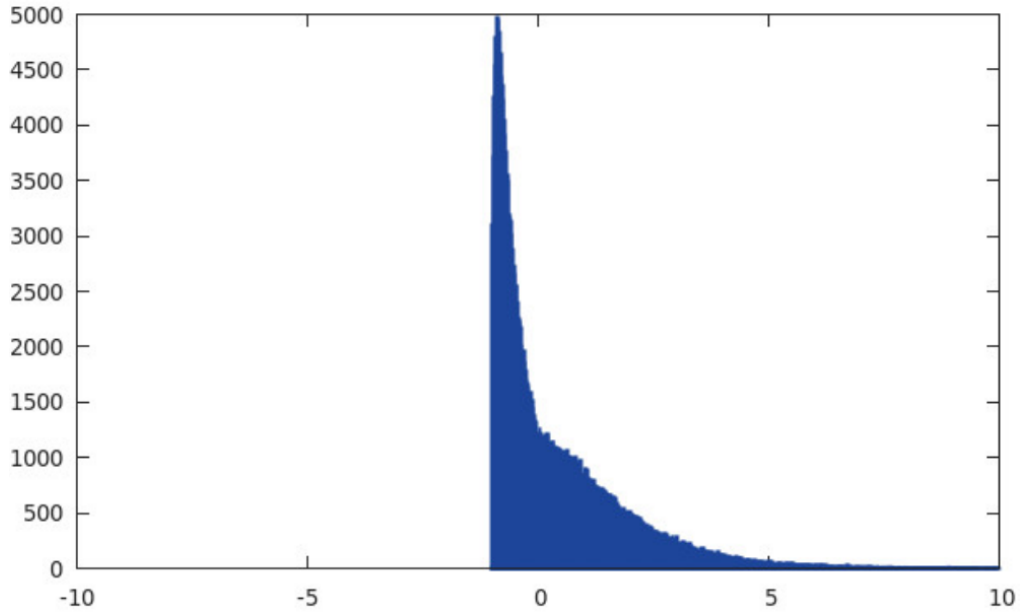


Figure 4.2: Histogram of activation function of 2^{nd} layer hidden units with PReLU activation function. Input is the 4753 correlation values obtained between the pairs of 98 ROIs of HCP S500 data set.

Due to representations most consistent with neurological findings are obtained using the latest network configuration, it is chosen as the architecture in the experiments performed in this thesis. Deep neural network architecture used in this thesis is given in the figure 4.4. As it can be seen from the figure, features pass through dropout unit, linear unit, batch normalization unit and PReLU unit three times.

Although it is shown in [1] that deep neural networks can decode the cognitive stimulus better than classifiers which do not create hierarchical representations between output and input layers, we apply the same analysis to understand whether it is also valid for the correlation data. Goal of this analysis is to show that both spatial and temporal hierarchies exist in the fMRI data. To obtain results, 5 random sampling is used to determine *training*, *test* and *evaluate* data sets. Note that, 80% of subjects are used as *training* set for each experiment. Remaining samples are divided into two subsets to form *test* and *validation* data sets.

Decoding performances given in table 4.3 is obtained by averaging the result of 5 different experiments. As shown in the table, decoding the stimulus with DNN architecture results with much better performance than SVM classifier, which is the most commonly applied classification algorithm in the neuroscience literature. Ex-

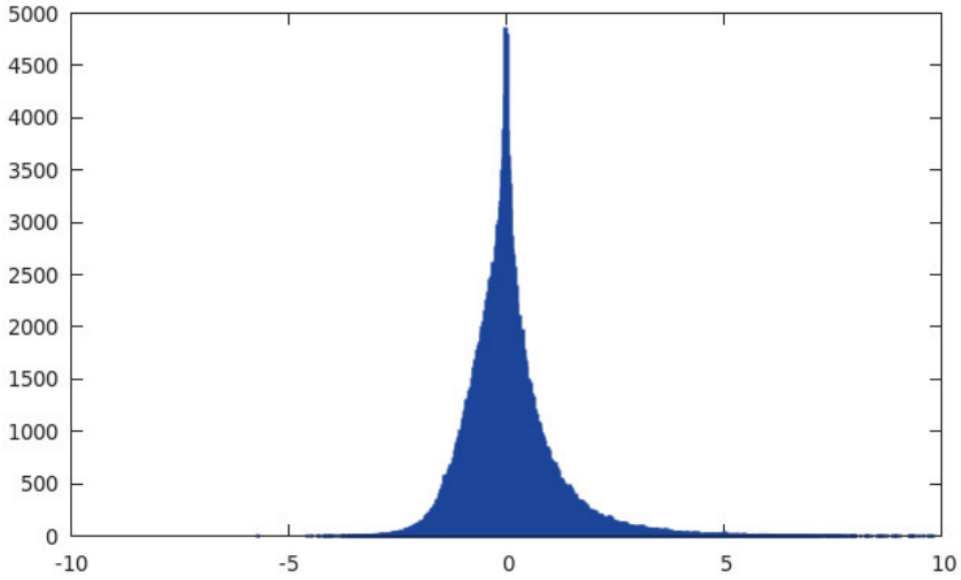


Figure 4.3: Histogram of activation function of 2^{nd} layer hidden units with PReLU activation function and Batch Normalization. Input is the 4753 correlation values obtained between the pairs of 98 ROIs of HCP S500 data set.

Table 4.3: Decoding the Cognitive task of S500 Data Set by Using Correlation Data With and Without Dropout.

Correlation Data	Window Size		
	60	90	120
Correlation + 2 Level DNN	87.2	89.2	90.4
Correlation + 3 Level DNN	88.9	91.1	**91.6**
Correlation + 4 Level DNN	86.6	88.3	89.1
Correlation + 2 Level DNN (without Dropout)	81.6	84.6	86.0
Correlation + 3 Level DNN (without Dropout)	84.4	88.5	90.4
Correlation + 4 Level DNN (without Dropout)	81.8	83.6	85.6
Correlation + SVM	76.5	81.3	83.6
Correlation + KNN	68.4	74.5	78.9

periments are applied with 2, 3 and 4 layered-DNNs. Architecture with 3 hidden layers results with the best performance with Pearson correlation vector.

Although type of activation function and optimization technique is determined by investigating the characteristics of the activation histograms, it is unfeasible to select regularization methods using only training data. In order to decide on whether to use regularization methods over the network or not, *validation* data set is utilized. Using the cross-validation technique coefficients γ_{L1} and γ_{L2} for L_1 and L_2 regu-

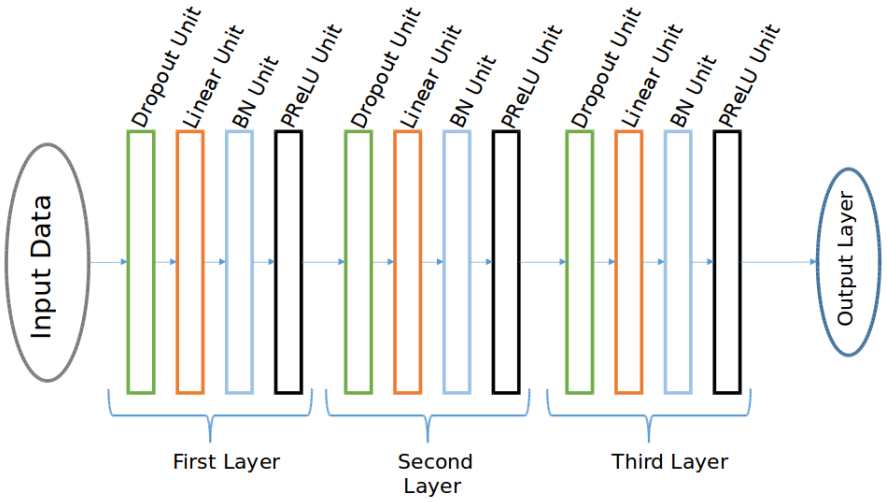


Figure 4.4: 3-Layered Deep Neural Network Architecture.

larization methods are found as 0.1 and 0.1. Logarithmic search space is used for cross-validation. The effect of Dropout on the performance of deep neural network is investigated by training the deep neural networks with and without dropout mechanism. Again by cross-validation it is observed that 0.9 dropout ratio for input layer and 0.1 dropout ratio for hidden layers provides best brain decoding results. That result is consistent with the noisy nature of the fMRI data. Note that for both input and hidden layers dropout ratio of $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ are tested. Classification performances with and without dropout mechanism can be observed from the table 4.3. Samples from all of 97 subjects are used to obtain results given in table 4.3.

In order to analyze chosen deep neural network architecture over the correlation data with different subject counts and different window sizes, 12 different experiments are applied over the HCP500 data set. Results are shared in the 4.4.

As shown in the table 4.4, to be consistent with the literature on the correlation analysis of fMRI data [78], windows with 60, 90 and 120 samples are used to obtain *Pearson* correlation values between each anatomical region. Since the correlation values are calculated independently at each stimulus, increasing the window size U can result with the shortage of samples from EP stimulus. In order to analyze the effect of subject counts on the brain decoding performance, same experiments are re-performed using 40, 60, 80 and 97 subjects.

As it can be seen from the results on the table 4.4, increasing both number of subjects

Table 4.4: Decoding the Cognitive Tasks Using Pearson Correlation Input Vector With Varying Subject Counts and Window Sizes on HCP500 Data Set.

SubjectCount/WindowSize	**60**	**90**	**120**
40	83.96	87.55	88.28
60	86.21	88.49	89.21
80	86.45	89.53	90.75
97	88.92	91.1	**91.64**

and length of correlation window monotonically increase the brain decoding performance.

As it can be seen from the table 4.3, DNN classifies cognitive states much better than SVM and KNN methods. Yet it is known that the connectivity among the brain regions can be represented by a more general model compared to Pearson correlation. As mentioned before, mesh networks model the connectivity among supervoxels considering both locality and distributivity properties of the brain. Note that, AAL regions are used as a supervoxel in this thesis. To make our model benefit the mesh network, the mesh arc weights among the anatomic regions are also fed as an input to the deep neural networks. Since relation among each ROI is captured with mesh arc weights, we called them as global connections. Note that, the architecture parameters which provides best results using the pearson correlation vectors are used as an architecture for classifying global connections into cognitive states.

4.3.2 Decoding the Mesh Model Weights

Although classifying cognitive states employing zero-order correlation provides better performance than single time point intensity values, it is known that using global relations between supervoxels improves the brain decoding performance [73]. In order to utilize global relations within formed architecture, *mesh model* [45] is used to obtain linear relation among supervoxels as explained in the section 3.1. Note that supervoxels are again chosen as 98 ROIs of AAL regions.

After obtaining the linear coefficients between each supervoxel, these are fed into deep neural networks as an input. It is known that coefficients of linear relations

Table 4.5: Decoding Temporal Mesh Weights with number of architectures and classifiers.

	Classification Result
Mesh Weights + 2 Level DNN	94.4
Mesh Weights + 3 Level DNN	95.6
Mesh Weights + 4 Level DNN	**96.5**
Mesh Weights + 2 Level DNN (without Dropout)	92.6
Mesh Weights + 3 Level DNN (without Dropout)	94.8
Mesh Weights + 4 Level DNN (without Dropout)	95.3
Mesh Weights + SVM	90.9

carry more information than pearson correlation values [73]. This implies that less number of units are used in hidden layers. The best classification performance is obtained with four-hidden-layered deep neural network as it can be seen in the table 4.5. $400, 300, 200, 100$ units are used in hidden layers, respectively. We observe that 4-layered network provides better results than 3-layered network using mesh weights. In our opinion global connections provides more complex hierarchy than Pearson correlation. Note that, results are obtained by averaging 5 different experiments performed with randomly sampled *training, test* and *validation* data sets. All of the samples from 97 subjects are used in these experiments.

As it is observed from the experiments applied in section 4.2, dropout layers with high ratio is added after the input layer. As in the experiments conducted using Pearson correlation input vectors, the dropout ratio of 0.9 in mesh model input layer results with the best performance. Remaining architecture options are same with options given in the section 4.2.

In the neuroscience literature, it is very important to specify the region of interests for performed cognitive task. As mentioned in the subsection 3.4.6, to identify these regions probabilistic approach is chosen. In order to find the active regions for the particular cognitive state, global connections which are most important for pre-trained network is found by saliency analysis. Because mesh weights provide us the global connections among each ROI, saliency analysis is applied over mesh weights. In the images from 4.5 to 4.11, linear relations that maximize the related cognitive state are given. Name of anatomical regions that play important role to obtain high output

values for different types of cognitive states are given at the end of this chapter. Represented 50 mesh arc weights are determined according to absolute values of weights.

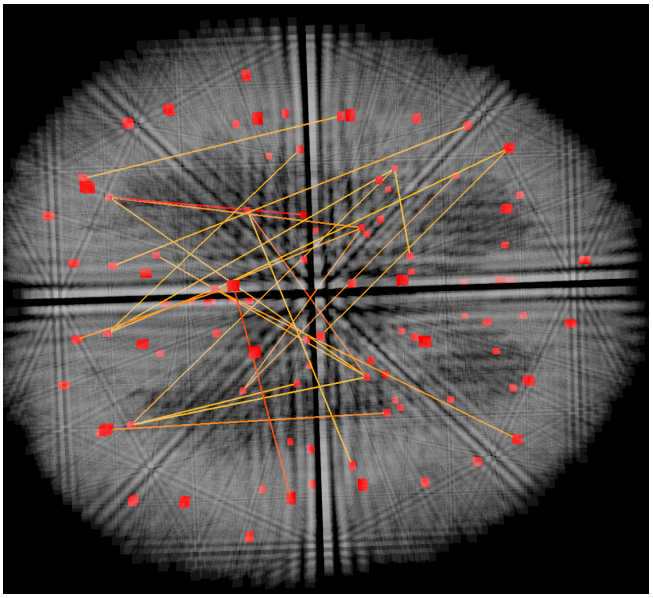


Figure 4.5: 50 mesh arc weights with highest absolute values for WM stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for WM stimulus are found by doing a saliency analysis over pre-trained neural network.

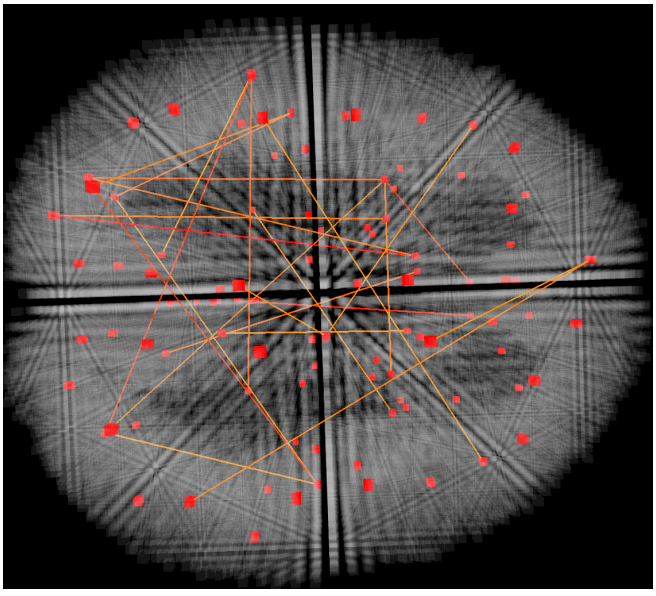


Figure 4.6: 50 mesh arc weights with highest absolute values for GB stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for GB stimulus are found by doing a saliency analysis over pre-trained neural network.

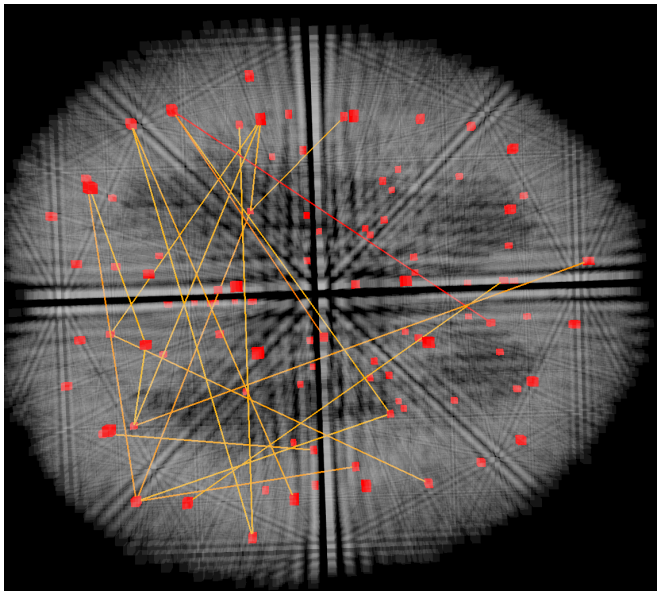


Figure 4.7: 50 mesh arc weights with highest absolute values for MT stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for MT stimulus are found by doing a saliency analysis over pre-trained neural network.

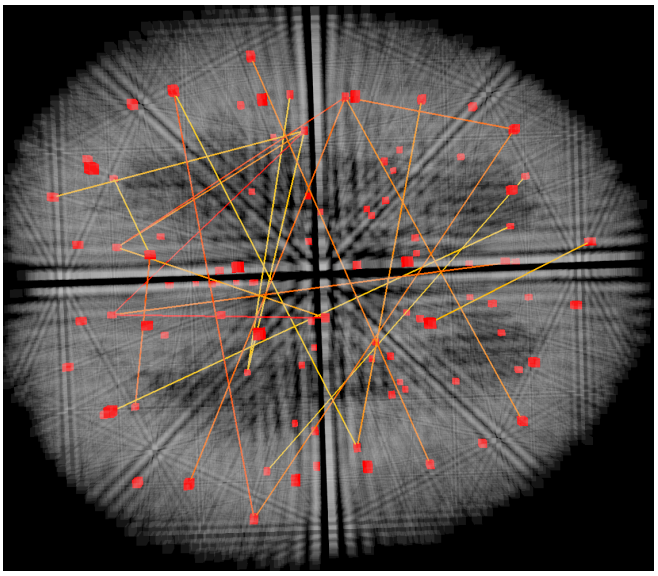


Figure 4.8: 50 mesh arc weights with highest absolute values for LG stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for LG stimulus are found by doing a saliency analysis over pre-trained neural network.

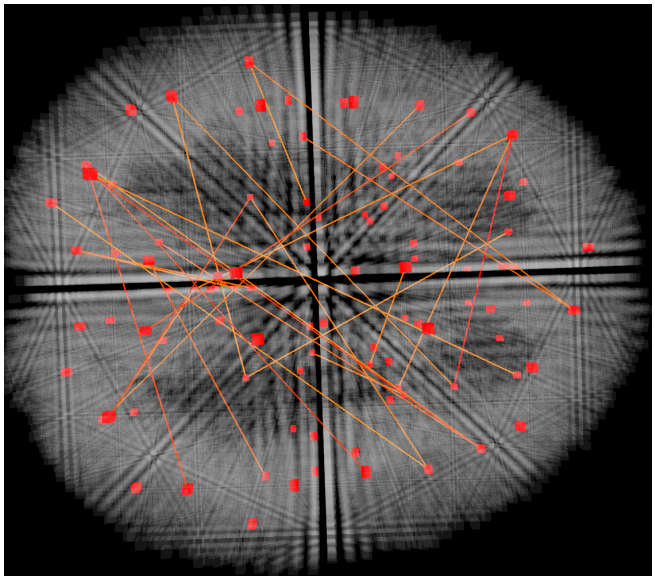


Figure 4.9: 50 mesh arc weights with highest absolute values for SC stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for SC stimulus are found by doing a saliency analysis over pre-trained neural network.

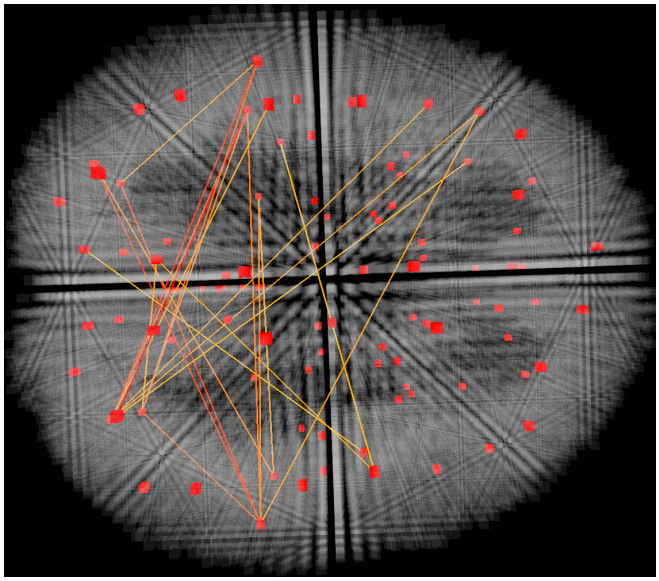


Figure 4.10: 50 mesh arc weights with highest absolute values for RP stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for RP stimulus are found by doing a saliency analysis over pre-trained neural network.

4.4 Transfer Learning with HCP500 and Object Recognition Data Set

Beside multi-subject brain decoding, we also transfer common representations among different experiments with deep neural networks.

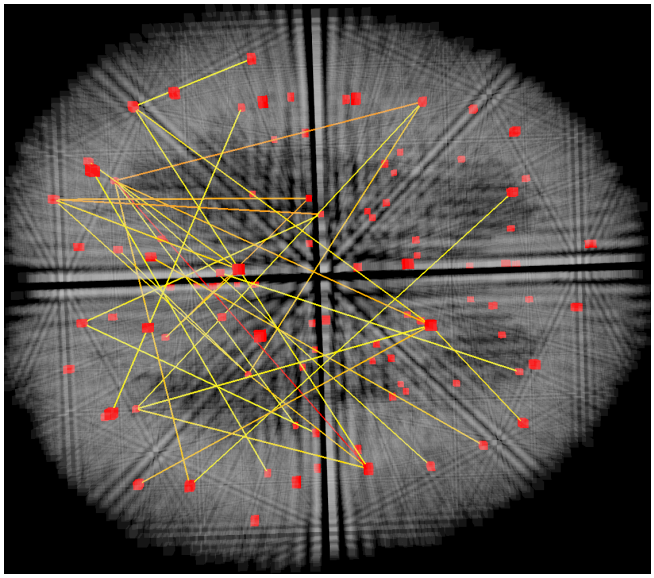


Figure 4.11: 50 mesh arc weights with highest absolute values for EP stimulus. Connections are obtained by applying temporal mesh model over S500 data. 50 connections for EP stimulus are found by doing a saliency analysis over pre-trained neural network.

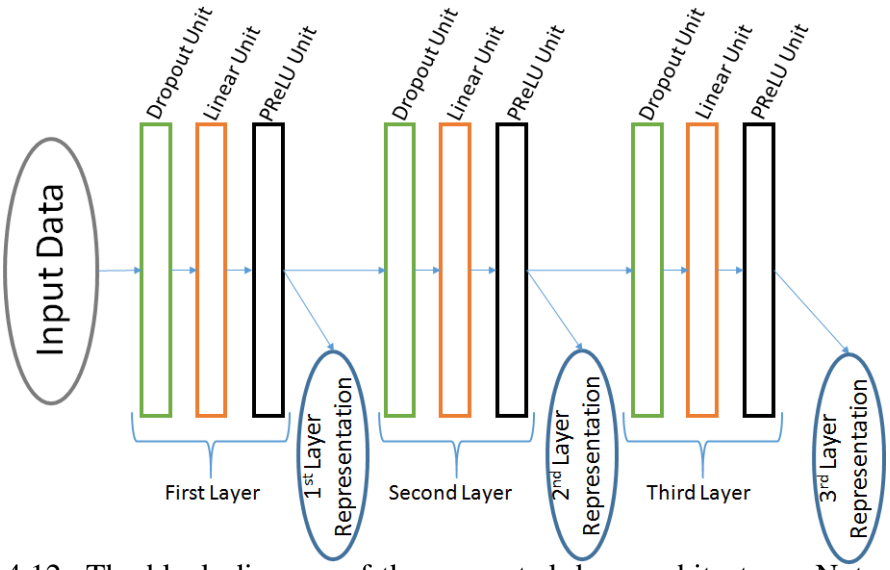


Figure 4.12: The block diagram of the suggested deep architecture. Note that this architecture is trained in an unsupervised fashion.

In order to obtain a hierarchical and abstract representation from HCP500 data set, deep neural network with three layers is trained with the pearson correlation values obtained from HCP500 data set in an unsupervised fashion. In order to provide $1^{st}, 2^{nd}$ and 3^{rd} layer representations of the output layer of each layer in figure 4.12, unsupervised training approach is utilized layer-by-layer as explained in the Chapter

3. It is known that upper layers of the unsupervised deep neural network provide more abstract representation compared to the lower layers.

One of the major problems in transfer learning is the variations between the learning and recognition data sets. The experimental setup of HCP500 data set, where we use to learn the representation and object recognition task data set where we recognize the cognitive states, are quite different from each other as explained at the beginning of this chapter. The learned representations must handle the variances between these two data sets. In order to block the experimental co-adaptation of features within the architecture, a denoising operation is applied on each layer. Setting a random subset of each layer's input to zero, denoising operation provides the system to learn abstract representations, which are affected essentially from the factors common in each fMRI experiments. Using the samples from 7 different tasks as the training set in the experiments, enables us to find the representations independent from the task type.

Although parameters of the unsupervised network can be optimized with ADADELTA, the hyper-parameters of the architecture can not be optimized with gradient-based methods. In order to find the best value for the hidden neuron counts and the denoising level u^i at each layer, a grid search is applied by implementing cross-validation. For each layer, $H = \{750, 1500, 3000\}$ hidden neuron counts and $U = \{0.1, 0.3, 0.5\}$ corruption levels are searched. The greedy search method yields, $3000, 1500, 750$ and $0.3, 0.1, 0.1$ for hidden neuron counts and corruption levels for each layer, respectively. In this thesis all of the deep neural network learning algorithms are implemented using the Torch 7 framework.

HRF signals obtained in two different fMRI data sets reach the baseline value at different time intervals, depending the type of the stimuli and the nature of the experiment. In order to capture the full range of HRFs, the unsupervised deep neural network is implemented with three different window sizes, used in object recognition task.

In object recognition task, each stimuli is shown to a subject for a time interval of 4 seconds. Then, rest period lasts for 8, 10 or 12 seconds, with TR = 2 seconds. Since the average time of returning to the baseline value is approximately 10 seconds for

Table 4.6: SVM classification results of Pearson correlation inputs with different window sizes of object recognition data set.

Subject ID / n	**4**	**5**	**6**
Subject 1	55.5	57.5	60.5
Subject 2	56	60.5	60.5
Subject 3	61.5	65.5	62
Subject 4	55.5	57.5	58
Subject 5	58.1	54.4	55
Average	57.3	59.1	**59.2**

Table 4.7: SVM classification results using the hierarchical features for the Pearson correlation inputs with U = 4 on object recognition data set.

	$\mathbf{e^1}$	$\mathbf{e^2}$	$\mathbf{e^3}$
Subject 1	61.0	62.5	64.0
Subject 2	61.5	64.0	64.5
Subject 3	64.5	66.0	65
Subject 4	59	58	63.0
Subject 5	63.8	64.7	66.8
Average	62.0	63.0	**64.6**

Table 4.8: SVM classification results using the hierarchical features for the Pearson correlation inputs with U = 5 on object recognition data set.

	$\mathbf{e^1}$	$\mathbf{e^2}$	$\mathbf{e^3}$
Subject 1	64.0	62.5	63
Subject 2	64.5	61	65
Subject 3	65	68	67.5
Subject 4	63	61.5	67
Subject 5	66.7	60	62
Average	64.6	62.6	**64.9**

the HRF, correlation window sizes U are taken as 4,5 and 6 for this task. In order to find Pearson correlation from the windows of same time length for unsupervised representation learning and stimuli classifying processes, 5,7 and 8 are chosen as U values for the HCP500 data set.

In Table 4.6, classification performance using the pearson correlation values with 5 different subjects and 3 different window sizes of $4, 5$ and 6 are shown for object recognition task. Linear SVM classifier is used for each classification task throughout

Table 4.9: SVM classification results using the hierarchical features for the Pearson correlation inputs with U = 6 on object recognition data set.

	$\mathbf{e^1}$	$\mathbf{e^2}$	$\mathbf{e^3}$
Subject 1	63	64	64.5
Subject 2	59	60.5	63.5
Subject 3	65.5	66	67
Subject 4	58	63.5	60.5
Subject 5	61.3	63.2	64.4
Average	61.4	63.4	**64.0**

Table 4.10: SVM classification results using the representation obtained with PCA when window size U = 4.

	750	**1500**	**3000**
Subject 1	54.0	56.5	56.0
Subject 2	53.0	52.5	53.0
Subject 3	60.0	59.5	60.0
Subject 4	51.0	51.0	51.5
Subject 5	56.0	55.5	57.5
Average	54.8	55	**55.6**

the experiments. In Table 4.7, 4.8 and 4.9 classification results with learned representations are shown. Note that e^1, e^2 and e^3 represent the 1^{st}, 2^{nd} and 3^{rd} layer representation of unsupervised deep neural network, respectively. As it can be seen from these tables, approximately 5% increase is observed when the learned representations are used to compared pearson correlation values obtained from object recognition data set, for each subject. These results indicate that the information in the learned representation is transferred to a different data set successfully. Note also that, the

Table 4.11: SVM classification results using the representation obtained with PCA when window size U = 5.

	750	**1500**	**3000**
Subject 1	56.0	57.5	57.0
Subject 2	54.0	52.5	54.0
Subject 3	63.5	64.5	60.5
Subject 4	56.0	55.0	54.5
Subject 5	55.5	56.0	55.0
Average	57	**57.1**	56.2

Table 4.12: SVM classification results using the representation obtained with PCA when window size $U = 6$.

	750	**1500**	**3000**
Subject 1	60.5	60.5	61.0
Subject 2	56.5	54.0	55.0
Subject 3	64.0	63.5	62.5
Subject 4	55.0	52.5	53.0
Subject 5	51.5	49.0	53.0
Average	**57.5**	55.9	56.9

Table 4.13: SVM classification results using the representation obtained with ICA when window size $U = 4$.

	750	**1500**	**3000**
Subject 1	56.5	51.5	55.0
Subject 2	55.0	53.5	53.5
Subject 3	63.0	59.0	62.5
Subject 4	54.0	51.5	52.0
Subject 5	60.5	58.0	55
Average	**57.8**	54.7	55.6

Table 4.14: SVM classification results using the representation obtained with ICA when window size $U = 5$.

	750	**1500**	**3000**
Subject 1	52.0	55.5	54.5
Subject 2	54.0	53.0	53.0
Subject 3	64.0	63.0	64.0
Subject 4	57.0	54.5	57.0
Subject 5	55.5	58.0	56.0
Average	56.5	56.8	**56.9**

Table 4.15: SVM classification results using the representation obtained with ICA when window size $U = 6$.

	750	**1500**	**3000**
Subject 1	60.0	61.5	59.5
Subject 2	60.0	56.0	54.5
Subject 3	63.5	59.0	61.0
Subject 4	55.0	52.0	55.5
Subject 5	50.0	54.5	50.5
Average	57.5	**57.7**	56.2

classification performance of 3^{rd} layer representation is the best on the average for each window size.

To compare the transfer learning results obtained using deep neural networks with commonly used techniques, common representations are also obtained with PCA and Group-ICA [79]. For a fair comparison In order to transfer information from HCP500 experiment to object detection experiment, pearson correlation feature vectors obtained from object detection experiment are projected into basis obtained from HCP500 data set with PCA and Group-ICA. As it can be seen from the results given in tables 4.10-4.15, transferring information from HCP500 experiment to object detection experiment decreases the generalization performance. Performances obtained with deep neural networks significantly better than these performances. These results show us that deep neural networks are more successful than commonly used methods to detect experiment-independent common representations.

4.4.1 Chapter Summary

Experimental results of brain decoding obtained by deep neural networks and the state of the art machine learning methods such as SVM and K-NN are given and compared in this chapter. To provide what kind of stimuli are decoded in the experiments, chapter started with the explanation of data sets. Then, the suggested architecture is tested by various hyper-parameters. After selecting the best set of architecture parameters, brain decoding experiments are conducted with different types of inputs. As it can be seen from the results deep neural networks are more successful than SVM and Logistic Regression classifiers on brain decoding task. Besides providing a better classification performance on supervised classification tasks, transfer learning can also be performed on fMRI data with deep neural networks, better than commonly used PCA and Group-ICA methods. As shown in the section 4.3, reducing the dimension of small fMRI data set with an unsupervised deep neural network trained using HCP500 data set improves the classification performance on small fMRI data set. As well as providing good brain decoding performance, probabilistic activation map of different kind of stimulus is also obtained. Using these activation maps, consistency between probabilistic activation maps and neuro-scientific findings can be analyzed.

Table 4.16: Name of Important ROIs for the emotion task

ROI Name	Count of Edges Connected
MNI_Parietal_Sup_L.mat	2
MNI_Occipital_Inf_R.mat	6
MNI_SupraMarginal_R.mat	1
MNI_Hippocampus_R.mat	1
MNI_Precentral_L.mat	1
MNI_Frontal_Inf_Tri_R.mat	1
MNI_Postcentral_L.mat	2
MNI_Supp_Motor_Area_L.mat	5
MNI_Amygdala_R.mat	2
MNI_Parietal_Inf_L.mat	1
MNI_Fusiform_R.mat	3
MNI_Amygdala_L.mat	3
MNI_Frontal_Med_Orb_L.mat	1
MNI_Occipital_Inf_L.mat	2
MNI_Cingulum_Ant_L.mat	1
MNI_Occipital_Sup_R.mat	2
MNI_Paracentral_Lobule_R.mat	2
MNI_Frontal_Mid_L.mat	2
MNI_Frontal_Inf_Tri_L.mat	1
MNI_Lingual_L.mat	1
MNI_Supp_Motor_Area_R.mat	1
MNI_Fusiform_L.mat	4
MNI_Calcarine_L.mat	2
MNI_Frontal_Inf_Orb_L.mat	1
MNI_Cingulum_Post_R.mat	1
MNI_Insula_L.mat	1

Table 4.17: Name of Important ROIs for the gambling task

ROI Name	Count of Edges Connected
MNI_Parietal_Inf_R.mat	3
MNI_Rolandic_Oper_R.mat	3
MNI_Calcarine_R.mat	3
MNI_Pallidum_L.mat	1
MNI_Precuneus_L.mat	1
MNI_Calcarine_L.mat	6
MNI_Angular_L.mat	2
MNI_Occipital_Mid_L.mat	1
MNI_Occipital_Sup_R.mat	3
MNI_Vermis_10.mat	1
MNI_Vermis_6.mat	1
MNI_Frontal_Mid_Orb_L.mat	2
MNI_Insula_R.mat	2
MNI_SupraMarginal_R.mat	2
MNI_Temporal_Sup_L.mat	1
MNI_Rectus_R.mat	1
MNI_Frontal_Inf_Tri_L.mat	2
MNI_Frontal_Inf_Oper_R.mat	1
MNI_Heschl_R.mat	2
MNI_SupraMarginal_L.mat	1
MNI_Temporal_Sup_R.mat	1
MNI_Occipital_Mid_R.mat	1
MNI_Occipital_Sup_L.mat	2
MNI_Cingulum_Mid_L.mat	4
MNI_Parietal_Sup_R.mat	1
MNI_Lingual_R.mat	1
MNI_Frontal_Sup_L.mat	1

Table 4.18: Name of Important ROIs for the language task

ROI Name	Count of Edges Connected
MNI_Fusiform_L.mat	1
MNI_Frontal_Inf_Oper_R.mat	2
MNI_Postcentral_L.mat	2
MNI_Heschl_R.mat	2
MNI_Temporal_Sup_R.mat	5
MNI_Cuneus_L.mat	4
MNI_Cuneus_R.mat	3
MNI_Parietal_Inf_R.mat	5
MNI_Precentral_L.mat	1
MNI_Temporal_Pole_Sup_L.mat	1
MNI_Parietal_Inf_L.mat	2
MNI_Cingulum_Ant_L.mat	2
MNI_Occipital_Mid_R.mat	3
MNI_Occipital_Inf_L.mat	2
MNI_Fusiform_R.mat	1
MNI_Angular_L.mat	1
MNI_Occipital_Sup_R.mat	1
MNI_Cingulum_Mid_L.mat	1
MNI_Temporal_Sup_L.mat	2
MNI_Frontal_Mid_Orb_R.mat	1
MNI_Occipital_Sup_L.mat	1
MNI_Parietal_Sup_L.mat	1
MNI_Temporal_Mid_L.mat	2
MNI_Angular_R.mat	2
MNI_Occipital_Mid_L.mat	1
MNI_Insula_R.mat	1
MNI_Frontal_Sup_L.mat	1

Table 4.19: Name of Important ROIs for the motor task

ROI Name	Count of Edges Connected
MNI_Occipital_Inf_R.mat	2
MNI_Cingulum_Mid_L.mat	2
MNI_ParaHippocampal_R.mat	1
MNI_Heschl_R.mat	4
MNI_Calcarine_L.mat	3
MNI_Parietal_Inf_R.mat	4
MNI_Rolandic_Oper_R.mat	4
MNI_SupraMarginal_L.mat	2
MNI_Precentral_L.mat	1
MNI_Calcarine_R.mat	3
MNI_Postcentral_R.mat	1
MNI_ParaHippocampal_L.mat	1
MNI_Precuneus_R.mat	1
MNI_SupraMarginal_R.mat	5
MNI_Frontal_Med_Orb_R.mat	1
MNI_Postcentral_L.mat	1
MNI_Parietal_Inf_L.mat	2
MNI_Rolandic_Oper_L.mat	1
MNI_Frontal_Mid_R.mat	1
MNI_Angular_R.mat	1
MNI_Fusiform_L.mat	2
MNI_Occipital_Inf_L.mat	1
MNI_Frontal_Mid_L.mat	1
MNI_Supp_Motor_Area_L.mat	1
MNI_Frontal_Inf_Oper_R.mat	1
MNI_Cingulum_Ant_R.mat	1
MNI_Frontal_Inf_Oper_L.mat	1
MNI_Occipital_Sup_R.mat	1

Table 4.20: Name of Important ROIs for the relational task

asd	Count of Edges Connected
MNI_Temporal_Sup_L.mat	3
MNI_Parietal_Sup_R.mat	6
MNI_Parietal_Sup_L.mat	6
MNI_Cingulum_Mid_L.mat	3
MNI_Frontal_Inf_Tri_L.mat	1
MNI_Temporal_Sup_R.mat	2
MNI_Calcarine_R.mat	2
MNI_Occipital_Mid_R.mat	1
MNI_Temporal_Mid_R.mat	2
MNI_Frontal_Mid_R.mat	1
MNI_SupraMarginal_R.mat	4
MNI_Parietal_Inf_L.mat	3
MNI_Temporal_Mid_L.mat	1
MNI_Cingulum_Mid_R.mat	2
MNI_Precuneus_R.mat	1
MNI_Frontal_Sup_Medial_R.mat	1
MNI_Cingulum_Ant_L.mat	3
MNI_Pallidum_R.mat	1
MNI_Rolandic_Oper_R.mat	1
MNI_Precentral_L.mat	2
MNI_ParaHippocampal_L.mat	1
MNI_Hippocampus_R.mat	1
MNI_Lingual_R.mat	1
MNI_Precuneus_L.mat	1

Table 4.21: Name of Important ROIs for the social task

ROI Name	Count of Edges Connected
MNI_Cuneus_L.mat	8
MNI_Parietal_Sup_R.mat	5
MNI_Occipital_Mid_R.mat	4
MNI_Parietal_Sup_L.mat	3
MNI_Temporal_Mid_R.mat	3
MNI_Occipital_Inf_R.mat	1
MNI_SupraMarginal_R.mat	4
MNI_Occipital_Inf_L.mat	2
MNI_Occipital_Mid_L.mat	5
MNI_SupraMarginal_L.mat	5
MNI_Cingulum_Mid_L.mat	3
MNI_Insula_R.mat	1
MNI_Cingulum_Mid_R.mat	1
MNI_Cingulum_Ant_L.mat	3
MNI_Cingulum_Ant_R.mat	1
MNI_Cuneus_R.mat	1

Table 4.22: Name of Important ROIs for the working memory task

ROI Name	**Count of Edges Connected**
MNI_Occipital_Inf_R.mat	5
MNI_Occipital_Inf_L.mat	4
MNI_Cuneus_L.mat	5
MNI_Precentral_L.mat	1
MNI_Fusiform_L.mat	5
MNI_Fusiform_R.mat	3
MNI_Temporal_Sup_R.mat	1
MNI_Precuneus_R.mat	2
MNI_Cingulum_Mid_L.mat	1
MNI_Cingulum_Post_L.mat	2
MNI_Precuneus_L.mat	3
MNI_Rolandic_Oper_R.mat	3
MNI_ParaHippocampal_R.mat	2
MNI_Lingual_L.mat	1
MNI_Occipital_Mid_L.mat	2
MNI_Occipital_Sup_R.mat	2
MNI_Frontal_Inf_Oper_R.mat	1
MNI_Cuneus_R.mat	1
MNI_Supp_Motor_Area_L.mat	3
MNI_Occipital_Sup_L.mat	1
MNI_Frontal_Sup_Orb_R.mat	1
MNI_Parietal_Inf_L.mat	1

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this chapter, the results obtained in this thesis will be discussed and analyzed. Future directions will be followed are also presented.

5.1 A Brief Summary

The main goal of this thesis is obtaining a common representation between fMRI data of different subjects and different fMRI experiments. It is known that brain has a hierarchical nature. Inspiring by this fact, deep neural networks are chosen as a model. State-of-the-art brain decoding performances are obtained by using the hierarchical representations extracted with deep neural networks.

One of the major obstacles against the multi-subject brain decoding studies is that different subjects give different responses to the same stimulus as emerging fMRI signals. We believe that obtaining an abstract and hierarchical representations from fMRI data, this obstacle can be overcome. Utilizing the subset of HCP500 data set with 97 subjects, deep neural networks are trained. Using unsupervised learning approach "denoising autoencoders", formed deep neural networks are pre-trained with three different initial representations. Note that, hyper-parameters of these deep neural networks are found by cross-validation technique.

Hierarchical representation obtained by using deep neural networks are used for two major tasks. For the first task, multi-subject brain decoding on the HCP500 data set, these pre-trained networks are fine-tuned with the labels of HCP500 data set. As an

input representation three different representations are used, namely average ROI intensity values, Pearson correlation values and mesh arc descriptors. Note that, brain decoding results with average ROI intensity values are obtained by [1]. Acquired results show us that, deep neural networks are more successful than linear classifiers commonly used in the neuroscience literature for brain decoding task. To obtain better results with deep neural networks for brain decoding problem, architectural choices are very important. According to our experiences two important problems must be solved to obtain these results. First one is using fMRI data on deep neural networks make them very susceptible to vanishing-gradient. To solve this problem, sparse representations must be acquired using ReLU-type activation functions, denoising operations and first-order per-dimension optimization methods. Second one is the natural differences between fMRI signal intensity values obtained from different subjects. To normalize these intensity values batch normalization method can be used. Brain-decoding experiments also show us the importance of connectivity in the brain. As it can be seen from the Chapter 4, utilizing the connections between each ROI provides the best brain decoding performance. There exist a $\sim 40\%$ performance gap between the brain-decoding with and without connectivity information.

For the second task, learned representations are used to improve the generalization performance of object recognition data set. Pre-trained weights are used to obtain hierarchical representations from object recognition data set. Then, these representations are used as a feature vector to train linear SVM classifier. Classification is implemented by linear SVM to be consistent with the neuroscience literature. Obtained results show us that, projecting the raw feature vectors to hierarchical feature vectors using deep neural networks improve the generalization performance of brain decoding significantly. To understand whether linear projection is ample to obtain these results or not, PCA and Group-ICA are also used. Performance gap between results obtained by deep neural networks and factor models is also significant.

After obtaining the brain decoding performance, we also visualize the most important relations between ROIs for different cognitive states. These relations are obtained by implementing saliency analysis over the trained deep neural network. Although these linear relations are not examined decently, obtaining probabilistic maps for each different cognitive states provides connectivity information about the brain.

5.2 Future Work

We believe that our approach provides a biologically consistent framework for multi-subject brain decoding and transfer learning on fMRI data. Yet, there are possible extensions exist to improve the performance of our framework.

As a first step, eliminating the unrelated voxels before obtaining the average ROI intensity values can enhance the performance. Although averaging the voxel intensity values of each ROI decrease the effect of these voxels on the overall performance, especially small ROIs are affected from them considerably.

Clustering voxels prior to averaging voxel intensity values can also increase the brain decoding performance. Although AAL indices provide a biological way to cluster voxels, clustering functionally similar voxels can provide more representative ROIs compared to AAL ROIs.

Finally, we believe that extracting connectivity information from voxel intensity values can automatically increase the performance of our framework. Although providing both Pearson correlation and mesh arc weights increase the classification performance, we believe that automatic extraction approaches can provide more abstract representations. Recurrent neural networks can be utilized to extract connectivity information from voxel intensity values.

REFERENCES

[1] Sotetsu Koyamada, Yumi Shikauchi, Ken Nakae, Masanori Koyama, and Shin Ishii. Deep learning of fmri big data: a novel approach to subject-transfer decoding. *arXiv preprint arXiv:1502.00093*, 2015.

[2] Seiji Ogawa, Tso-Ming Lee, Alan R Kay, and David W Tank. Brain magnetic resonance imaging with contrast dependent on blood oxygenation. *Proceedings of the National Academy of Sciences*, 87(24):9868–9872, 1990.

[3] Francisco Pereira, Tom Mitchell, and Matthew Botvinick. Machine learning classifiers and fmri: a tutorial overview. *Neuroimage*, 45(1):S199–S209, 2009.

[4] Federico De Martino, Giancarlo Valente, Noël Staeren, John Ashburner, Rainer Goebel, and Elia Formisano. Combining multivariate voxel selection and support vector machines for mapping and classification of fmri spatial patterns. *Neuroimage*, 43(1):44–58, 2008.

[5] Jonas Richiardi, Hamdi Eryilmaz, Sophie Schwartz, Patrik Vuilleumier, and Dimitri Van De Ville. Decoding brain states from fmri connectivity graphs. *Neuroimage*, 56(2):616–626, 2011.

[6] Itir Onal, Mete Ozay, Orhan Firat, Ilke Oztekin, and FT Vural. An information theoretic approach to classify cognitive states using fmri. In *Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on*, pages 1–6. IEEE, 2013.

[7] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

[8] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[9] LJP Van der Maaten. An introduction to dimensionality reduction using matlab. 2007.

[10] Dietmar Cordes and Rajesh R Nandy. Estimation of the intrinsic dimensionality of fmri data. *Neuroimage*, 29(1):145–154, 2006.

[11] Gagan S Sidhu, Nasimeh Asgarian, Russell Greiner, and Matthew RG Brown. Kernel principal component analysis for dimensionality reduction in fmri-based diagnosis of adhd. *Frontiers in systems neuroscience*, 6(74):1–16, 2012.

[12] Peter Mannfolk, Ronnie Wirestam, Markus Nilsson, Freddy Ståhlberg, and Johan Olsrud. Dimensionality reduction of fmri time series data using locally linear embedding. *Magnetic Resonance Materials in Physics, Biology and Medicine*, 23(5-6):327–338, 2010.

[13] Burak Velioglu, Emre Aksan, Itir Onal, Orhan Firat, Mete Ozay, and Fatos T Yarman Vural. Functional networks of anatomic brain regions. In *Cognitive Informatics & Cognitive Computing (ICCI* CC), 2014 IEEE 13th International Conference on*, pages 53–60. IEEE, 2014.

[14] Orhan Firat, Itir Onal, Emre Aksan, Burak Velioglu, Ilke Oztekin, and Fatos T Yarman Vural. Large scale functional connectivity for brain decoding.

[15] Gopikrishna Deshpande, Stephan LaConte, George Andrew James, Scott Peltier, and Xiaoping Hu. Multivariate granger causality analysis of fmri data. *Human brain mapping*, 30(4):1361–1373, 2009.

[16] Karl J Friston. Bayesian estimation of dynamical systems: an application to fmri. *NeuroImage*, 16(2):513–530, 2002.

[17] Junghoe Kim, Vince D Calhoun, Eunsoo Shim, and Jong-Hwan Lee. Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: Evidence from whole-brain resting-state functional connectivity patterns of schizophrenia. *NeuroImage*, 124:127–146, 2016.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[19] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[20] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[21] Matthew M Botvinick. Multilevel structure in behaviour and in the brain: a model of fuster's hierarchy. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 362(1485):1615–1626, 2007.

[22] David C Van Essen, Stephen M Smith, Deanna M Barch, Timothy EJ Behrens, Essa Yacoub, Kamil Ugurbil, WU-Minn HCP Consortium, et al. The wu-minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013.

[23] Orhan Firat, Like Oztekin, and Fatos T Yarman Vural. Deep learning for brain decoding. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 2784–2788. IEEE, 2014.

[24] Sergey M Plis, Devon R Hjelm, Ruslan Salakhutdinov, and Vince D Calhoun. Deep learning for neuroimaging: a validation study. *arXiv preprint arXiv:1312.5847*, 2013.

[25] Orhan Firat, Emre Aksan, Ilke Oztekin, and Fatos T Yarman Vural. Learning deep temporal representations for fmri brain decoding. In *Machine Learning Meets Medical Imaging*, pages 25–34. Springer, 2015.

[26] Paul C Lauterbur. Image formation by induced local interactions: examples employing nuclear magnetic resonance.

[27] Seiji Ogawa, David W Tank, Ravi Menon, Jutta M Ellermann, Seong G Kim, Helmut Merkle, and Kamil Ugurbil. Intrinsic signal changes accompanying sensory stimulation: functional brain mapping with magnetic resonance imaging. *Proceedings of the National Academy of Sciences*, 89(13):5951–5955, 1992.

[28] Xiaoping Hu and Essa Yacoub. The story of the initial dip in fmri. *Neuroimage*, 62(2):1103–1108, 2012.

[29] Martin A Lindquist et al. The statistical analysis of fmri data. *Statistical Science*, 23(4):439–464, 2008.

[30] Geoffrey M Boynton, Stephen A Engel, Gary H Glover, and David J Heeger. Linear systems analysis of functional magnetic resonance imaging in human v1. *The journal of neuroscience*, 16(13):4207–4221, 1996.

[31] Edson Amaro and Gareth J Barker. Study design in fmri: basic principles. *Brain and cognition*, 60(3):220–232, 2006.

[32] JW Belliveau, DN Kennedy, RC McKinstry, BR Buchbinder, RM Weisskoff, MS Cohen, JM Vevea, TJ Brady, and BR Rosen. Functional mapping of the human visual cortex by magnetic resonance imaging. *Science*, 254(5032):716–719, 1991.

[33] Tom M Mitchell, Rebecca Hutchinson, Marcel Adam Just, Radu S Niculescu, Francisco Pereira, and Xuerui Wang. Classifying instantaneous cognitive states from fmri data. In *American medical informatics association annual symposium*, 2003.

[34] Douglas M Hawkins. The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12, 2004.

[35] Jerome Friedman Trevor Hastie, Robert Tibshirani. The elements of statistical learning: Data mining, inference, and prediction. 2009.

[36] Gordon P Hughes. On the mean accuracy of statistical pattern recognizers. *Information Theory, IEEE Transactions on*, 14(1):55–63, 1968.

[37] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10:66–71, 2009.

[38] Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[39] John W Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, (5):401–409, 1969.

[40] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[41] Steven Lemm, Benjamin Blankertz, Thorsten Dickhaus, and Klaus Robert Muller. Introduction to machine learning for brain imaging. *NeuroImage*, 56(2):387–399, 2011.

[42] Joset A Etzel, Jeffrey M Zacks, and Todd S Braver. Searchlight analysis: promise, pitfalls, and potential. *Neuroimage*, 78:261–269, 2013.

[43] Martijn P Van Den Heuvel and Hilleke E Hulshoff Pol. Exploring the brain network: a review on resting-state fmri functional connectivity. *European Neuropsychopharmacology*, 20(8):519–534, 2010.

[44] Federica Agosta, Michela Pievani, Cristina Geroldi, Massimiliano Copetti, Giovanni B Frisoni, and Massimo Filippi. Resting state fmri in alzheimer's disease: beyond the default mode network. *Neurobiology of aging*, 33(8):1564–1578, 2012.

[45] Mete Ozay, Ilke Öztekin, Uygar Öztekin, and Fatos T Yarman Vural. Mesh learning for classifying cognitive processes. *arXiv preprint arXiv:1205.2382*, 2012.

[46] Maogeng Xia, Sutao Song, Li Yao, and Zhiying Long. An empirical comparison of different lda methods in fmri-based brain states decoding. *Bio-Medical Materials and Engineering*, 26(s1):1185–1192, 2015.

[47] Vladimir N Vapnik. Statistical learning theory. 1998.

[48] Michael J Hawrylycz, Ed S Lein, Angela L Guillozet-Bongaarts, Elaine H Shen, Lydia Ng, Jeremy A Miller, Louie N van de Lagemaat, Kimberly A Smith, Amanda Ebbert, Zackery L Riley, et al. An anatomically comprehensive atlas of the adult human brain transcriptome. *Nature*, 489(7416):391–399, 2012.

[49] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, DTIC Document, 1961.

[50] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[51] Peter Dayan and Laurence F Abbott. *Theoretical neuroscience*, volume 10. Cambridge, MA: MIT Press, 2001.

[52] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[53] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[54] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[55] Dong-Chen He and Li Wang. Texture unit, texture spectrum, and texture analysis. *IEEE transactions on Geoscience and Remote Sensing*, 28(4):509–512, 1990.

[56] John G Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2(7):1160–1169, 1985.

[57] Lawrence Rabiner and Biing-Hwang Juang. Fundamentals of speech recognition. 1993.

[58] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[59] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[60] Daphne Koller and Nir Friedman. Probabilistic graphical models. 2009.

[61] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[62] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[63] Peter Lennie. The cost of cortical computation. *Current biology*, 13(6):493–497, 2003.

[64] David Attwell and Simon B Laughlin. An energy budget for signaling in the grey matter of the brain. *Journal of Cerebral Blood Flow & Metabolism*, 21(10):1133–1145, 2001.

[65] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[66] Ian Goodfellow Yoshua Bengio and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016.

[67] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[68] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[69] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[70] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[71] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[72] Nathalie Tzourio-Mazoyer, Brigitte Landeau, Dimitri Papathanassiou, Fabrice Crivello, Olivier Etard, Nicolas Delcroix, Bernard Mazoyer, and Marc Joliot. Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage*, 15(1):273–289, 2002.

[73] Itir Onal, Mete Ozay, and Fatos T Yarman Vural. Modeling voxel connectivity for brain decoding. In *Pattern Recognition in NeuroImaging (PRNI), 2015 International Workshop on*, pages 5–8. IEEE, 2015.

[74] Joerg F Hipp and Markus Siegel. Bold fmri correlation reflects frequency-specific neuronal correlation. *Current Biology*, 25(10):1368–1374, 2015.

[75] Simon B Eickhoff, Klaas E Stephan, Hartmut Mohlberg, Christian Grefkes, Gereon R Fink, Katrin Amunts, and Karl Zilles. A new spm toolbox for combining probabilistic cytoarchitectonic maps and functional imaging data. *Neuroimage*, 25(4):1325–1335, 2005.

[76] Deanna M Barch, Gregory C Burgess, Michael P Harms, Steven E Petersen, Bradley L Schlaggar, Maurizio Corbetta, Matthew F Glasser, Sandra Curtiss, Sachin Dixit, Cindy Feldt, et al. Function in the human connectome: task-fmri and individual differences in behavior. *Neuroimage*, 80:169–189, 2013.

[77] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.

[78] Danilo Bzdok, Michael Eickenberg, Olivier Grisel, Bertrand Thirion, and Gaël Varoquaux. Semi-supervised factored logistic regression for high-dimensional neuroimaging data. In *Advances in neural information processing systems*, pages 3348–3356, 2015.

[79] Vince D Calhoun, Jingyu Liu, and Tülay Adalı. A review of group ica for fmri data and ica for joint inference of imaging, genetic, and erp data. *Neuroimage*, 45(1):S163–S172, 2009.