MACHINE LEARNING FOR THE DIAGNOSIS OF AUTISM SPECTRUM DISORDER

A Thesis Submitted to the

College of Graduate and Postdoctoral Studies

In Partial Fulfillment of the Requirements

For the Degree of Master of Science

In the Division of Biomedical Engineering

University of Saskatchewan

Saskatoon SK, Canada

By

SAKIB MOSTAFA

PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis/dissertation work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other uses of materials in this thesis in whole or part should be addressed to:

Head of the Division of Biomedical Engineering
University of Saskatchewan
Saskatoon, Saskatchewan
S7N 5A9
Canada

OR

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan
S7N 5C9
Canada

ABSTRACT

Autism Spectrum Disorder (ASD) is a neurological disorder. It refers to a wide range of behavioral and social abnormality and causes problems with social skills, repetitive behaviors, speech, and nonverbal communication. Even though there is no exact cure to ASD, an early diagnosis can help the patient take precautionary steps. Diagnosis of ASD has been of great interest recently, as researchers are yet to find a specific biomarker to detect the disease successfully. For the diagnosis of ASD, subjects need to go through behavioral observation and interview, which are not accurate sometimes. Also, there is a lack of dissimilarity between neuroimages of ASD subjects and healthy control (HC) subjects which make the use of neuroimages difficult for the diagnosis. So, machine learning-based approaches to diagnose ASD are becoming popular day by day. In the machine learning-based approach, features are extracted either from the functional MRI images or the structural MRI images to build the models.

In this study at first, I created brain networks from the resting-state functional MRI (rs-fMRI) images, by using the 264 regions based parcellation scheme. These 264 regions capture the functional activity of the brain more accurately compared to regions defined in other parcellation schemes. Next, I extracted spectrum as a raw feature and combined it with other network based topological centralities: assortativity, clustering coefficient, the average degree. By applying a feature selection algorithm on the extracted features, I reduced the dimension of the features to cope up with overfitting. Then I used the selected features in support vector machine (SVM), K-nearest neighbor (KNN), linear discriminant analysis (LDA), and logistic regression (LR) for the diagnosis of ASD. Using the proposed method on Autism Brain Imaging Data Exchange (ABIDE) I achieved the classification accuracy of 78.4% for LDA, 77.0% for LR, 73.5% for SVM, and 73.8% for KNN.

Next, I built a deep neural network model for the classification and feature selection using the autoencoder. In this approach, I used the previously defined features to build the DNN classifier. The DNN classifier is pre-trained using the autoencoder. Due to the pre-training, there has been a significant increase in the performance of the DNN classifier. I also proposed an autoencoder based feature selector. The latent space representation of the autoencoder is used to create a discriminate and compressed representation of the features. To make a more discriminate representation, the

autoencoder is pre-trained with the DNN classifier. The classification accuracy of the DNN classifier and the autoencoder based feature selector is 79.2% and 74.6%, respectively.

Finally, I studied the structural MRI images and proposed a convolutional autoencoder (CAE) based classification model. The T1-weighted MRI images without any pre-processing are used in this study. As the effect of age is very important when studying the structural images for the diagnosis of ASD, I used the ABIDE 1 dataset, which covers subjects with a wide range of ages. Using the proposed CAE based diagnosis method, I achieved a classification accuracy of 96.6%, which is better than any other study for the diagnosis of ASD using the ABIDE 1 dataset.

The results of this thesis demonstrate that the spectrum of the brain networks is an essential feature for the diagnosis of ASD and rather than extracting features from the structural MRI image a more efficient way is to use the images directly into deep learning models. The proposed studies in this thesis can help to build an early diagnosis model for ASD.

KEYWORDS: Functional Magnetic Resonance Imaging, Brain Network, Feature, Machine Learning, Deep Neural Network, Autoencoder

ACKNOWLEDGEMENTS

I would like to convey my most sincere gratitude to my supervisor, Professor Fang-Xiang Wu, for all his patience and all those inspiring discussions. Without help from Dr. Wu, I will not be able to finish my research and this thesis. I would like to thank him for always leading the way and supporting me with all my decisions.

I would like to thank other committee members: Professor W. J. Chris Zhang and Professor Khan A. Wahid, for their suggestions and comments on my research and thesis.

My appreciation also goes to the group members, who are Lingkai Tang, Ali Jamali Beyrami, Lin Wu, Ping Luo, Fei Wang, Yulian Ding, Rayyan Khan, etc.

I should also thank all my friends here who have made my life much easier and less lonely. Thank you all for your support and advice.

My special gratitude belongs to my parents and my little brother, who have made many sacrifices in their lives to make me happy. I can never repay what they have done for me.

I would like to thank my lovely wife. I am grateful to her for always being supportive and listening to everything I had to say. I am thankful to her for coming with me to the university late in the night when I needed support. Lastly, I would like to thank God for everything I have.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABIDE	Autism Brain Imaging Data Exchange
ACC	Accuracy
AD	Alzheimer’s Disease
AFNI	Analysis Of Functional Neuroimages
ASD	Autism Spectrum Disorder
AUC	Area Under ROC Curve
BOLD	Blood Oxygenation Level-Dependent
CAE	Convolutional Autoencoder
CNN	Convolutional Neural Network
CT	Computerized Tomography
DNN	Deep Neural Network
fMRI	Functional MRI
FSL	FMRIB's Software Library
GAN	Generative Adversial Network
HC	Healthy Control
KNN	K-Nearest Neighbor
LDA	Linear Discriminant Analysis
LR	Logistic Regression
MEG	Magnetoencephalography
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NN	Neural Network
PCC	Pearson Correlation Coefficient
PET	Positron Emission Tomography
PSNR	Peak Signal to Noise Ratio
QEEG	Quantitative Electroencephalography
RF	Radiofrequency
ROC	Receiver Operating Characteristic

ROI	Region of Interest
rs-fMRI	Resting State Functional MRI
SSIM	Structural Similarity Index
STDV	Standard Deviation
SVM	Support Vector Machine

CHAPTER 1
INTRODUCTION

1.1 Background

1.1.1 Autism Spectrum Disorder

Autism spectrum disorder (ASD) is a neurobehavioral disorder. It refers to a broad range of conditions characterized by impaired social skills, co-occurring behaviors, reduced speech, and nonverbal communication, depression, anxiety, attention deficit, and a limited extent of interests and activities that are carried out differently. There isn't any particular type of ASD but many because of the different combinations of genetic and environmental impact. The term "spectrum" in ASD is due to the variation of the conditions of the patients, as it affects different individuals differently.

The subjects suffering from ASD are 2.5 times more likely to have a premature death than the healthy controls (HC), i.e., people who don't have ASD [1]. According to [2], the mean age of death for the ASD subjects is 36.2 years (±20.9 years) compared to the 72.0 years (±19.2 years) for HC subjects. 1 in 66 children in Canada between the ages 5 to 17 years has ASD. Males are four times more likely to suffer from ASD than females [3]. In the USA, one-third of the subjects suffering from ASD remain nonverbal, and around 50,000 teens grow up and lose their school-based autism services, each year [4]. So, ASD is more common in today's society than we think. In the past, for a child suffering from ASD, the parents would get little to no help at all. But it has changed recently. Now, there are organizations that fund the education of the ASD patients, schools that are designed especially for ASD patients, social support groups, and clinics for better understanding of the disease. Even though there is no cure for ASD, a timely and accurate diagnosis can help the family take preliminary and effective steps to ensure the normal life of the patient.

1.1.2 Diagnosis of Autism Spectrum Disorder

Patients start to show symptoms of ASD during the first three years of life. But sometimes they grow normally and then start showing symptoms at the age between 18 to 36 months. Despite the

extensive research into the diagnosis of ASD, it has been a difficult task to accomplish. Apart from monitoring the behavior and development of the patient, there are no other signs for effective diagnosis of ASD. The traditional diagnosis process includes Autism Diagnostic Observation Schedule [5] and Autism Diagnostic Interview [6]. But, these diagnosis processes are time-consuming and can be at fault sometimes, as there are no specific behaviors that can be described as ASD. So, it is necessary to invent ways that can diagnose ASD more accurately and more efficiently without relying on the behavioral pattern.

Recently, using machine learning algorithms for the diagnosis of ASD has become popular. It is necessary to define features to train a machine learning classifier. The features need to be discriminative, for the classifiers to work efficiently. There are different ways to determine the features. The structural property of the brain can be used as features where the property of different regions are studied and if there is any change due to a particular disease the change is used as a feature [7], [8], [9]. The phenotypic information of the patients [10], [11] and the behavioral attributes [12] can also be incorporated into the study and used as features. However, the features used in the mentioned studies to diagnose ASD aren't discriminative and competent enough as they lack having a satisfactory classification result over a larger dataset. So, a better approach for the diagnosis of ASD is to use machine learning classifiers where the features are extracted from the neuroimages [13], [14], [15].

1.1.3 Neuroimage

There are different ways for the acquisition of neuroimages, such as magnetic resonance imaging (MRI), computerized tomography (CT) and positron emission tomography (PET). However, MRI images are preferable for the study of the brain because it is better at imaging the soft tissues of the body. MRI images can differentiate between the white and grey matter better than other neuroimaging techniques. MRI images don't require the use of x-ray or other radiation for image acquisition, which makes it safer compared to other image acquisition techniques. Also, it provides more accurate information about the structural and functional activity and composition of the brain [16].

In MRI, a strong magnetic field is applied through the patient's body, which forces the protons in the body to align with the magnetic field. Now, if a radiofrequency (RF) current is applied, the protons start to spin out of their equilibrium. When the current is turned off, the protons go back

to their actual position, which takes time and causes them to release energy. Based on the emission of energy and the time, the sensors in the MRI scanner can create images [17].

The two basic types of MRI images based on the RF current are T1-weighted MRI images and T2-weighted MRI images. The acquisition of the T1-weighted images depend on the time it takes for the spins to align to its equilibrium state after the current is removed. In the T1-weighted images, the fatty tissues, melanin, and protein-rich fluids are brighter. T2-weighted imaging reflects the time it takes for the spins to decay to its equilibrium state in the transverse plane. Tissues containing water are brighter in the T2-weighted images [18]. Both T1-weighted and T2-weighted images can be used to build structural networks. The structural network is a representation of the structural integrity of the brain [19].

Apart from the T1-weighted and T2-weighted images, there is another MRI imaging technique called functional MRI (fMRI). The fMRI looks into the functional activity of the brain by using the temporal similarity of the blood oxygenation level-dependent (BOLD) signals from the different regions of the brain. If the patient is resting while the images are collected, then they are called resting-state functional MRI (rs-fMRI). A functional network can be built from the fMRI images as the similarity between the BOLD signals in each region illustrates that the regions are communicating with one another [20]. As a result, the rs-fMRI provides information about the neural activity of the brain [21], [22].

1.1.4 Related Studies

Due to the lack of a definite biomarker for the diagnosis of ASD, researchers have been trying to locate one. A biomarker is a medical sign, that can help identify a disease accurately [23]. A medical sign can be a medical symptom, biological fluid, and so on. In the search for biomarkers, the researchers looked into the abnormality of the chromosomes [24], studied magnetoencephalography (MEG) to define MEG related biomarkers [25], used quantitative electroencephalography (QEEG) to understand the brain function and connectivity abnormality [26], investigated the regional and global growth of brain in early childhood [27], studied the hyper-activation and hypo-activation of cortical regions [28], [29]. However, a more popular approach is the study of neuroimages for the detection of brain disease. The neuroimages are studied not only for the diagnosis of ASD but also for other brain diseases such as Alzheimer's disease [30], [31] and schizophrenia [32], [33], [34].

The combination of machine learning algorithms and MRI images has allowed performing a better diagnosis of ASD. A study has found a lack of a pattern of brain activation in the MRI images of the ASD subjects, compared to the HC subjects [35]. Another study has achieved a classification accuracy of 76.7% by dividing the brain into 7266 regions of interest (ROIs), the pairwise correlation between each ROI is calculated in [36] to fit a general linear model for each type of subject (ASD and HC) to diagnose the ASD accurately.

A popular approach for studying neuroimages is based on graph theory or network theory [37], [38]. In the network theory, a network is a combination of nodes and edges, where the nodes are connected with each other through the edges. In this approach, a network is created from the MRI images by dividing the brain into different ROIs. A connectivity matrix can be constructed from the brain network, where every ROI is the node, and the Pearson correlation coefficient (PCC) between any two ROIs is the edge. There are different ways to use the connectivity matrix for the classification. One approach is to use every correlation coefficient as a feature for the machine learning classifiers [13]. Another method is to measure the strength between different ROIs to find the ROI responsible for ASD [39]. In [36], a brain network is created by defining every voxel of gray matter as ROIs and then analyzing the networks. The network-based studies [40] have found that functional integration and segregation are altered in ASD. The brain becomes more integrated and segregated for the HC subjects compared to ASD subjects. Also, lower intrinsic functional connectivity [41] and lower default mode network connectivity [42] is reported for the ASD patients. During the development of the HC subjects, the integrity of the white matter also increases [43]. Changes in brain networks due to the development over different ages are studied in [44].

In the study [45], features are extracted from the brain networks for the classification of ASD. The features include different centralities, which are measured directly from the brain networks. The brain networks are created from the rs-fMRI and T1-weighted images. So, both structural and functional images are used. Based on the features, machine learning classifiers are trained for the diagnosis of ASD. The local and global graph-theoretical matrices are measured in [19]. They have also studied the relation and difference between the structural and functional properties of the brain between the ASD subjects and HC subjects. Machine learning classifiers are incorporated with functional brain network analysis in [46] to find biomarkers for the diagnosis of ASD.

Apart from the traditional machine learning algorithms, deep neural network (DNN) is also used to develop better methods for the classification of brain diseases. In [47], seven brain states have been classified based on the neuroimages using the DNN. The prospect of DNN for the disease classification is thoroughly studied in [48], where schizophrenia is detected from the controlled subjects using T1-weighted images. A more direct approach is implemented in [49]. The authors have created a connectivity matrix from the brain network, which is then used in a DNN model for classification of ASD. Structural brain networks are studied in [50] using a fully-connected autoencoder. The brain networks are also used in the convolutional neural network (CNN). A very basic CNN containing four convolutional layers and one fully-connected layer is used in [51]. The study acquired a classification accuracy of 73.0% using CNN. CNN can also be applied for the analysis of the data over the graphical domains [52]. The fact is further proved in [53], where the connectivity matrix is treated as an image to train a CNN model. In [54], the brain is divided into 90 ROIs, and the connectivity matrix is created based on these ROIs. The connectivity matrix is then applied to a modified CNN model, which works by assigning weights to the edges of the network. After reducing the dimensionality of the features, a fully connected layer is applied for the final classification. Using a combination of recurrent neural network and long-short term memory on the rs-fMRI, the authors classified ASD from the HC [55]. A bootstrapped version of the graph-CNN is implemented in [56] for the classification of ASD. They proposed graph-based predictive modeling, which relies on the ensemble of the population graphs. In [57] at first a graph-CNN is used to extract features from the images, then the features are combined with phenotypic information in the final layer to build a classifier to diagnose ASD. So, both traditional and advanced machine learning algorithms can be used to develop methods to diagnose ASD more efficiently.

1.2 Problem Formulation

There have been a number of researches for the detection of a biomarker for the diagnosis of ASD. However, the accuracy of the researches isn't satisfactory due to the use of inefficient features. A group of studies that achieved good accuracy using supervised ML and brain imaging data used a relatively small number of participants. A reliable result of ML studies is obtained using fewer than 100 subjects [58]. There is a significant drop in the classification accuracy for larger sample size and also for the data collected from different sites [36].

In [15], a large dataset containing 871 subjects from the ABIDE database are used. The data is collected from 17 different sites. Applying the DNN, they achieved an accuracy of 71.1%. There have been more studies using the same 871 subjects [57], [59], where they obtained a classification accuracy of 68.0% and 69.5%, respectively. Using an almost similar number of subjects, an accuracy of 70.9% is acquired in [56]. Applying DNN on 1035 subjects from the ABIDE dataset [49] had a classification accuracy of 70%. Two problems can be identified from these studies. First of all, the inconsistency of data. Because of the inconsistency, it is very hard to make the ASD based studies comparable with each other. The second problem is, even though researchers have obtained a good accuracy of around 70.0%, but it still isn't satisfactory.

In brain network-based approaches, the complexity of a network depends on how the nodes and edges are defined in the MRI images [60], [61]. If the nodes and edges aren't defined correctly, the network will become too complex to study [62]. So, the ROIs should be able to capture as much as information possible, and the information needs to be discriminative.

To train a machine learning classifier, it is necessary to define features. The features have to be independent and discriminative. The features that are used in previous studies aren't discriminative enough for the classification of ASD, as they couldn't achieve a satisfactory classification result for a large dataset. So, it is necessary to define some new features that can classify ASD from HC with good classification accuracy.

As the size of the brain networks is large, the dimension of the feature vector is also large. However, not all the features in the feature vector are necessary. Unnecessary features make the classification model complicated and inefficient. So, a feature selection algorithm is required to reduce the dimension of the selected features and also find the discriminative features that will help the classification. Feature selection also solves the problem of overfitting by reducing the dimension of the features and removing the noise in the features, which tend to overfit the model.

ASD covers a wide range of behavioral abnormality. So, the findings of different studies to locate structural abnormality for the diagnosis are also heterogeneous, and the findings differ for different studies [63]. However, recent studies of structural MRI images for the diagnosis of ASD have shown that there are some abnormalities in the grey matter for both children and adults who have ASD [64]. Also, for children with ASD, there is overgrowth in the frontal cortex [65], and there is variation in the cortical thickness, which is also dependent on the age [66]. Decreased grey matter

volume in the cerebellum [67], decreased mid-sagittal area of the brainstem [68], dissimilarity in the cortical volume, cortical thickness, the surface area of the brain [69] is also found in the ASD subjects compared to the HC subjects.

In most of the structural brain imaging studies for the classification of ASD, a particular age group is used for the analysis, and the effect of age is ignored [63]. However, age plays a vital role in the development of ASD. In [67] it is reported that there is an overgrowth of the brain in early age followed by abnormal slowed growth. The overgrowth of the brain for children under the age of 2 years is reported in [70]. A longitudinal study in [71] shows a brain overgrowth in early age for ASD subjects and then convergence with average volume around ten years of age.

The advantage of using CNN for the analysis of the images is that there is no need to define hand-crafted features. The CNN model searches for features in the images. When using the hand-crafted features of the structural MRI images (grey matter and white matter ratio, cortical thickness), the focus in only on a particular ROI of the image. The relation of the ROIs with each other is ignored in this case. In the CNN model relationships between the ROIs is also considered. However, there hasn't been any study related to the diagnosis of ASD, where the structural images are used in the CNN model for classification.

1.3 Objectives

The main focus of this research is to develop better, consistent, and reliable machine learning based diagnosis processes for ASD using structural and functional MRI images. The objectives that are to be achieved for this are presented in the following.

Firstly, brain networks are created from the rs-fMRI images of the ABIDE 1 database by focusing on the ROIs that creates a better representation of the functional activity of the brain. The spectrum of the Laplacian matrix of the brain network is proposed as a feature to enhance the performance of the machine learning classifiers.

Secondly, the effectiveness and robustness of the proposed feature (spectrum of the Laplacian matrix of the brain network) are proved by experimenting with different traditional machine learning algorithms. Also, deep learning based models are studied to improve the classification accuracy.

Thirdly, a convolutional autoencoder (CAE) based ASD diagnosis method is proposed where the raw images are used for the diagnosis. Finally, the findings of the proposed studies in this thesis are compared with state-of-the-art methods.

1.4 Thesis Organization

In Chapter 1, I have presented the background of this thesis, along with a discussion of the related works. In this chapter, I have also formulated the problem that I am trying to solve and mentioned the objectives that I am trying to achieve. In Chapter 2, a brief introduction is given about the dataset that I have used in this thesis. Besides talking about the dataset, I have also described the pre-processing steps of the rs-fMRI images and the process of building brain networks from images in this chapter. Chapter 3 starts with the description of the features I have used in this study and also the process of extracting the features from brain networks. Then I have discussed the feature selection process. The performance of different machine learning classifiers is discussed at the end of this chapter. In Chapter 4, I have proposed the use of an autoencoder based classifier and feature selector for the diagnosis of ASD. After introducing the autoencoder, I have discussed in brief the architecture of the proposed models. I have shown that there is an increase in the performance due to the pre-training of the DNN classifier and feature selector. Chapter 5 is about the study of the structural MRI images for the diagnosis of ASD. In this chapter, I have shown that a classification model can be built to classify ASD subjects from the HC subjects using the CAE. Finally, in Chapter 6, concluding remarks about this study are presented along with the indication of possible future directions.

CHAPTER 2
MRI IMAGE DATASET AND DATA PROCESSING

2.1 Introduction

This chapter provides a preliminary discussion about the dataset and preprocessing steps. In this chapter at first, I will introduce the database that I have used in my thesis study. I will also discuss the challenges with the database. Next, I will discuss different techniques of creating brain networks from the MRI images. I will also discuss the method that I have implemented in my study, where I will describe the preprocessing steps of the MRI images and then describe the process of creating brain networks. Finally, I will explain the different matrix representations of the brain networks and then conclude the chapter.

2.2 Autism Brain Imaging Data Exchange

To date, one of the most consistent dataset for the study of ASD is the Autism Brain Imaging Data Exchange (ABIDE, http://fcon_1000.projects.nitrc.org/indi/abide/) [72]. There have been a number of researches for the detection of ASD. However, the pace and clinical impact of the improvement still isn't remarkable. The major hurdle in the direction of a suitable diagnosis process is the complexity and diversity of ASD. To establish a diagnosis process, it is necessary to have a strategy that works on different types of ASD patients, covering different age, sex, and so on. If neuroimages are used, then the proposed approach should work for different scanners and scanning parameters. Thus, a large dataset is needed. It is tough to obtain this sort of data from a single source. So, with the ultimate goal of advancement in the diagnosis of ASD, the ABIDE dataset is created. In the ABIDE dataset, there are two large-scale collections of the MRI images named ABIDE 1 and ABIDE 2. Both the datasets are created by collecting data separately and independently from 24 different neuroimaging laboratories all over the world studying ASD.

In this thesis, I have worked on the ABIDE 1 [73] dataset for all the studies. It is the first initiative of the ABIDE. There are data from 17 different sites in this dataset. The data includes functional MRI images (rs-fMRI), structural MRI images (T1-weighted MRI), and also phenotypic information. There are data of a total of 1112 subjects in ABIDE 1. Among the 1112 subjects, 539 subjects have ASD and 573 subjects are HC. ABIDE 1 covers ages from 6.5 to 64 years, median 14.7 years. As the data are acquired independently and separately, the scanner and scanning

***Table 2-1**: Scanning parameters of different sites of ABIDE 1*

Sites	MRI Scanner	TR (ms)		TE (ms)		Flip Angle (Degree)		Voxel Size (mm)	
		S	**F**	**S**	**F**	**S**	**F**	**S**	**F**
CALTECH	SIEMENS	1590	2000	2.73	30	10	75	1.0×1.0×1.0	3.50×3.50×3.50
CMU	SIEMENS	1870	2000	2.48	30	8	73	1.0×1.0×1.0	3.00×3.00×3.00
KKI	PHILLIPS	8	2500	3.70	30	8	75	1.0×1.0×1.0	3.05×3.15×3.00
MAX_MUN	SIEMENS	1800	3000	3.06	30	9	80	1.0×1.0×1.0	3.00×3.00×4.00
NYU	SIEMENS	2530	2000	3.25	15	7	90	1.3×1.0×1.3	3.00×3.00×4.00
OLIN	SIEMENS	2500	1500	2.74	27	8	60	1.0×1.0×1.0	3.40×3.40×4.00
OHSU	SIEMENS	2300	2500	3.58	30	10	90	1.0×1.0×1.0	3.80×3.80×3.80
SDSU	GE	11.08	2000	4.3	30	45	90	1.0×1.0×1.0	3.40×3.40×3.40
SBL	PHILLIPS	9	2200	3.50	30	8	80	1.0×1.0×1.0	2.75×2.75×2.72
STANFORD	GE	8.4	2000	1.80	30	15	80	0.8×1.5×0.8	3.12×3.12×4.50
TRINITY	PHILLIPS	8.5	2000	3.80	28	8	90	1.0×1.0×1.0	3.00×3.00×3.50
UCLA	SIEMENS	2300	3000	2.84	28	9	90	1.0×1.0×1.2	3.00×3.00×4.00
LEUVEN	PHILLIPS	9.6	1656	4.6	33	8	90	0.9×0.9×1.2	3.59×3.59×4.00
UM	GE	250	2000	1.8	30	15	90	1.0×1.0×1.0	3.43×3.43×3.00
PITT	SIEMENS	2100	1500	3.93	25	7	70	1.1×1.1×1.1	3.10×3.10×4.00
USM	SIEMENS	2300	2000	2.91	28	9	90	1.0×1.0×1.2	3.40×3.40×3.00
YALE	SIEMENS	1230	2000	1.73	25	9	60	1.0×1.0×1.0	3.40×3.40×4.00

S: Structural, **F**: Functional, **CALTECH**: California Institute of Technology, **CMU**: Carnegie Mellon University, **KKI**: Kennedy Krieger Institute, **MAX_MUN**: Ludwig Maximilians University Munich, **NYU**: NYU Langone Medical Center, **OLIN**: Olin, Institute of Living at Hartford Hospital, **OHSU**: Oregon Health and Science University, **SDSU**: San Diego State University, **SBL**: Social Brain Lab BCN NIC UMC Groningen and Netherlands Institute for Neurosciences, **STANFORD**: Stanford University, **TRINITY**: Trinity Centre for Health Sciences, **UCLA**: University of California, Los Angeles, **LEUVEN**: University of Leuven, **UM**: University of Michigan, **PITT**: University of Pittsburgh School of Medicine, **USM**: University of Utah School of Medicine, **YALE**: Yale Child Study Center.

parameters are different. The details about the scanner and scanning parameters are provided in Table 2-1. From Table 2-1, it can be seen that along with the scanners, the different pulse sequences (TR and TE), flip angle and voxel size differed for different sites over the whole dataset.

Not only the scanner and scanning parameters but also the scanning process are different for different sites. The subjects of some sites participated in a mock scan session before the actual scan session. During the acquisition of MRI images, some sites allowed the participants to fall asleep, where some sites ensured the participants are awake. Also in some sites, the participants

***Table 2-2**: Phenotypic information of the subjects of this study*

Sites	Age (Years)	Count		Total
		ASD	HC	
CALTECH	17.0 - 56.2	5	10	15
CMU	19.0 – 40.0	6	5	11
KKI	8.0 - 12.8	12	21	33
MAX_MUN	7.0 – 58.0	19	27	46
NYU	6.5 - 39.1	74	98	172
OLIN	10.0 – 24.0	14	14	28
OHSU	8.0 - 15.2	12	13	25
SDSU	8.7 - 17.2	8	19	27
SBL	20.0 – 64.0	12	14	26
STANFORD	7.5 - 12.9	12	13	25
TRINITY	12.0 - 25.9	19	25	44
UCLA 1	8.4 - 17.9	37	27	64
UCLA 2	9.8 - 16.5	11	10	21
LEUVEN 1	18.0 – 32.0	14	14	28
LEUVEN 2	12.1 - 16.9	12	16	28
UM 1	8.2 - 19.2	34	52	86
UM 2	12.8 - 28.8	13	21	34
PITT	9.3 - 35.2	24	26	50
USM	8.8 - 50.2	43	24	67
YALE	7.0 - 17.8	22	19	41
Total		**403**	**468**	**871**

are asked to keep their eyes closed during the image acquisition. However, in some sites, the participants are asked to keep their eyes open and focus on a particular object. So, the ABIDE 1 covers different scanner, scanning parameters, and scanning process over different sites [74].

A major problem related to the study of ASD is the inconsistency of the dataset. The researchers don't use a consistent dataset to make their work comparable with others. Rather than using a diverse dataset, they prefer to work with subjects within a specific age range or sex. So, the datasets aren't large or generalized enough to validate their proposed methods. So, to make my thesis work consistent and comparable with other studies [15], [57], [59], I have worked with the same 871 subjects out of the 1112 subjects of the ABIDE 1 dataset. 403 subjects of the 871 subjects are diagnosed with ASD and the 468 subjects are the HC. In [59], it is mentioned that the subjects are selected after three experts examined the images of the subjects and excluded the rest of the subjects due to incomplete brain coverage, high movement peaks, ghosting, and other scanners.

Information about the age of the subjects and the number of subjects from different sites are recorded in Table 2-2. Analyzing Table 2-1 and Table 2-2, it can be said that ABIDE 1 is a dataset with variable scanners and scanning parameters covering a wide range of ages.

2.3 Brain Network Analysis

2.3.1 Resting State Functional MRI Preprocessing

The slices in the MRI images sometimes suffer from variation in the signal intensity, random noise spikes, and data glitches. There are different reasons for this discontinuity in individual slices. So some preprocessing of the MRI images is required before creating brain networks. There are various tools and software available for the analysis and preprocessing of the images. However, in my study, I have used the Analysis of functional neuroimages (AFNI) [75] and FMRIB's software library (FSL) [76] for the analysis of the MRI images. Both the software have a wide range of toolboxes for the study of neuroimages. To create brain networks from the neuroimages, I have only used the rs-fMRI images. However, for the preprocessing, I have used both T1-weighted MRI images and rs-fMRI images. The only purpose of the T1 weighted images is to register the rs-fMRI images to the standard space.

The first step of the preprocessing is to remove the spikes from the images due to the head movement of the subject. I have used **3dDespike** of AFNI to remove the spikes, as the motion correction algorithms aren't capable of eliminating these movements. The next step is called skull stripping, which removes the non-cerebral tissues like skull, eyes, scalp, etc. The skull stripping helps to identify the ROIs by segmenting only the brain tissue [77]. I have used **3dAutomask** and **3dSkullStrip** of AFNI to remove the skull tissues. Head motion is one of the most significant causes of error in the MRI images. The head is strapped with belts and heavy paddings, so it remains still during the image acquisition process. Still, there are motions in the captured images. So, after skull stripping, I motion-corrected the images using **MCFLIRT** of FSL [78]. The motions are corrected in the functional volumes on specific time series, and then the mean functional volume is calculated. Now for the MCFLIRT, the cost function is the normalized correlation ratio. To increase the resolution of the images sinc interpolation is used for resampling. Now, to make the images of different subjects to be comparable with each other, it is necessary to register the images to a standard space. The registration is required because the slices of the MRI images are acquired one by one. So, it is highly likely the MRI of two different subjects may not align with

each other. As a result, the comparison becomes very hard. I have used **FLIRT** of FSL for the image registration purpose. At first, for every patient I have registered the rs-fMRI images to the T1 weighted MRI images. Then I have registered the T1 weighted images to the MNI 152 standard space [78], [79]. The MNI 152 is the average of the nonlinearly registered 152 T1 weighted MRI images. MNI 152 exhibits the best resolution and detail to date. So, it is called to be a representative of the average brain of the population.

Spatial smoothing is a prevalent preprocessing step for the neuroimages. It helps reduce the resampling related artifacts after image registration [80]. So, I have applied a full width half maximum (FWHM) Gaussian kernel of 5mm for the spatial smoothing. Then I have rotated, translated, scaled, and skewed the images in X, Y, and Z directions of the MNI 152 space to align the volumes of each time series. To minimize the effect of body movement, I have also regressed out the six rigid body movement parameters, which are rotation and translation in X, Y, and Z direction. Along with the six rigid movements, the white matters (WM) and cerebrospinal fluids (CSF) are also regressed out of the images, because WM and CSF reflect non-neural fluctuation rather than neural activities. So at first, the **FLIRT** of FSL is used to segment the WM and CSF from the T1 weighted images. Then **3dmaskave** of AFNI is used to calculate the average time series. Finally, the regression is done by **3dConvolve** and **3dREMLfit** of AFNI. Heartbeat and respiration sometimes cause fluctuation in the images acquired. To get rid of the effects of heartbeat and respiration, a bandpass filter is used with a passband of 0.01Hz to 0.1Hz.

2.3.2 Defining Nodes and Edges

After processing the rs-fMRI images, the next step is to create a network representing the brain. A network is a collection of nodes and edges. There are different ways to define the nodes and edges. It is the most crucial step in creating a brain network from the neuroimages [60]. If the nodes and edges are not appropriately defined, the network might become too complicated for analysis [62].

Each voxel in the neuroimages can be defined as a node to create a network from the MRI images [13], [81], [82]. The connection between the voxels are the edges, where the connection is measured in different ways (e.g. PCC). This approach is called the voxel-based parcellation scheme. Voxels are individual array elements of a volume in the three-dimensional space. It is the smallest building block of the MRI images. However, in this approach, it is ignored that the human brain is a complex macroscopic network. Particular ROIs contribute more to the functionality of

the brain, compared to a small portion of the brain, such as voxel [83]. So, rather than using the voxel as a node, ROI based parcellation scheme is a better approach.

In ROI based parcellation scheme, the brain is divided into different ROIs. An ROI is a form of annotation, and in neuroimages, it is considered to be a reference point. In this approach, each ROI is regarded to be a node, and the connection between the ROIs are the edges. There are different ways to define ROIs. Defining the anatomical structures, such as sulci, gyri, cortex, amygdala [60], [84], [85], [86] as ROIs is one of the popular ways. However, an anatomical ROI is substantial in volume, and it might contain several sub-regions. Hence, the activity of individual regions cannot be captured. This problem can be solved if the ROIs represent small structures of the brain. So, researchers have defined different parts of the brain through coordinates. Along with the coordinates, an approximate volume is given. Thus the volumes around the coordinates together from the predefined atlas. Automated anatomical labeling (AAL), Desikan, DKT 40, Destrieux are some of the predefined atlases.

The mentioned predefined atlases focus on the structural property of the brain. So, if the brain goes through structural changes due to a disease, then the atlas should be able to detect the difference by analyzing the structural changes of the ROIs in the atlases. However, in [87], it is mentioned that the size of the ROIs in these atlases are very large. So, the ROIs might contain small functional ROIs within them. Creating a functional brain network using the structural ROIs, the network will become too complex to analyze. The complexity will arise because we will need a different method to define the small functional ROIs and integrate them with the bigger ones.

In my study, I have used the atlas defined in [87], where the brain is divided into 264 ROIs. These 264 ROIs are based on the functional activity of the brain. During the analysis of the rs-fMRI and task-based fMRI, 322 ROIs are identified to be active. The activity is measured by asking the subjects to perform specific tasks. Also for rs- fMRI, the images show an obvious transition of fMRI signal correlation, which forms boundaries that partition the cortex into ROIs. Combining these two methods and excluding the overlapped ROIs, in [87] finally, 264 ROIs are defined. These 264 ROIs are coordinate points in the MNI 152 standard space. Around the coordinate points, a sphere of 5mm radius is considered.

So, I have used the 264 ROIs based parcellation scheme in this study. These 264 ROIs are the nodes of my network. Now, the nodes need to be connected with each other through the edges.

Edges in a network can be either weighted or unweighted. In a weighted edge, a numerical value is assigned to the edges. Edge weights can be represented as a function $\omega = E \rightarrow R$ that assigns each edge $e \in E$ a weight $\omega(e)$ in a network with V number of nodes and E number of edges. The edge weight can be the travel time or distance between nodes, capacity, the strength of interaction, similarity, etc. An unweighted edge represent only a connection between nodes where $\omega(e) = 1$ for all $e \in E$. For the structural MRI images, the number of fibers between two ROIs is the weighted edge of the network [19]. However, the calculation is a bit different for the functional images. In a functional image, the Pearson Correlation Coefficient (PCC) of the time series measurement between two ROIs is the edge of the network [88]. The PCC is a measurement of the linear correlation between any two variables. The PCC, r_{xy} of any two time series is calculated as follows

$$r_{xy} = \frac{\sum_{b=1}^{s}(x_b - \bar{x})(y_b - \bar{y})}{\sqrt{\sum_{b=1}^{s}(x_b - \bar{x})^2}\sqrt{\sum_{b=1}^{s}(y_b - \bar{y})^2}} \qquad (2.1)$$

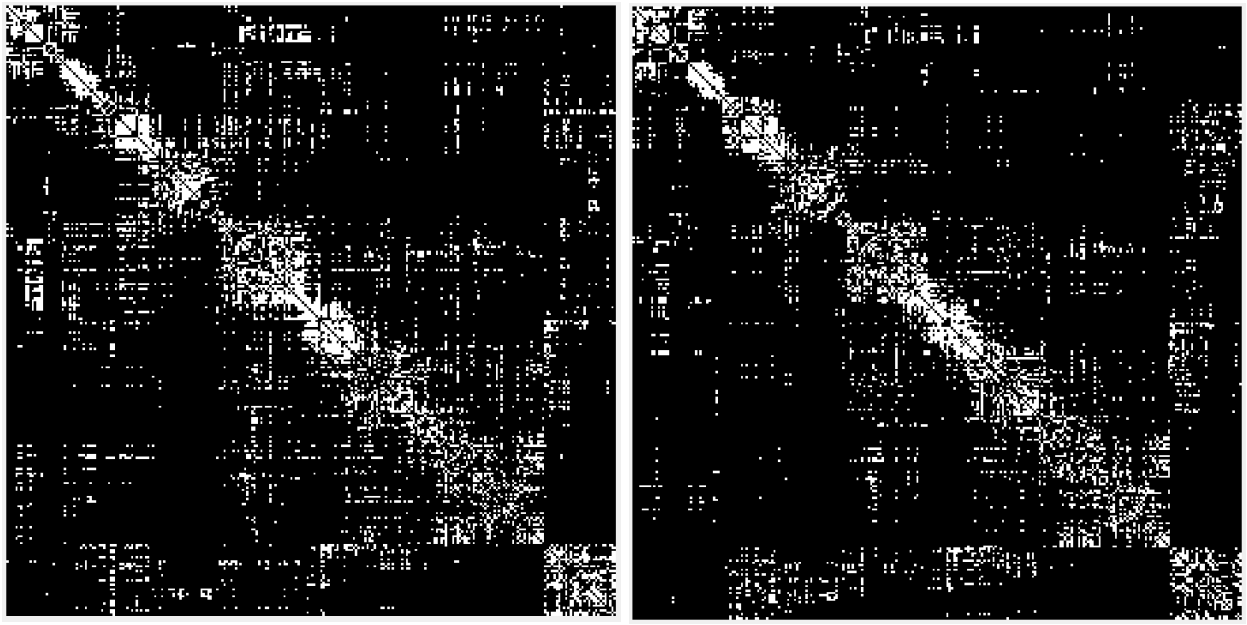
where s is the length of time series, x_b and y_b are the b-th component of x and y, respectively, $\overline{x}$ and $\overline{y}$ are the mean of x and y, respectively. The range of PCC is from +1 to -1. In this study, a positive PCC represents that, both the ROIs have a similar activity during the same time point, i.e. the stimulation between the regions are more or less similar at that particular time point. On the other hand, a negative PCC represents that the stimulation between the ROIs doesn't have similarity, i.e., the ROIs have a difference in the activation during the stimulation. So, a positive PCC represents the similarity of activation, and a negative PCC represents the dissimilarity of the activation.

2.3.3 Matrix Representation of Brain Networks

2.3.3.1 Thresholding the Connectivity Matrix

There are 264 nodes in the brain network defined in this study. The PCC of any two nodes is the edge in the network. Using this information, I have created a connectivity matrix of size 264×264 for every patient from the brain network. Each row and column represent one of the 264 ROIs (node of the network), and every element of the matrix is a PCC of the time series

measurement of the corresponding ROIs (edge of the network). A graphical representation of the connectivity matrix is shown in Figure 2-1.



(a) (b)

Figure 2-1*: A graphical representation of the connectivity matrix, (a): ASD, (b): HC*

To train the machine learning classifiers for the diagnosis of ASD, the next step is to define features from the connectivity matrix. The features have to be independent and discriminant. Rather than extracting features directly from the connectivity matrix, I have applied thresholding to the connectivity matrix. Thresholding the connectivity matrix means to use only the values in the matrix that meet a particular condition. Previous studies that are related to the thresholding of the connectivity matrix removed the edges with negative PCC and worked with only the edges with positive PCC [89], [90]. This is because the positively correlated regions tend to cluster together more. However, a negative PCC doesn't mean the ROIs don't have any connection or correlation between them. Instead, it means they have an anti-correlation [91], [92]. It indicates the activation of the ROIs is dissimilar or opposite to each other. There are a significant number of negative PCC values in every connectivity matrix. If the edges with negative PCC values are discarded, then a large amount of information will be lost. So, I have used both positive and negative PCC values.

Now, thresholding can be applied to remove weak edges or edges with small PCC values (both positive and negative). As mentioned, the range of PCC is in between +1 to -1, where +1 indicates the maximum correlation and -1 indicates the maximum anti-correlation between two ROIs. The

values close to zero for both positive and negative PCC represent decreased correlation and anti-correlation. In practice, the removal of weak edges plays a crucial role while extracting features. If no thresholding is applied, then all the nodes in the network will be connected to every other node in the network. As a result, the network will be very complex to analyze. Also, the contribution of the edges representing small PCC and the edges representing large PCC will be the same, which is not reasonable. So in this study, I have applied thresholding to remove weak edges (edges representing small PCC). I have also varied the thresholding values to find the optimal threshold.

2.3.3.2 Adjacency Matrix

In network theory, the adjacency matrix is a representation of a finite network. The elements of the matrix (i.e., edges of the network) indicate whether the nodes are adjacent to each other or not. In this study, for n nodes, I have calculated the adjacency matrix, $A = (a_{i,j})_{n\times n}$ by thresholding every element of the connectivity matrix, $CM = (cm_{i,j})_{n\times n}$ using different threshold values $T > 0$, as follows

$$a_{i,j} = \begin{cases} 1, & if\ cm_{i,j} \geq T \\ -1, & if\ cm_{i,j} \leq -T \\ 0, & if\ i = j \\ 0, & otherwise \end{cases} \qquad (2.2)$$

2.3.3.3 Laplacian Matrix

To calculate the Laplacian matrix, at first, I have calculated the degree matrix. The degree matrix is a diagonal matrix that contains information about the degree of each node. From the adjacency matrix, the degree matrix, $D = (d_{i,j})_{n\times n}$ is calculated as follows

$$d_i = \sum_{j=1}^{n} a_{i,j}\ ,\ if\ i = j \qquad (2.3)$$

The Laplacian of an undirected graph $G = (V, E)$ is a $n \times n$ matrix and is defined as follows

$$L(G) = D(G) - A(G) \qquad (2.4)$$

where $D(G)$ is the degree matrix and $A(G)$ is the adjacency matrix. So, the Laplacian matrix is the difference between the degree matrix and adjacency matrix. The Laplacian matrix, $L = (l_{i,j})_{n\times n}$ is calculated as follows

$$l_{i,j} = \begin{cases} d_{i,}, & if\ i = j \\ -a_{i,j}, & if\ i \neq j \end{cases} \quad (2.5)$$

I have extracted features from these matrices to train the machine learning classifiers. Specifically, topological centralities are calculated from the adjacency matrix, and the eigenvalues are calculated from the Laplacian matrix.

2.4 Summary

In this chapter, I have discussed in brief about the ABIDE dataset. Because of the difference of the scanner, scanning parameters, and phenotypic information over different sites, it is a very complex dataset. I have also described the preprocessing steps of the MRI images. I have used FSL and AFNI for the preprocessing. I have also described the process of defining nodes and edges to create brain networks. I have used the 264 ROI based parcellation scheme to create a brain network. For the simplicity of calculation, I have used different matrix representation of the brain networks, which I have described at the end of the chapter.

CHAPTER 3
DIAGNOSIS OF ASD USING EIGENVALUES OF BRAIN NETWORK: A MACHINE LEARNING BASED APPROACH

3.1 Introduction

The diagnosis of ASD is still based on interviews and observation. In the observation-based diagnosis, the patients perform a series of task and a clinician observe the social and communication skills of the patient. The clinician then scores the patient based on the observation. In the interview-based diagnosis, the parents of the patients are interviewed. They are asked different questions about the patient's behavior. Based on the answers, the clinician decides whether the patient has ASD. In both techniques, the diagnosis can be biased by the clinician and the parents. As ASD covers a wide range of symptoms, it is very hard to define a baseline for the tasks. Also, it is very hard for the parents to detect the abnormality of the behavior if they don't know what they are looking for. Hence, it is necessary to establish a diagnosis method that can diagnose ASD without human intervention.

Recent developments in machine learning have allowed researchers to look into the machine learning based diagnosis process of ASD using brain networks. In [13], [93], the weight of the edge is used as a feature to train machine learning classifiers. Also, features from the images have been used for classification. In [94] concavity, curvature, folding of the cortex is measured from the MRI images as features. Also, the spatial gradient among voxels is used for the classification of ASD in [95]. Morphological features, such as cortical thickness, subcortical volume, change in cortical thickness, are used to train machine learning classifiers in [96]. However, most of the studies either have a good classification accuracy on a small dataset or a poor classification accuracy on a large dataset.

The use of feature selection techniques is very common in machine learning based approaches. The feature selection algorithm tries to reduce the dimension of the feature vector so that only discriminating and independent features are used to train the machine learning classifiers. In [97], a simple feature selection method is described, where at first all the features are ranked and then the low ranked features are removed. In [98], a deep learning based feature selection model is

proposed. Another advantage of using feature selection algorithm is that it ensures overfitting doesn't occur while training the machine learning models.

In this chapter, I will discuss the features I have extracted from the brain networks. I will describe in brief how the features are extracted. I will also be proposing the use of the spectrum of the Laplacian matrix of the brain network as a raw feature. The feature set defined in this study contains the spectrum along with three other topology centralities. To solve the problem of overfitting, I will also propose a feature selection algorithm. Finally, I will discuss the results I have achieved through experiments where I have used the proposed features on some traditional machine learning classifiers. The main aim of this chapter is to focus on the effectiveness, robustness, and efficiency of the proposed features.

3.2 Feature Extraction

A feature is a measurable independent, informative, non-redundant, and discriminating property or characteristic of a data being observed [99]. Feature extraction, the process of defining features of a dataset, is crucial for machine learning classifiers. The features need to be sufficient enough to be able to classify. The efficiency of the classifiers largely depends on the defined features. In the case of well-defined features, the classification algorithms will be able to classify different classes of data more accurately. There are studies to extract features from the brain networks for the analysis of ASD [45]. However, the features defined in the previous studies fail to produce an acceptable classification result. So in this study, I am proposing the spectrum of the Laplacian matrix of the brain networks as a feature. The spectrum is combined with three topological centrality based features named assortativity, clustering coefficient and average degree of a network.

3.2.1 Spectrum of Brain Network

The set of all eigenvalues of a matrix is called its spectrum. The eigenvalues λ of a matrix M can be obtained by solving its following characteristic equation

$$P(\lambda) = \det(M - \lambda I) = 0 \qquad (3.1)$$

where I is an identity matrix with the same size of M. The spectrum of the Laplacian matrix of the brain networks is used as features. Spectrum is a unique representation of a particular matrix. The

spectrums are different for different matrices. However, in [100], it is shown that the eigenvalues can be used as a measure of similarity. It implicates, the similarity between two matrices can be captured using the spectrum. If two matrices are strongly correlated, the spectrum of the matrices is also strongly correlated.

As mentioned, the Laplacian matrix is calculated after applying a threshold to the connectivity matrix. So after threshold, the network represents edges with higher positive and negative PCC. The higher (positive and negative) PCC indicates the regions are more correlated with each other. The reason behind using spectrum is, if there is any relation in the activations of the ROIs of the ASD subjects then the Laplacian matrix will be able to represent it. Therefore, spectra of the Laplacian matrix calculated from the brain networks of the ASD subjects will have more similarity than the spectra of the HC subjects. As a result, the spectrum of the Laplacian matrix can be used as a feature.

3.2.2 Topology Centralities

Topological centrality reflects the topological position of nodes and edges in a network. It is the measure of the importance of nodes in the network. In the network theory there several topological centralities that can be measured to get an idea about the importance of different nodes. Through the centralities, it is possible to find out ROIs that are responsible for any particular disease, based on ROI's importance in the network. In [45], the authors studied different centralities, such as modularity, eccentricity, minimum and maximum global efficiency, strength, triangles, minimum and maximum path length for the machine learning based classification of ASD. Also, in [19] clustering coefficient, characteristic path length, local efficiency, global efficiency, normalized clustering coefficient, and normalized characteristic path length are studied for the detection of ASD.

In this study, I have measured global and local efficiency, average path length, graph strength, assortativity, clustering coefficient, and the average degree of a network as features for the machine learning classifiers. However, after applying feature selection algorithm (in Section 2.5) on all the available features, along with a part of the spectrum only assortativity, clustering coefficient and the average degree of a network are included in the final feature set. So in the following sections, I will only discuss selected topology centralities.

3.2.2.1. Assortativity

Assortativity (denoted by ρ) is the measurement of the tendency of the nodes to associate with each other based on their degree [101]. It is calculated as the cross-correlation of the degree of every pair of nodes connected through an edge. To measure the assortativity at first the adjacency matrix A for n nodes is transformed to $\overline{A} = (\overline{a}_{i,j})_{n\times n}$ as follows

$$\overline{a}_{i,j} = \begin{cases} 1, & if\ a_{i,j} > 0 \\ 0, & otherwise \end{cases} \qquad (3.2)$$

Then the assortativity is calculated as follows

$$\rho = \frac{\frac{1}{k}\left(\sum_{i,j\in V} d_i.d_j\right) - \left(\frac{1}{k}\left(\sum_{i,j\in V}\frac{1}{2}\left(d_i.d_j\right)\right)\right)^2}{\frac{1}{k}\left(\sum_{i,j\in V}\left(d_i^{\,2} + d_j^{\,2}\right)\right) - \left(\frac{1}{k}\left(\sum_{i,j\in V}\frac{1}{2}\left(d_i + d_j\right)\right)\right)^2} \qquad (3.3)$$

where for a network $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ with V set of nodes and E set of edges let i and j be any two nodes in the upper triangle of $\overline{A}$ connected through an edge and $i, j \in V$, k is the number of total non-zero elements in the upper triangle of $\overline{A}$, d_i and d_j be the respective degree of the node i and j. The assortativity is computed by using the MATLAB function defined in [102].

In some networks, similar nodes tend to connect with each other more [103]. There are different indices to measure the similarity, but most of the nodes connect to the nodes that have a similar degree [104]. In our brain network, nodes with similar degree mean they have a high correlation with other nodes, so they are more or less similarly active during the resting state.

3.2.2.2. Clustering Coefficient

In simple words, the clustering coefficient of a node is the fraction of triangles around it [105], a triangle in a network is a structure where three vertices are connected with each other. Clustering coefficient is the measurement of the local connectivity of a network. To measure the clustering coefficient at first, the adjacency matrix A is transformed to $\overline{A}$ using Eq. 3.2. Then the number of triangles of each node (denoted by β_G) is calculated as follows

$$\beta_G = \boldsymbol{diag}\,(\overline{A} \times U(\overline{A}) \times \overline{A}) \qquad (3.4)$$

where $diag$ is the MATLAB function which returns the diagonal elements of the matrix and $U(\overline{\mathbf{A}})$ is the upper triangular matrix of $\overline{A}$. Finally the clustering coefficient, C is calculated as follows

$$C = \frac{1}{f}\left(\sum_{i \in V} 2 \times \left(\frac{\beta_G(i)}{d_i \times (d_i - 1)}\right)\right) \qquad (3.5)$$

where for a network $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, f is the total number of nodes in the network and d_i is the degree of node i.

Clustering coefficient is a prominent feature of the network that shows small-world properties. According to [106], brain networks show small-world network properties. A study for Alzheimer disease detection [107] using network analysis on fMRI showed that the high average clustering coefficient of a network could be interpreted as densely connected local clusters. Also, in [19], it is mentioned that the ASD subjects have lower clustering coefficient than the HC subjects. So, using this topological centrality as a feature, the classifiers might perform better.

3.2.2.3. Average Degree of a Network

The average degree (denoted by Q) of a network is the ratio of the number of all the edges to the number of all nodes in a network, and it can be calculated from the adjacency matrix $A = (a_{i,j})_{n \times n}$ as follows

$$Q = \frac{2}{f} \times \sum_{i=1}^{f} \sum_{j=1}^{f} a_{i,j} \qquad (3.6)$$

where for a network $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, f is the total number of nodes in the network.

The average degree of a network is an overall representation of the number of edges in a network. So, the average degree of a network is similar for similar networks. Various studies have shown that there is some similarity in the functional activity of the brain for ASD subjects [28], [29], [108]. Due to the similarity of the functional activity, the adjacency matrix representing the connection (edges) in the brain networks also have a similarity. However, due to the difference in the functional activity of the ASD subjects and HC subjects, the average degree of the network proves to be a discriminating feature for the machine learning classifiers.

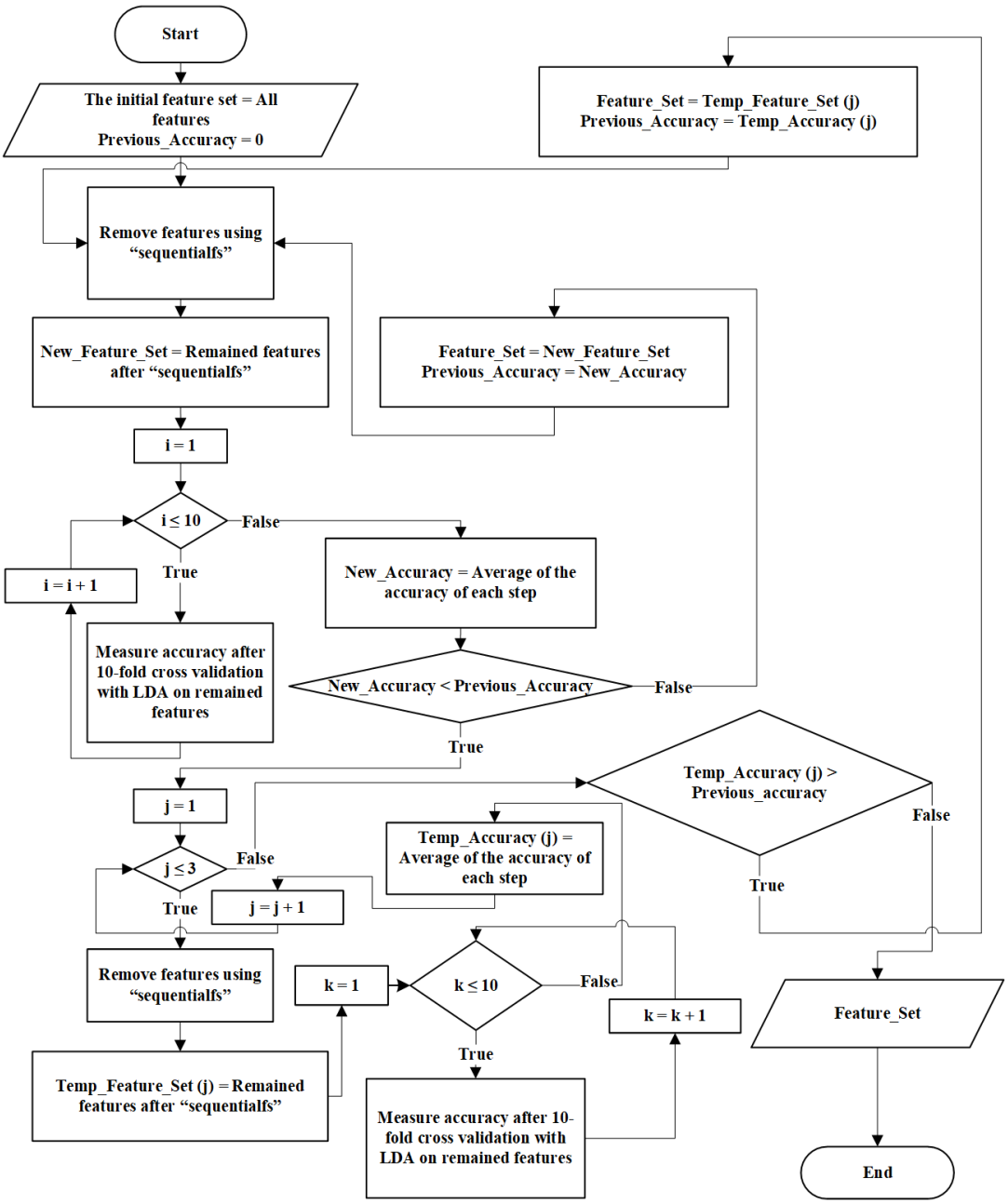


***Figure 3-1**: Flowchart of the feature selection algorithm*

3.3 Feature Normalization

Feature normalization is a vital data preprocessing step. Feature normalization is necessary when the magnitude and range of the features are different from each other. In the experiments, the range of assortativity and clustering coefficient is 0 to 1, and the range of the spectrum and the average degree of network is $+\infty$ to $-\infty$. In some machine learning classifiers, the higher ranged features influence the classification results due to its large value. So to resolve this issue, the spectrum and

average degree of the network are normalized before applying the feature selection algorithm. The features are normalized independently as follows

$$z_i' = \frac{z_i - \min(z)}{\max(z) - \min(z)} \qquad (3.7)$$

where $z_i \; and \; z_i', i = 1, \ldots, f$ are original values and normalized values, respectively and f is the dimension of feature vector z.

3.4 Feature Selection

Feature selection is the process of selecting a subset of relevant features (variables, predictors) for use in machine learning classifiers [109]. The machine learning classifiers can achieve a better performance if the features are discriminate. The performance tends to increase as the features are more discriminant. Unnecessary features make the model complex, reduce the efficiency of the machine learning classifiers. It also overfits the model. These problems can be solved by using a feature selection algorithm [110]. The main purpose of the feature selection algorithms is to find the minimal subset of the most discriminate features [111] while improving the prediction accuracy of the model [112].

Out of different feature selection algorithms, the most popular one is the sequential feature selection. The algorithms are called sequential because of the iterative nature. The main component of the feature selection algorithm is the objective function. The feature selection algorithms try to minimize this objective function through different criteria, such as mean squared error, misclassification rate. So, in the forward sequential feature selection, initially, there is an empty feature set. In the first step, one feature is added from the available features, which give the highest value for the objective function [113]. In the next steps, new features are added sequentially, and the new subset is evaluated. If adding a feature to the subset increase the value of the objective function, then the feature is added permanently to the subset. This process continues, until adding a new feature doesn't change or reduce the value of the objective function. There is another variant of the feature selection algorithm, backward sequential feature selection. Rather than an empty set, it starts with a set of all the features. Then during each step, features are removed sequentially. This process continues until removing any features doesn't increase the value of the objective function. I have used the backward sequential feature selection algorithm in this study. Specifically

MATLAB function “sequentialfs” is used to perform the backward sequential feature selection algorithm with linear discriminant analysis (LDA) and 10 fold cross-validation.

The **“**sequentialfs**”** select a subset of features from the available features that does the best prediction by sequentially removing features until removing more features reduce the prediction accuracy [114]. So, the “sequentialfs” starts with a feature set of all the features. For each step, candidate feature subsets are created by sequentially removing features from the original feature set. After that, 10-fold cross-validation is performed on each feature subset. During the cross-validation step, the data is divided into training and testing set. The data are trained and tested on the linear discriminant analysis (LDA), which is a machine learning classifier. The classification accuracy of LDA is the objective function. So to optimize the objective function, it is necessary to maximize the classification accuracy. For any particular feature subset, for each fold of the 10-fold, the classification accuracy is measured. The average classification accuracy over all the folds is the accuracy of that feature subset. So, the classification accuracy is measured for all the subsets. After each step, the feature subset for which the classification accuracy is the maximum is the selected feature subset of that step. In the next step, the selected feature subset of the last step is the original feature set, and features are removed from that set. This process continues until removing more features does not increase the classification accuracy.

Table 3-1*: Number of features selected after using the feature selection algorithm*

Thresholding Condition	**Number Of Selected Features After Feature Selection**
All edges with positive PCC	88
All edges with negative PCC	76
$\boldsymbol{T = 0}$	66
$\boldsymbol{T = 0.1}$	103
$\boldsymbol{T = 0.2}$	62
$\boldsymbol{T = 0.3}$	78
$\boldsymbol{T = 0.4}$	77
$\boldsymbol{T = 0.5}$	111
$\boldsymbol{T = 0.6}$	37

The **“**sequentialfs**”** algorithm divides the data only once at the beginning of the computation and tries to optimize the classification accuracy with those sets. This algorithm stops before reaching the optimal feature subset. To resolve this problem, I have implemented an iterative backward sequential feature selection algorithm. During each iteration, at first, a feature subset is created

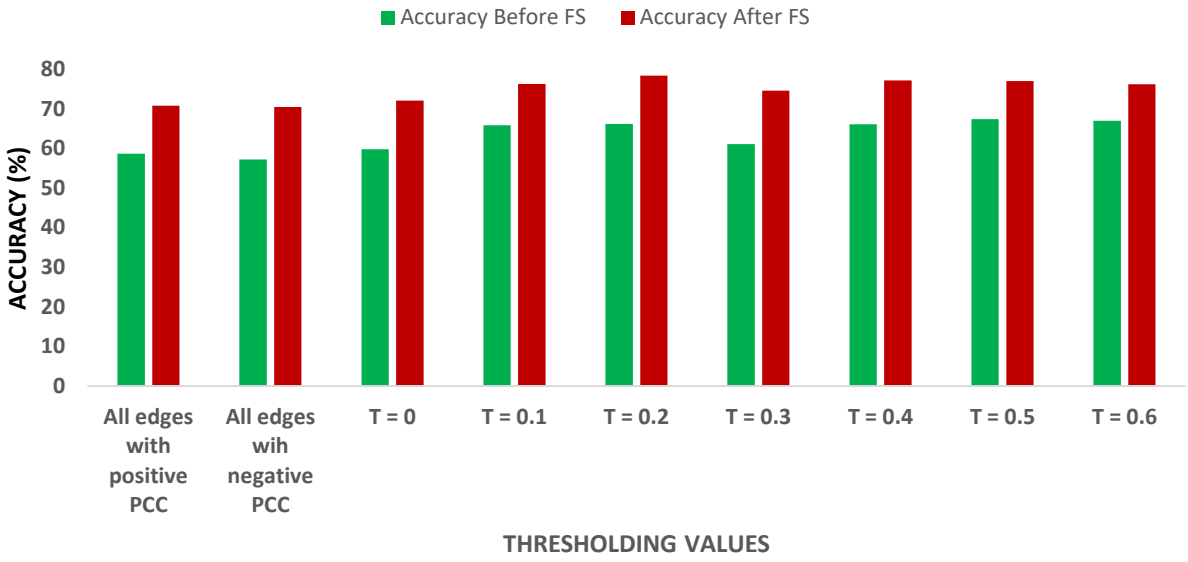


***Figure 3-2**: The classification accuracy of LDA before and after applying the feature selection (FS) algorithm*

using the **"**sequentialfs**"** by removing redundant features from the previously remaining feature subset. Then, 10-fold cross-validation with LDA is performed on the remaining feature subset. The 10-fold cross-validation is repeated ten times using the same feature subset. The average classification accuracy over each step is the classification accuracy of that subset. The purpose of using the 10-fold cross-validation multiple times is to ensure the classifier isn't biased by any particular set of data. Now, if the accuracy of the current step (new accuracy) after the cross-validation, is lower than the accuracy of the previous step then features are further removed using "sequentialfs" from the newly remained feature subset. The removal of features is performed three times, and each time "sequentialfs" is used on the previously remaining feature subset. This step is used because sometimes even though the accuracy of the next step is lower than the current step, but the accuracy of the consecutive steps is better than the current step. Measuring the accuracy of the following three steps ensured the algorithm wouldn't stop before reaching a minimum subset of features with the maximum accuracy. Again, for measuring the accuracy of each step of three steps, I have used the 10-fold cross-validation ten times mentioned previously. Now, if the accuracy of any of the three steps is higher than the new accuracy, then the feature subset of that step is considered as the feature subset for the next iteration. However, if the new accuracy is lower than the previous accuracy and the accuracy of the future three iterations are also lower than the

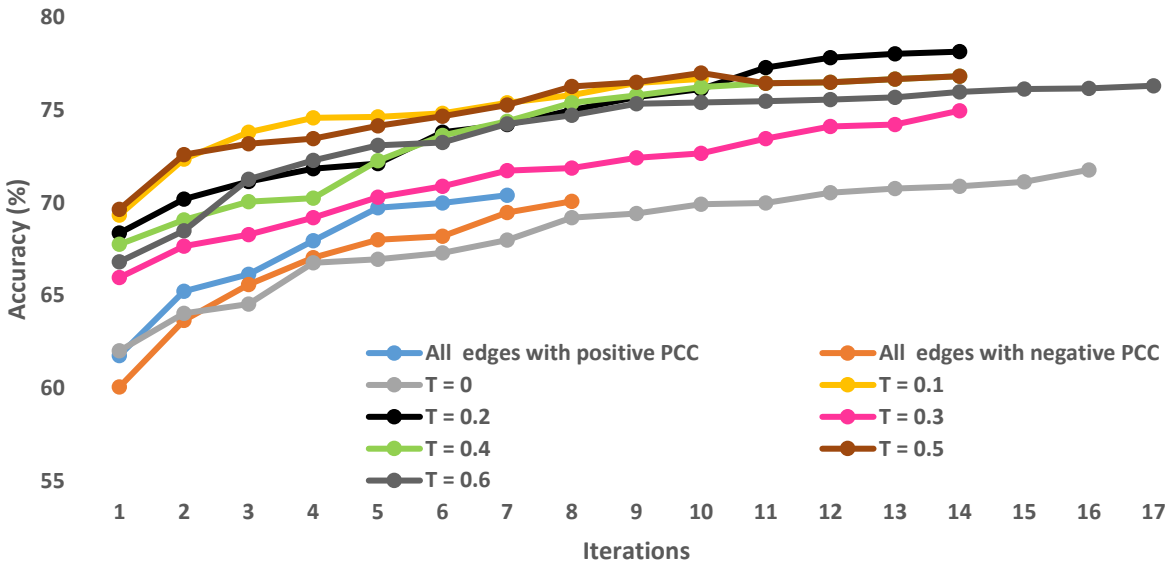


Figure 3-3: *Change of the classification accuracy during each iteration of the feature selection*

new accuracy, the current feature subset is considered to be the final set of features for the classifiers. The mentioned feature selection algorithm is illustrated in Figure 3-1.

I have implemented the feature selection algorithm to define independent and discriminating features and to avoid the overfitting issue. One of the reasons that machine learning algorithms are overfitted is due to the high dimensionality of features. Most times, the study of the neuroimages includes a high dimensional feature set due to the use of a large number of voxels, or ROIs. As a result, there are noisy or unnecessary features in the feature set. The noises in the feature set profoundly bias the performance of the machine learning algorithms. Machine learning algorithms try to find a pattern in the feature set. However, if there are too many noises in the feature set then machine learning algorithms search for a pattern in the noises, which isn't an actual representation of data. So, excessive features overfit machine learning algorithms by introducing noises into the feature set.

It can be summarized from Table 3-1 that the number of selected features has reduced significantly from 267 raw features after applying the feature selection algorithm. Figure 3-2 shows a comparison of the classification accuracy before and after applying the feature selection algorithm for different thresholding conditions. From the results in Figure 3-2 it can be seen that the classification accuracy has increased for all the thresholding conditions after applying the feature selection algorithm, which indicates that the method has avoided the overfitting by reducing the

dimension of the feature set. Figure 3-3 shows the change of the classification accuracy during each iteration of the feature selection algorithm. From Figure 3-3, it can be seen that the classification accuracy has increased by removing noises from the feature set.

3.5 Results and Discussion

3.5.1 Effect of Threshold

One of the crucial steps of defining the matrices from the brain networks is applying the threshold. The threshold is applied to the connectivity matrix to create the adjacency matrix, from which the degree matrix and the Laplacian matrix is calculated. Now, strong edges (both positive and negative large PCC) in the connectivity matrix imply that the corresponding ROIs are similarly activated during the same time point, where weak edges (both positive and negative small PCC) indicate that the ROIs are active, but the difference or dissimilarity of the activation is large. According to Eq. 2.2, if the positive PCC value is larger than the threshold value T then it is replaced by +1 in the adjacency matrix, and if the negative PCC value is smaller than the negative of the threshold T then it is replaced by -1. In this approach, the contribution of both large and small (positive and negative) PCC values are considered to be the same. Because either large PCC or small PCC both are replaced by a +1 or -1, respectively. Applying the threshold will remove the weak edges from the connectivity matrix. However, large threshold values will make nodes of the brain network sparse. On the contrary, weak edges also carry some information about the relation of the ROIs. So, applying a high threshold value, more information will be lost. It is necessary to find a threshold value, which will remove a sufficient number of weak edges to make the network sparse without losing too much information.

In this study, I have experimented with different threshold values T from 0 to 0.6, with an increment of 0.1. For, threshold values greater than 0.6, most of the edges are removed. As a result, there is too little information in the adjacency matrix to extract. So, I have experimented with only the mentioned threshold conditions. Along with these, I have also conducted the study with the edges with only positive PCC values and only negative PCC values. The purpose is to prove that rather using than one type of edges (positive or negative PCC values), combining the edges perform better.

***Table 3-2**: Comparison of performance of different machine learning classifier for different thresholding values*

Thresholding Condition	LDA (%)		LR (%)		SVM (%)		KNN (%)		NN (%)	
	ACC	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**
All edges with positive PCC	70.8	76.0	70.1	75.0	66.88	71.0	67.3	76.0	66.0	72.0
All edges with negative PCC	70.5	74.0	69.2	74.0	65.1	70.0	63.9	71.0	64.4	68.7
$\boldsymbol{T = 0}$	72.1	77.0	71.4	78.0	63.0	77.0	61.7	71.0	66.0	73.0
$\boldsymbol{T = 0.1}$	76.3	82.0	76.5	81.0	72.2	77.0	73.2	80.0	71.3	77.5
$\boldsymbol{T = 0.2}$	**78.4**	**83.0**	**77.0**	**83.0**	**73.5**	**77.0**	**73.8**	**81.0**	**71.7**	**78.7**
$\boldsymbol{T = 0.3}$	74.6	78.0	73.6	78.0	67.4	72.0	71.2	78.0	68.3	74.5
$\boldsymbol{T = 0.4}$	77.2	82.0	75.2	81.0	71.6	76.0	72.7	80.0	71.3	77.1
$\boldsymbol{T = 0.5}$	77.0	81.0	75.4	81.0	72.7	78.0	73.2	80.0	70.1	76.9
$\boldsymbol{T = 0.6}$	76.2	81.0	76.0	81.0	71.8	77.0	72.9	80.0	68.4	75.5

The adjacency matrix, $A = (a_{i,j})$ (Eq. 2.2) is modified for the calculation of edges with only negative PCC values as follows

$$a_{i,j} = \begin{cases} 0, & if\ cm_{i,j} > 0 \\ 1, & if\ cm_{i,j} < 0 \end{cases} \qquad (3.8)$$

Also for edges with only positive PCC and edges with only negative PCC, the first eigenvalue is discarded before applying the feature selection. Because there is no variance in them.

The process of classification is different for different machine learning algorithms. As I have proposed eigenvalues of the brain network as raw features, I have experimented with different machine learning algorithms to find out which machine learning algorithm performs the best for the proposed set of features. I have used LDA, logistic regression (LR), support vector machine (SVM), K-nearest neighbor (KNN) and neural network (NN) to classify ASD and HC subjects. MATLAB is used to implement LDA, LR, SVM, and KNN. NN is implemented in PyCharm with python.

I have used the KNN classifier with cosine as the distance metric. The SVM is used with the medium Gaussian kernel. The LR with the default value is used on the classification learner app

***Table 3-3**: Classification accuracy and AUC of the different sites of ABIDE 1*

Sites	ASD Subjects	HC Subjects	Total Subjects	Features After FS	LDA (%)		LR (%)	
					ACC	AUC	ACC	AUC
PITT	24	26	50	15	100	100	100	100
OLIN	14	14	28	14	100	100	92.9	99
OHSU	12	13	25	2	100	100	100	100
SDSU	8	19	27	7	92.6	96	96.3	99
TRINITY	19	25	44	19	97.7	100	97.7	100
UM 1	34	52	87	79	97.7	97	87.2	90
UM 2	13	21	34	14	100	100	97.1	99
UM	47	73	120	54	92.5	98	94.2	95
USM	43	24	67	24	98.5	100	97.0	99
YALE	22	19	41	5	100	100	100	100
CMU	6	5	11	3	100	100	100	100
LEUVEN 1	14	14	28	11	96.4	100	96.4	100
LEUVEN 2	12	16	28	13	100	100	100	100
LEUVEN	26	30	56	27	100	100	96.4	100
KKI	12	21	33	7	97.0	100	93.9	99
NYU	74	98	172	5	99.4	100	100	100
STANFORD	12	13	25	12	100	100	100	100
UCLA 1	37	27	64	13	100	100	96.9	100
UCLA 2	11	10	21	9	100	100	100	100
UCLA	48	37	85	17	100	100	100	100
MAX_MUN	19	27	46	24	100	100	100	100
CALTECH	5	10	15	6	93.3	100	93.3	100
SBL	12	14	26	1	100	100	100	100

of MATLAB [115]. The NN is built with the TensorFlow. An NN consisting of one input layer, one hidden layer, and one output layer is used. In the input layer, the number of neurons equals the number of selected features. “Softmax” is used as the activation function after the input layer. Then a fully connected hidden layer with 2 neurons is used. Rather than using the stochastic gradient descent, I have used the adam optimizer to update the weights. For the machine learning algorithms, I have used 10-fold cross-validation to measure accuracy (ACC) and area under the receiver operating characteristic (ROC) curve (AUC). However, for the NN I have split the data into a training set (80%), testing set (10%), and validation set (10%). I have used the training set and testing set to create the NN model and the validation set to measure the final accuracy. To show the stability of the classification process, I have repeated the experiments 10 times for the

machine learning algorithms and NN. The average of the ACC and AUC, along with the standard deviation (stdv) of the experiments are shown in Table 3-2.

It is evident from Table 3-2 removing the weak edges produce a better result than either all the edges with positive PCC or all the edges with negative PCC. The results are even better for including all the edges than either type of edges. For the threshold value $T = 0.2$, all the machine learning classifiers and NN achieve their best performance. In most of the cases, LDA outperforms other machine learning classifiers. The best classification accuracy of 78.4% is achieved for the threshold value of 0.2 and LDA as the classifier. Using the simplest NN, I have achieved the highest classification accuracy of 71.7%, which is better than the study in [15].

3.5.2 ASD Classification for Different Sites of ABIDE 1

The inter-site variation of ABIDE 1 dataset is very prominent, as it covers a wide range of age groups, sexes, scanning parameters, and scanner types (Table 3-1 and Table 3-2). This represents the real-world scenario, where the scanner or scanning parameters cannot be controlled uniformly at different sites or institutions. However, for a particular institution, it is very unlikely for them to have a prominent difference in the scanner or the scanning parameters. Therefore, the classification of the intra-site classes is also important. Therefore I have conducted the experiments on 17 sites of ABIDE 1 database separately. For intra-site experiments, I have kept the threshold value T= 0.2 to be consistent. I have applied the feature selection method to the dataset from each site, separately. In all the cases, the number of selected features is smaller than the number of subjects in the dataset, which again illustrates the efficiency of the feature selection algorithm to avoid the overfitting issue. The ACC and AUC are calculated based on the 10-fold cross-validation with LDA and LR and are shown in Table 3-3. UCLA, UM, and LEUVEN sites have two different datasets. I have calculated the accuracy based on each dataset separately and, as well as the entire dataset. From Table 3-3, it can be seen that defined and selected features do an excellent job for classifying ASD and HC subjects in case of the intra-site dataset. Specifically, the proposed process can diagnose ASD with an average ACC of 98.5% ± 2.5% and an average AUC of 99.6% ± 1.1% for all the sites using LDA. At some sites, the accuracy and/or the AUC have reached 100%. The efficiency of the defined and selected features is illustrated by not only LDA but also LR, which also has the average ACC of 97.3% ± 3.3% and the average AUC of 99.1% ± 2.3%.

3.6 Summary

The main objective of this chapter is to describe the features and a feature selection process for the diagnosis of ASD. I have explained in detail the spectrum and topological centralities of the brain network. I have also presented my modified feature selection algorithm in this chapter. Using the selected features, I have trained different machine learning classifiers. The results of the classification are reported in this chapter. In the results, the effect of changing threshold value is also described. Along with inter-site classification, intra-site classification is also carried out and reported. Using my proposed features and feature selection algorithm, I have achieved a classification accuracy of 78.4% using LDA, which is better than the state-of-the-art studies.

CHAPTER 4
DEEP LEARNING METHODS TO DIAGNOSE ASD USING BRAIN NETWORKS

4.1 Introduction

In Chapter 3, I have discussed the use of traditional machine learning classifiers for the diagnosis of ASD. The recent development in neural network algorithms has inspired new approaches to study brain diseases. As mentioned, in ASD studies, deep learning has been used as both feature extractor and classifier. In this chapter, I will discuss the use of autoencoders for the diagnosis of ASD.

An autoencoder is a type of artificial neural network, where the training of the network is done in an unsupervised manner. Autoencoder compresses the data and creates an encoded representation of the data. Then from the compressed data, autoencoder tries to regenerate the actual data. In [116], CNN is proposed where they have used the autoencoder to detect the structural shape variation from the MRI scans for the classification of Alzheimer's Disease (AD). The AD-based study [117] used stacked autoencoder for unsupervised feature learning for AD diagnosis. In an ASD based study [49], a stacked denoising autoencoder is used in the pre-training step for the extraction of the lower-dimensional data representation of the whole dataset. Another use of stacked autoencoders for feature extraction is described in [118], where they have extracted features from the brain networks and used two stacked autoencoders connected to a softmax function for classification.

The autoencoder can be used both as a feature extractor and classifier for the diagnosis of ASD. Deploying the autoencoder as feature extractor, the latent view representation of the data that is being created through the encoder is considered to be the compressed features, and these features are then analyzed for diagnosis [116]. On the other hand, using the autoencoder as a classifier, the information learned from the autoencoder is incorporated into a classifier model for the classification [118].

In the case of feature extractor, the popularity of autoencoder is because it can approximate the nonlinear relation of features. Filter, wrapper, and embedded are some of the common feature selection methods. In the filter method, the features are selected based on their scores in various statistical tests for their correlation with the outcome variables. Filter methods don't use any

classifier algorithms to predict performance. In the wrapper method, the selection of features is done through a searching process. In this method, subsets of features are selected from the actual feature set, and the subset which produces the maximum performance for the classifier algorithm is considered to be the final feature set. The feature selection process described in the previous chapter (Chapter 3) use the wrapper method. In the embedded method, a model tries to learn the features that best contribute to the accuracy of the model while the model is being created. Regularizations are most common embedded feature selection method. However, all these methods can only find the linear relation among the features. The embedded method can be used to find the nonlinear relationship of the features by using a nonlinear kernel, but the learning of the model greatly depends on the kernel [119]. Autoencoder can analyze the nonlinear relation among the features using the nonlinear activation functions. Also, in [119], the authors have shown that an autoencoder based feature extraction scheme works better than the traditional feature selection algorithms.

In this chapter, at first, I will discuss the dataset and data pre-processing. Then I will talk in brief about the autoencoder and how it works. I will describe the autoencoder that I have used in this study, and I will also show that the learning of the autoencoder can be used in a DNN classifier for the diagnosis of ASD. After that, I will describe the autoencoder based feature extraction method. There is a significant increase in the performance of the traditional machine learning classifiers by using the autoencoder based feature extraction, which I will describe in this chapter. Finally, I will discuss the results of different experiments and conclude the chapter.

4.2 Materials and Methods

4.2.1 Data Preprocessing

To be consistent and make the study comparable with other studies, I have worked with the same 871 subjects from the ABIDE 1 dataset mentioned in the previous chapters. There are 403 ASD subjects and 468 HC subjects out of a total of 871 subjects. Using the rs-fMRI images of the subjects, I have divided the brain into 264 ROIs and then extracted the time-series measurement of the ROIs. The 264 ROIs are the nodes, and the PCC of the time-series measurement between any two nodes are the edges. So, using these nodes and edges, I have created brain networks from the rs-fMRI images. The processing steps are the same as mentioned in chapter 2.

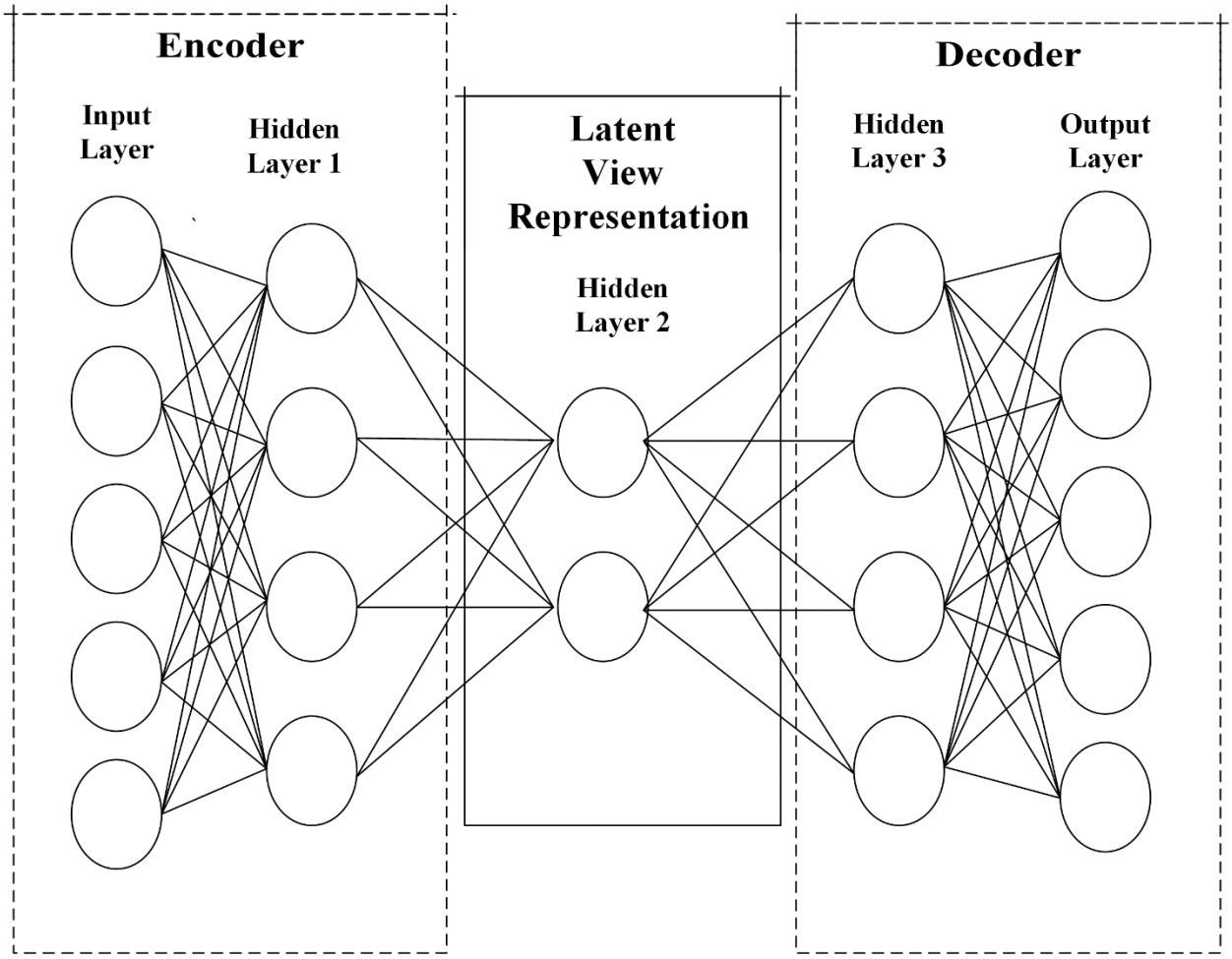


***Figure 4-1**: Example of a simple autoencoder*

4.2.2 Feature Extraction

The feature extraction process is the same as described in Chapter 3 (Section 3.2). At first, I have created a connectivity matrix from the brain networks for individual subjects. Then from the connectivity matrix, I have created the adjacency matrix, degree matrix, and Laplacian matrix. I have used the same spectrum, assortativity, clustering coefficient, and average degree of the network as features.

4.2.3 Feature Normalization

The feature normalization is an important step for machine learning classifiers as uneven features may bias results. Therefore, I have normalized features before applying them to the DNN. I have used MATLAB to calculate the spectrum of the Laplacian matrix. As the size of the Laplacian matrix is 264 × 264, so there are 264 eigenvalues for each subject. The eigenvalues are sorted in

ascending order for each subject and they can range from $-\infty\ to\ +\infty$. However, when eigenvalues (say the first eigenvalue of each subject) are normalized, the contribution of each eigenvalue is measured over all the subjects. The contribution of that eigenvalue for that subject is ignored in this scenario. So, in this study, I have normalized the eigenvalues for a particular subject rather than normalizing each eigenvalue over all subjects by Eq. 3.7. After this normalization, the maximum value and the minimum value in the spectrum of each Laplacian matrix are 1 and 0, respectively, which are excluded from the raw feature set. I haven't applied normalization to the assortativity and the clustering coefficient, as these are normalized when calculated. However, the average degree of the network is normalized over all the subjects using Eq 3.7.

4.2.4 Autoencoder

4.2.4.1. Introduction

In simple words, an autoencoder is a neural network that learns to replicate its input. Figure 4-1 shows the architecture of a simple autoencoder. There are three main components of the autoencoder. They are described as follows

- **Encoder:** The main work of this component is to compress the input data and create a latent view representation. There might be one or more hidden layers in the encoder. Usually, the number of neurons is reduced in consecutive hidden layers, limiting the amount of information that can flow through the network. As a result, the network learns the most important features of the input data.
- **Latent space representation:** This is the compressed representation of the input data.
- **Decoder:** The last part is the decoder. Decoder tries to reconstruct the input data from the latent space representation. Usually, in the decoder, the number of neurons gradually increases in the consecutive layers. The number of neurons in the output layer of the decoder is the same as the number of neurons in the input of the encoder.

The autoencoder is trained by comparing the reconstructed data with the actual input data and then trying to minimize the prediction error. The autoencoder uses unsupervised training, i.e. no label

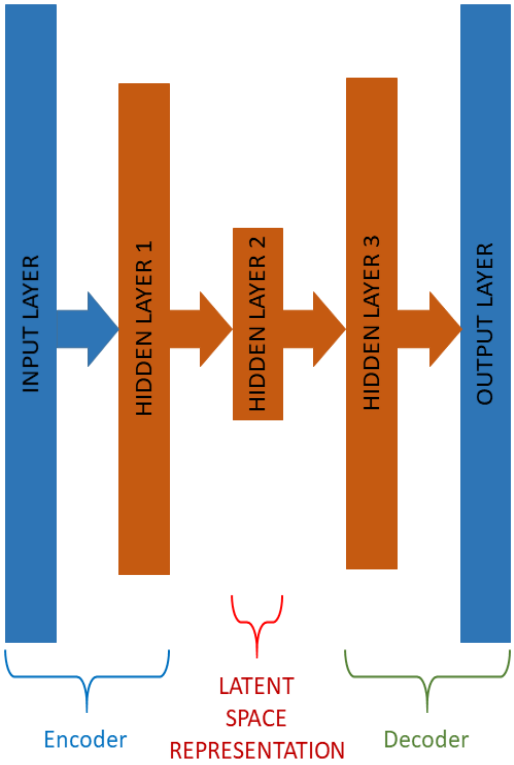


***Figure 4-2**: Block representation of the proposed autoencoder architecture*

of the input data is required for the training of the network. Rather than using the autoencoder to copy the input information to the output, the latent space is used to teach the network the useful attributes of the data. This is done by constraining the learning of the network. As the dimension (the number of neurons) is reduced in the latent space, the autoencoder is forced to learn the salient features of the training data.

4.2.4.2. Proposed Autoencoder

The block representation of the autoencoder I have used in this study is shown in Figure 4-2. There is a total of four hidden layers, one input layer, and one output layer in my proposed autoencoder. The encoder part of the proposed autoencoder is the input layer and the hidden layer 1. The hidden layer 2 creates the latent space representation of the input data. Hidden layer 3 and the output layer is the decoder of this model.

Figure 4-3 shows the layers and neurons of the proposed autoencoder. In the consecutive layers of the encoder, I have decreased the number of neurons from 267 (input layer) to 200 (hidden layer 1). Then the information learned by the encoder is projected into the latent space representation through 10 neurons (hidden layer 2). So, these ten neurons are learning to represent the input data

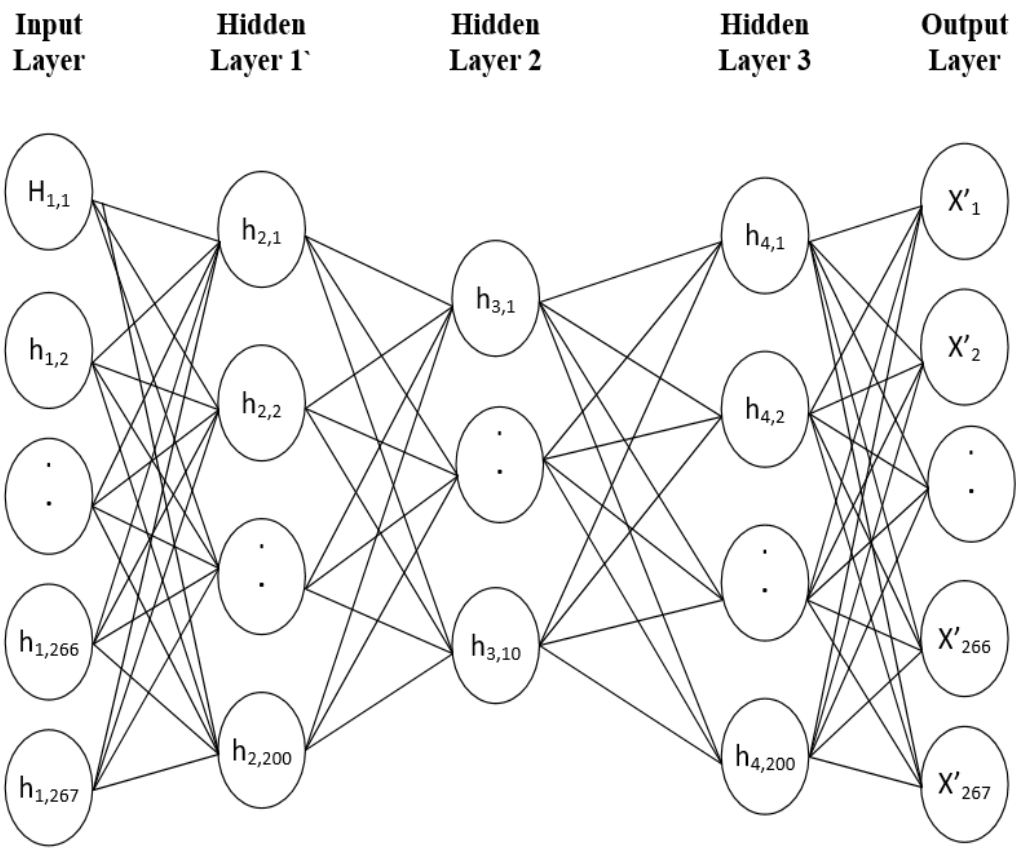


***Figure 4-3**: Architecture of the proposed autoencoder*

(267 features). Then in the decoder, I have increased the neurons from 200 (hidden layer 3) to 267 (output layer). The decoder will reconstruct the data from the latent view representation. So, the autoencoder is trying to recreate x' from the input data x by minimizing the reconstruction error, $L(x, x')$, where L is the measurement of the difference between original input and the consequent reconstruction. The network is trained to minimize $L(x, x')$. The description of the layers are as follows

- **Input Layer :** Neurons = 267 (number of features)
- **Hidden Layer 1 :** Neurons = 200, Activation = '**relu**'
- **Hidden Layer 2 :** Neurons = 10, Activation = '**relu**'
- **Hidden Layer 3 :** Neurons = 200, Activation = '**relu**'
- **Output Layer :** Neurons = 267 (number of input features), Activation = '**sigmoid**'

I chose the layers after experimenting with different layers and activation functions. A well-trained autoencoder should be sensitive to the input data so that it can create an accurate reconstruction.

Also, it should be insensitive enough so that it doesn't copy the data from the input to the output. These two aspects are controlled by the reconstruction error ($L(x, x')$) and the number of neurons in the network. The reconstruction error controls how sensitive the network will be to the input, and the number of neurons ensures the network doesn't copy the information.

4.2.4.3. Autoencoder Based Classifier

In case of a simple autoencoder (one input layer, one hidden layer, and one output layer), in response to an input $x \in \boldsymbol{R}^n$, the activation of each neuron, $h_i, i = 1, \ldots, m$ is

$$h(x) = f(W_1 x + b_1) \qquad (4.1)$$

where $f(a)$ is the nonlinear activation function applied to every neuron, $h(x) \in \boldsymbol{R}^m$ is the neuron activation, $W_1 \in R^{m \times n}$ is a weight matrix and $b_1 \in \boldsymbol{R}^m$, is a bias vector.

The output of the network is as follows

$$x' = f(W_2 h(x) + b_2) \qquad (4.2)$$

where $x' \in \boldsymbol{R}^n$ is the output of the network, $W_2 \in R^{m \times n}$ is a weight matrix and $b_2 \in \boldsymbol{R}^m$, is a bias vector.

During the training for a set of p input data $x_i, i = 1, \ldots, p$, the weight matrices W_1 and W_2, the bias vector b_1 and b_2 are updated in the backpropagation so that the reconstruction error, $L(x, x')$ is minimized while creating a set of p output data $x'_i, i = 1, \ldots, p$. The reconstruction error is calculated as follows

$$L(x, x') = \sum_{i=1}^{p} \left\| X_i - X'_i \right\|^2 \qquad (4.3)$$

So in the proposed autoencoder (Figure 4-3), The weight matrices $W_k,\ k = 1, \ldots, 3$ and bias vectors $b_l,\ l = 1, \ldots, 4$ are updated by calculating the activation of the neurons $h_{i,j},\ i = 1, \ldots, 3$ and $j = 1, \ldots, r$ in each layer using Eq. 4.1, where r is the number of neurons in each layer. The objective of the network is to minimize Eq. 4.3 and the output is calculated using Eq. 4.2.

As mentioned, the decoder recreates the input data in the output from the latent space representation. So, if the reconstruction error is minimum, it means the latent space have learned

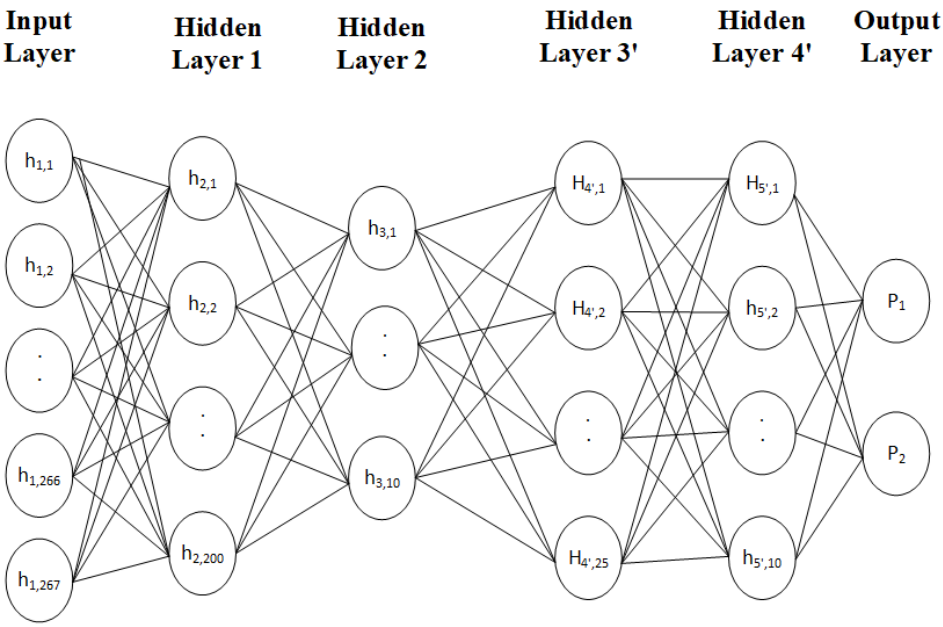


***Figure 4-4**: Proposed deep neural network classifier*

the salient features of the input data. These features can be used in a classifier for the purpose of classification between ASD subjects and HC subjects.

The architecture of my proposed DNN classifier is in Figure 4-4. The proposed model is chosen as it produces the most accurate and stable classification results. The architecture of the DNN from the input layer to hidden layer 2 is the same as the encoder and latent space representation of the autoencoder (Figure 4-3). However, after the latent space representation instead of the decoder I have used two new layers (hidden layer 3' and hidden layer 4'). Finally, the probability of data belonging to a particular class is shown through the output layer. The description of the new layers are

- **Hidden Layer 3' :** Neurons = 25, Activation = '**tanh**
- **Hidden Layer 4' :** Neurons = 10, Activation = '**tanh'**
- **Output Layer :** Neurons = 2, Activation = '**softmax'**

In the proposed classification process at first, the autoencoder is trained to reconstruct the input data. After completing the training of the autoencoder the DNN is trained. The first two hidden layers of the DNN are pre-trained using the weights and biases of the first two hidden layers of the autoencoder. The assumption behind this approach is, as the latent space representation has learned

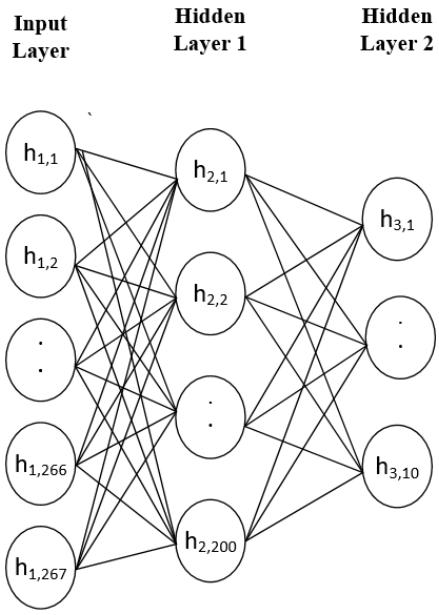


Figure 4-5: *Feature selector part of the autoencoder*

to find the salient features of the input data, these features should be able to classify between ASD and HC subjects more accurately.

4.2.4.4. Autoencoder Based Feature Extractor

The autoencoder is trained in an unsupervised manner. So, no label information is used to train the autoencoder. When training the autoencoder, I have used the data from both ASD and HC subjects. Even though the latent space representation created a discriminate and salient representation of the input data, the difference in the features between different classes is minimal. So, the features in the latent space representation don't produce a satisfactory result when used for classification. However, when the DNN classifier model is trained, the weights in the latent space representation

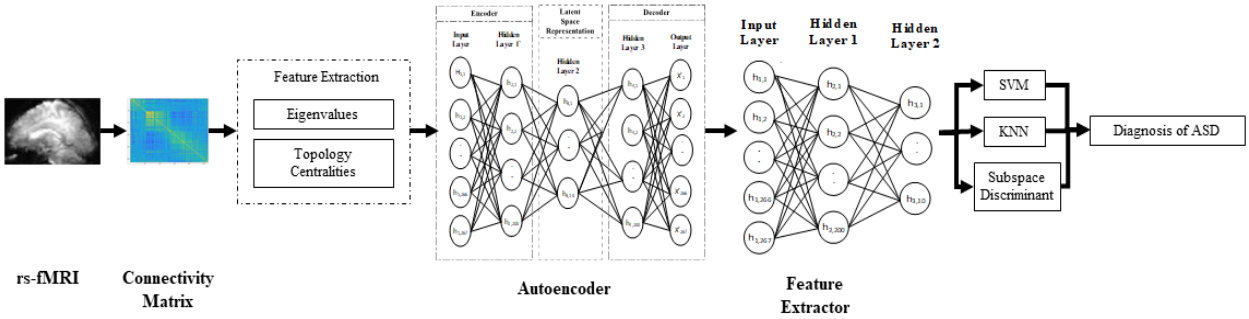


Figure 4-6*: Illustration of the main steps of using autoencoder as feature extractor*

is updated in such a manner so that it can create a representation of the input data which is discriminate enough to classify between two classes.

After training the DNN classifier, the first three hidden layers of the DNN model which are also the encoder and latent space representation part of the autoencoder is used as a feature extractor. Because when training the DNN classifier, the latent space is forced to create a representation that will help the DNN model to classify better. So along with creating a compressed representation of the input, it is also creating a representation that is discriminate for different classes. As a result, this model can be used to create a compressed and discriminate representation of the features from the available feature set. The feature selector model is shown in Figure 4-5.

4.3 Results and Discussion

At first, I have implemented an autoencoder based feature extraction method. As mentioned, the decoder reconstructs the input data in the output from the latent space representation. Therefore, if the reconstruction error is small enough, it means the latent space have learned the salient features of the input data. Thus, the latent space can produce discriminate and salient representation (high-level features) of the input data. These features can be used in machine learning algorithms for the purpose of classification between ASD subjects and HC subjects. To evaluate the performance of the autoencoder based feature extractor, I have divided the entire dataset into 80% training data (697 subjects) and 20% testing data (174 subjects). Then I have trained the autoencoder using only the training data. After completing the training, the autoencoder is used to extract features from the testing data. Then using the extracted features of the training data I have trained all the machine learning algorithms available in the classification learner toolbox in MATLAB. However, SVM and KNN with different kernels, and subspace discriminant of the ensemble method have better and consistent results. Therefore, I have included the results of only

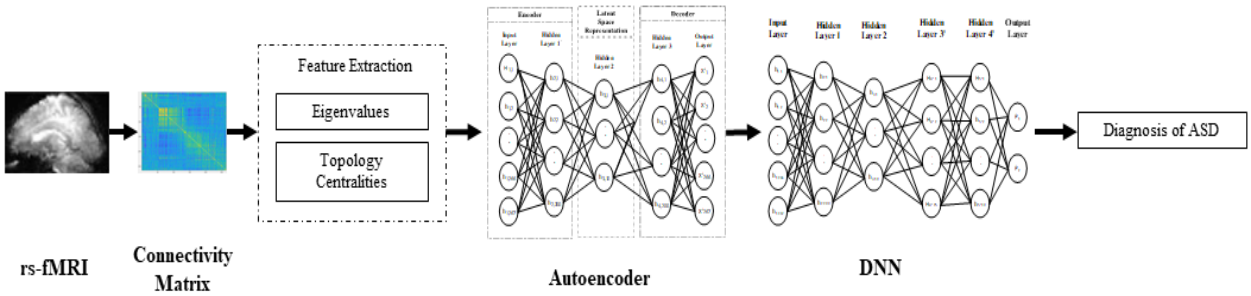


***Figure 4-7**: Illustration of the main steps neural network based classifier using autoencoder*

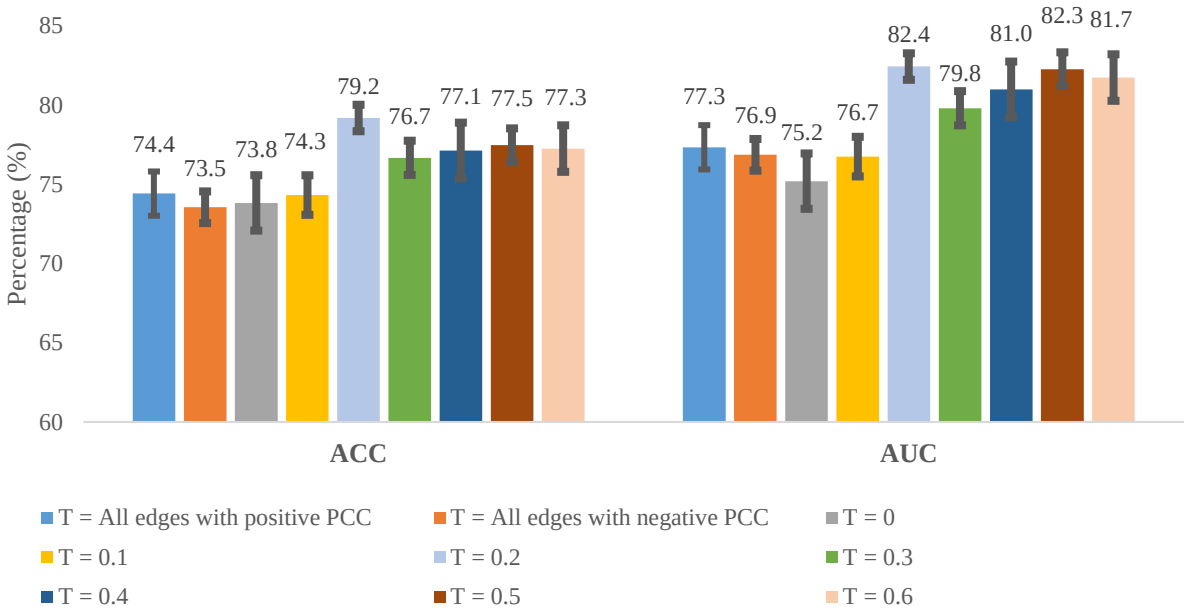


***Figure 4-8**: Comparison of the performance of the autoencoder based classifier for different thresholding values*

machine learning algorithms mentioned above. The illustration of the framework is shown in Figure 4-6. I repeat this process 10 times to ensure the results aren't biased by the data. Also, I carry out the experiments varying the threshold value T from 0 to 0.6 and using all the edges with positive PCC values and all the edges with negative PCC values. I use the ACC and AUC to evaluate the performance and standard deviation to evaluate the stability. The results of the experiments are shown in Table 4-1.

From Table 4-1, it can be seen that higher classification results are achieved when a threshold is applied to the connectivity matrix. Both ACC and AUC are the highest for the threshold value of T = 0.4. However, a significant amount of information is lost when applying a large threshold, which is evident from the results of thresholds T=0.5 and T=0.6. Also, the results are comparatively better when the information of both edges with positive PCC values and edges with negative PCC values are combined.

Apart from using the autoencoder as a feature extractor, I also develop a neural network based classifier combined with an autoencoder. To develop the DNN based classifier at first I train the autoencoder (Figure 4-3) in a similar way as the previous method. After completing the training of the autoencoder, I train the DNN (Figure 4-4). The first two hidden layers of the neural network are pre-trained using the weights and biases of the first two hidden layers of the autoencoder. I use

***Table 4-1**: Performance analysis of the autoencoder based feature selector*

Thresholding Conditions	Linear SVM (%) (stdv)		Medium Gaussian SVM (%) (stdv)		Coarse Gaussian SVM (%) (stdv)		Medium KNN (%) (stdv)		Cosine KNN (%) (stdv)		Weighted KNN (%) (stdv)		Subspace Discriminant (%) (stdv)	
	ACC	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**
All edges with positive PCC	57.2 (2.0)	57.4 (3.7)	62.2 (1.8)	70.2 (2.1)	54.7 (1.9)	57.5 (1.5)	64.9 (3.2)	70.5 (3.0)	64.8 (1.2)	70.8 (2.0)	68.1 (3.1)	74.8 (2.5)	59.9 (1.4)	58.2 (2.0)
All edges with negative PCC	61.2 (2.1)	65.3 (1.9)	64.6 (1.7)	71.3 (1.2)	59.7 (2.7)	62.0 (2.5)	65.4 (1.3)	70.5 (1.1)	65.8 (3.1)	70.4 (2.0)	65.5 (1.7)	73.1 (1.3)	61.3 (2.4)	65.7 (1.8)
T = 0	61.3 (3.6)	66.4 (4.0)	67.5 (3.0)	72.2 (1.6)	57.3 (5.4)	62.2 (5.3)	63.6 (2.7)	68.7 (4.6)	63.3 (2.8)	68.1 (4.3)	65.3 (1.7)	71.0 (3.4)	61.0 (3.0)	64.6 (3.3)
T = 0.1	63.4 (2.1)	67.2 (1.1)	67.4 (2.1)	70.9 (0.9)	59.6 (0.6)	64.7 (0.9)	65.2 (1.6)	71.8 (1.8)	67.2 (2.1)	74.1 (1.0)	65.5 (1.6)	73.4 (1.6)	59.9 (1.5)	65.6 (1.7)
T = 0.2	66.7 (1.7)	71.0 (1.2)	69.3 (2.2)	73.3 (0.7)	67.6 (3.2)	71.6 (3.0)	66.1 (1.8)	72.9 (1.2)	68.5 (2.5)	74.7 (2.9)	66.6 (2.5)	74.3 (1.3)	68.9 (2.6)	72.2 (1.6)
T = 0.3	63.8 (1.1)	69.1 (0.7)	64.6 (2.9)	69.5 (2.6)	61.2 (3.8)	63.8 (3.2)	66.2 (3.0)	72.2 (2.6)	65.2 (2.2)	71.3 (1.8)	65.4 (2.9)	72.5 (3.2)	65.2 (2.9)	67.3 (1.6)
T = 0.4	**68.0 (1.2)**	**71.4 (1.8)**	**72.2 (0.7)**	**78.0 (1.1)**	**66.9 (3.6)**	**70.9 (3.1)**	**72.4 (2.1)**	**77.7 (1.9)**	**72.6 (1.3)**	**79.0 (1.1)**	**72.4 (2.1)**	**77.6 (1.4)**	**69.6 (2.6)**	**72.1 (1.7)**
T = 0.5	58.9 (5.6)	61.3 (8.0)	69.9 (2.1)	73.1 (1.5)	55.5 (3.5)	61.8 (2.8)	65.3 (1.8)	70.9 (1.4)	64.7 (2.2)	69.1 (2.3)	69.3 (1.7)	77.2 (2.0)	60.1 (2.3)	61.9 (4.8)
T = 0.6	54.3 (1.6)	55.1 (2.3)	62.9 (2.3)	62.6 (4.6)	56.0 (2.6)	58.1 (1.1)	64.5 (3.2)	69.3 (1.3)	60.9 (3.9)	63.6 (1.4)	63.7 (1.8)	67.9 (1.7)	54.9 (1.6)	57.7 (1.1)

the 10-fold cross-validation to train and evaluate the performance of the DNN classifier. In this training process, the DNN classifier is trained on 784 subjects, and the performance of the model is evaluated on the testing set of 87 subjects. The classification accuracy of the testing set is considered to be the accuracy of a particular fold. Finally, the average accuracy over all the folds is the accuracy of the model. I have repeated 10 times the process of training the autoencoder and 10-fold cross-validation of the DNN. Each time the subjects are selected randomly. The process is illustrated in Figure 4-7.

The results in Figure 4-8 are the average ACC and AUC achieved after repeating the autoencoder based classification process ten times. From Figure 4-8, it can be seen that the highest classification accuracy is achieved for the threshold value T=0.2. The classification ACC is the least when all edges (both edges with positive PCC values and edges with negative PCC values) are used. Analyzing Figure 4-8 it can be said that rather than using all edges for analysis it is better to remove some edges, as the results are comparatively better for threshold values greater than 0.2. However, removing too many edges could lose a significant amount of information, which is evident from

***Table 4-2**: Performance comparison of the DNN classifier with and without pre-training*

Threshold Values	ACC (%)(stdv)		AUC (%)(stdv)	
	Without Pre-training	**With Pre-training**	**Without Pre-training**	**With Pre-training**
All edges with positive PCC	69.1 (7.4)	74.4 (1.4)	70.9 (10.3)	77.3 (1.3)
All edges with negative PCC	69.5 (0.6)	73.6 (1.1)	72.4 (1.0)	76.9 (0.9)
$\boldsymbol{T = 0}$	69.4 (5.2)	73.8 (1.7)	70.8 (7.0)	75.2 (2.1)
$\boldsymbol{T = 0.1}$	70.3 (3.4)	74.3 (1.3)	72.1 (4.1)	76.7 (0.9)
$\boldsymbol{T = 0.2}$	**76.2 (4.1)**	**79.2 (0.8)**	**79.7 (0.7)**	**82.4 (0.8)**
$\boldsymbol{T = 0.3}$	72.9 (1.0)	76.7 (1.1)	76.6 (1.4)	79.8 (1.2)
$\boldsymbol{T = 0.4}$	74.6 (1.1)	77.1 (1.8)	77.7 (0.9)	80.9 (1.4)
$\boldsymbol{T = 0.5}$	75.5 (1.2)	77.5 (1.1)	80.4 (1.4)	82.2 (0.9)
$\boldsymbol{T = 0.6}$	75.8 (1.7)	77.3 (1.5)	80.7 (1.9)	81.7 (3.4)

the less accuracy for threshold value T=0.6. Also, from the standard deviation bars in Figure 4-8, it can be seen that the results are very stable as the standard deviations are small.

To illustrate the effect of pre-training I also develop a DNN classifier without the pre-training. Apart from pre-training, everything else is kept the same. I repeat the 10-fold cross-validation ten times for the classifier. The comparison of the performance of the DNN with and without the pre-training is shown in Table 4-2.

From Table 4-2 it can be seen that the DNN without pre-training can achieve a classification ACC of only 76.2%. However, the ACC increases to 79.2% when pre-training is applied to the DNN. There is an increase in the classification ACC for every thresholding condition. From the standard deviation (stdv) in Table 4-2, it can be seen that the DNN with pre-training is more stable. However, in both cases, there is an increase of ACC and AUC after applying some thresholds, rather than using all edges in the brain network and the results are maximum for the threshold of $T = 0.2$.

I use the data from both ASD and HC subjects when training the autoencoder. Even though the latent space representation creates a discriminate and salient representation of the input data, the difference in the features between different classes is not satisfactory (Table 4-1). However, when the DNN with pre-training is trained, the weights and biases of the hidden layers are updated to classify between ASD and HC subjects. As a result, the weights and biases of the encoder and latent space representation are also updated to create a more discriminate representation of the data. After completing the training of the DNN, the first two hidden layers can be used to extract

***Table 4-3**: Performance analysis of feature extracted from the combination of the pretrained autoencoder and the DNN*

Thresholding Conditions	Linear SVM (%) (stdv)		Medium Gaussian SVM (%) (stdv)		Coarse Gaussian SVM (%) (stdv)		Medium KNN (%) (stdv)		Cosine KNN (%) (stdv)		Weighted KNN (%) (stdv)		Subspace Discriminant (%) (stdv)	
	ACC	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**
All edges with positive PCC	62.8 (5.5)	66.1 (8.0)	66.3 (4.1)	70.1 (5.5)	55.3 (2.5)	60.8 (4.3)	67.8 (2.5)	71.4 (2.7)	66.1 (2.9)	69.9 (3.1)	67.7 (1.3)	72.4 (1.2)	60.7 (4.3)	63.3 (6.2)
All edges with negative PCC	66.0 (2.1)	71.4 (3.6)	66.6 (1.4)	74.2 (2.1)	63.7 (4.8)	68.1 (4.8)	66.8 (2.6)	72.4 (1.5)	66.7 (3.6)	72.3 (2.9)	67.2 (2.2)	73.8 (2.3)	66.6 (2.7)	70.9 (3.7)
T = 0	67.2 (4.4)	71.8 (4.8)	68.3 (2.1)	71.9 (2.2)	64.4 (6.7)	68.3 (7.0)	67.0 (2.6)	70.5 (3.1)	66.1 (3.4)	70.5 (2.5)	65.6 (2.9)	70.3 (3.0)	64.9 (4.1)	68.9 (5.3)
T = 0.1	67.0 (3.2)	71.2 (2.9)	69.1 (2.0)	74.7 (2.4)	66.6 (3.7)	71.2 (2.8)	68.4 (2.6)	74.5 (2.3)	67.9 (2.8)	76.0 (2.2)	67.9 (3.9)	75.1 (3.0)	67.1 (2.3)	71.1 (2.5)
T = 0.2	**72.8 (2.9)**	**76.3 (3.4)**	**73.7 (1.6)**	**76.7 (1.6)**	**73.0 (2.5)**	**77.9 (1.4)**	**73.2 (1.7)**	**77.7 (2.1)**	**74.6 (1.9)**	**78.7 (2.1)**	**72.7 (2.2)**	**77.2 (2.0)**	**74.4 (1.1)**	**77.8 (1.5)**
T = 0.3	66.1 (3.3)	71.7 (3.3)	66.4 (2.9)	71.9 (2.0)	63.0 (6.1)	67.7 (5.5)	67.1 (2.6)	72.8 (2.6)	66.9 (3.6)	72.1 (2.8)	66.4 (3.3)	72.9 (3.4)	67.8 (1.7)	71.6 (3.8)
T = 0.4	71.1 (4.7)	75.9 (3.8)	73.4 (1.1)	78.8 (1.0)	67.3 (3.1)	73.4 (3.1)	73.2 (2.5)	78.2 (1.8)	73.5 (1.2)	78.8 (1.1)	73.6 (1.5)	78.5 (1.5)	69.8 (3.8)	75.6 (2.8)
T = 0.5	64.4 (7.7)	69.9 (7.0)	70.3 (1.3)	73.1 (1.7)	60.2 (6.9)	64.9 (6.3)	67.7 (3.1)	72.5 (3.2)	67.2 (3.7)	72.8 (3.6)	70.6 (3.6)	76.7 (3.0)	64.0 (7.5)	68.5 (7.8)
T = 0.6	63.8 (3.1)	64.4 (3.2)	65.2 (3.8)	71.2 (3.0)	61.6 (2.8)	59.0 (1.9)	63.7 (1.8)	70.5 (1.5)	64.7 (1.9)	70.0 (2.2)	64.9 (2.6)	71.1 (2.5)	60.4 (3.0)	60.1 (1.9)

a more discriminate representation of features. To enhance the performance of the feature extraction at first, I train the autoencoder. Then, I train the DNN with pre-training using the autoencoder. After completing the training of the DNN, I use the first two hidden layers to extract features and train different machine learning algorithms. Table 4-3 shows the performance of machine learning algorithms. Comparing Table 4-1 and Table 4-3 it can be seen that there is an increase in performance due to pre-training. The highest accuracy of 74.6% is achieved using the KNN classifier with the cosine kernel and the threshold value of T=0.2.

4.4 Summary

In this chapter, I have explained in brief the use of autoencoder for the diagnosis of ASD. I have proposed an autoencoder based classifier, where autoencoder is used to pre-train a DNN classifier. I have shown that the performance of the DNN classifier has increased due to the pre-training. In this chapter, I have also proposed an autoencoder based feature selection method. In this method, I have demonstrated that the learning of the DNN classifier can be incorporated in the autoencoder for dimensionality reduction. I have used traditional machine learning classifiers to evaluate the

performance of the autoencoder based feature selection method. I have achieved a classification ACC of 79.2% using the autoencoder as a classifier and a classification ACC of 74.6% using the autoencoder as feature selector.

CHAPTER 5
DIAGNOSIS OF ASD WITH CONVOLUTIONAL AUTOENCODER AND STRUCTURAL MRI

5.1 Introduction

In the previous chapters, I have discussed different diagnosis processes of ASD using traditional machine learning classifier and DNN. I have used the rs-fMRI images of ASD and HC subjects to create brain networks and analyzed the networks for the diagnosis. The analysis only considered the functional activity of the brain, based on which the features are extracted.

In most of the neuroimage (either structural or functional) based studies related to ASD, different aspects of the brain are analyzed. The aspects are related to the time-series information of functional images [49], [121], deficiency in the white matter [122] and grey matter [63] of the structural images. In the structural MRI image based study, the researchers have used voxel-based morphology (VBM), surface-based morphology (SBM), tensor-based morphology (TBM), volume-based analysis, ROI based morphology, etc. According to [123], ROI based studies have reported increased total brain volume of young children, VBM based studies have revealed increased grey matter volume but decreased grey matter density, SBM based studies have shown increased cortical thickness for the ASD subjects compared to the HC subjects. So, there is dissimilarity in the structural images of an ASD subject and HC subject. However, the studies conducted so far are based on the different extracted properties of the brain. They haven't used the raw image for the analysis.

In this chapter, I will look into the raw structural MRI images (T1-weighted) for the diagnosis of ASD. For the analysis, I have used a convolutional autoencoder (CAE) based classifier. A CAE is a type of deep neural network where the convolutional neural network is used in the autoencoder. The main difference between an autoencoder and CAE is that the CAE uses convolutional layers rather than the fully connected layers. In this study, I have trained the CAE on the raw MRI images. After completing the training, I have used the autoencoder to reconstruct the images. Then I have measured the similarity of the input images and reconstructed images. To measure the similarity between images, I have used structural similarity index measure (SSIM), peak signal to noise ratio

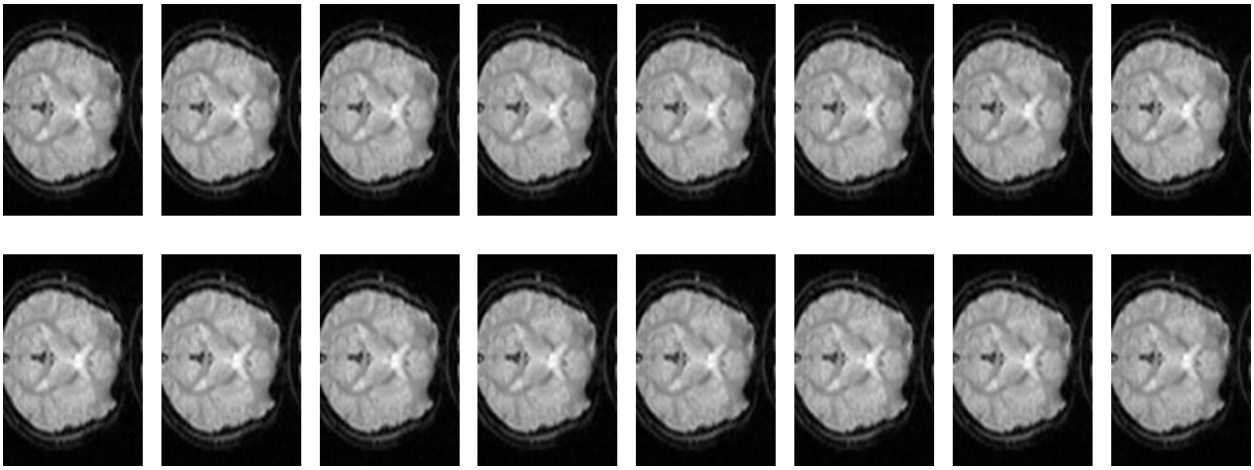


Figure 5-1*: rs-fMRI images of a subject for different time point*

(PSNR), and mean squared error (MSE). Using the similarity indices as features, I have trained different machine learning classifiers for the diagnosis of ASD.

In this chapter at first, I will discuss the data I have used along with the pre-processing steps. Then I will describe in detail the CAE. In this part, after introducing CAE, I will describe the model I have used in this study. Finally, I will compare the results of this study with the previous studies of my thesis and also state-of-the-art methods.

5.2 Materials and Methods

5.2.1 Data Preprocessing

In ABIDE 1 dataset, both rs-fMRI and T1-weighted MRI images are available for every subject. The details about the scanners and scanning parameters to acquire the structural images are given in Table 2-1. By analyzing the phenotypic information of the subjects (Table 2-2), it can be seen that this study covers a minimum age of 6.5 years to a maximum of 64 years. Working with this wide range of ages will help to combat the problem of the effect of age on ASD subjects.

In the functional images, the activity of the brain is measured through the change in the blood flow. The functional images do not contain any structural information. Only the functional activity is captured for different time points. So, the analyses are based on the time-series measurement acquired from different ROIs. Figure 5-1 shows some examples of the rs-fMRI images of a subject captured at different time points for a particular slice. It is very hard to predict the structure of the brain for the images in Figure 5-1.

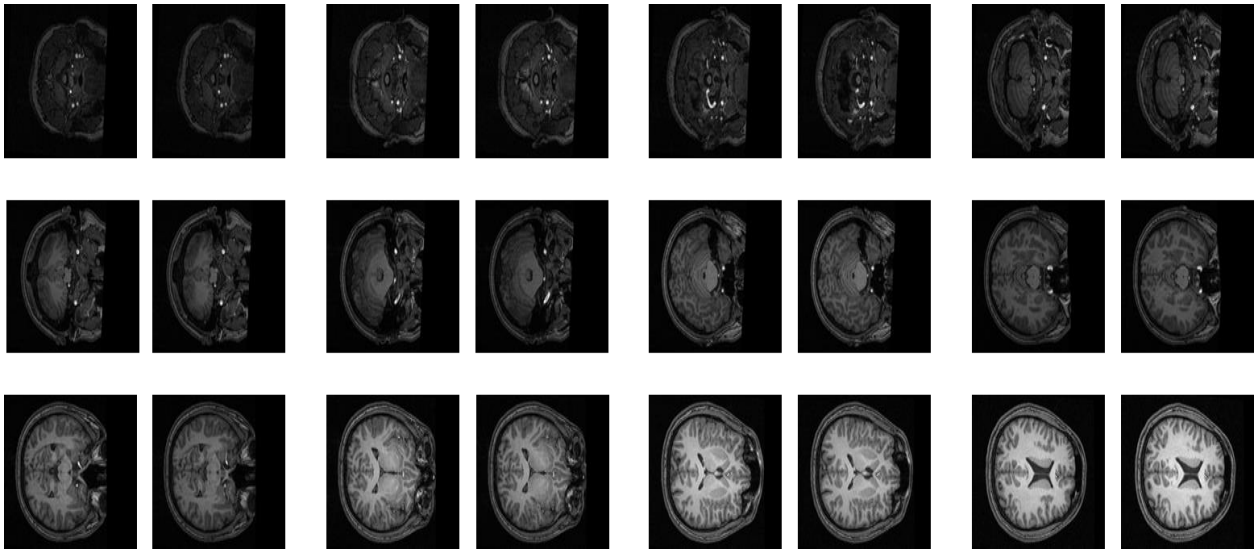


***Figure 5-2**: Example of two consecutive slices of T1-weighted MRI images of a subject which are 10 slices apart*

The structural (T1-weighted) images are more concentrated to capture the structural integrity of the brain. The T1-weighted MRI images are three dimensional (3D) in nature. The image of the whole brain is captured into slices. Stacking all the images together can generate a 3D view of the brain. The resolution of the T1-weighted images of every subject in ABIDE 1 is 176×265, and for every subject, there are 256 slices of images. One way to train the CAE is to feed it with all 256 slices of images of every subject. From Figure 5-2, it can be seen that there are some similarities in the images for consecutive slices, but moving away from a particular slice the similarities are very few. So, in this study out of the 256 slices of images for every subject, I have chosen the

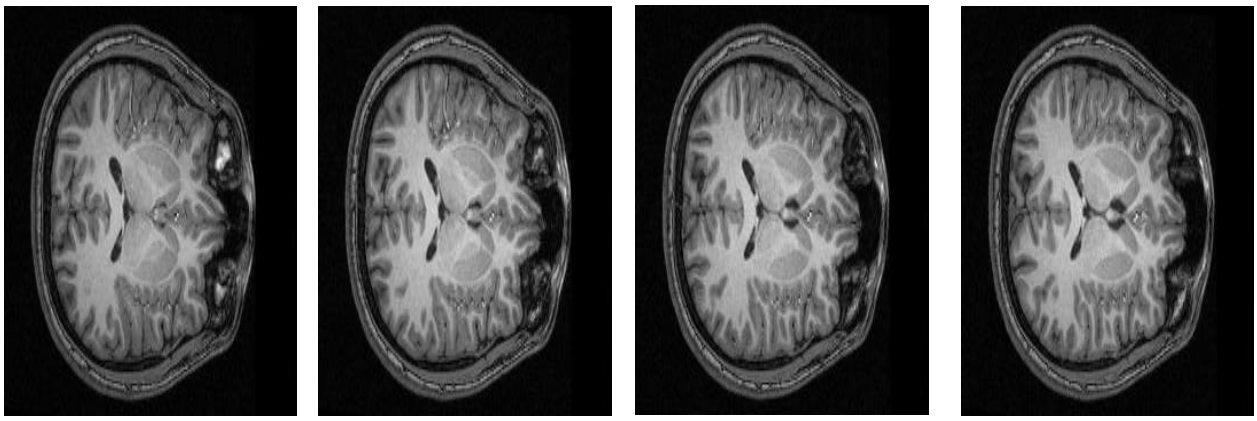


***Figure 5.3:** Example of the T1-weighted images used in the study*

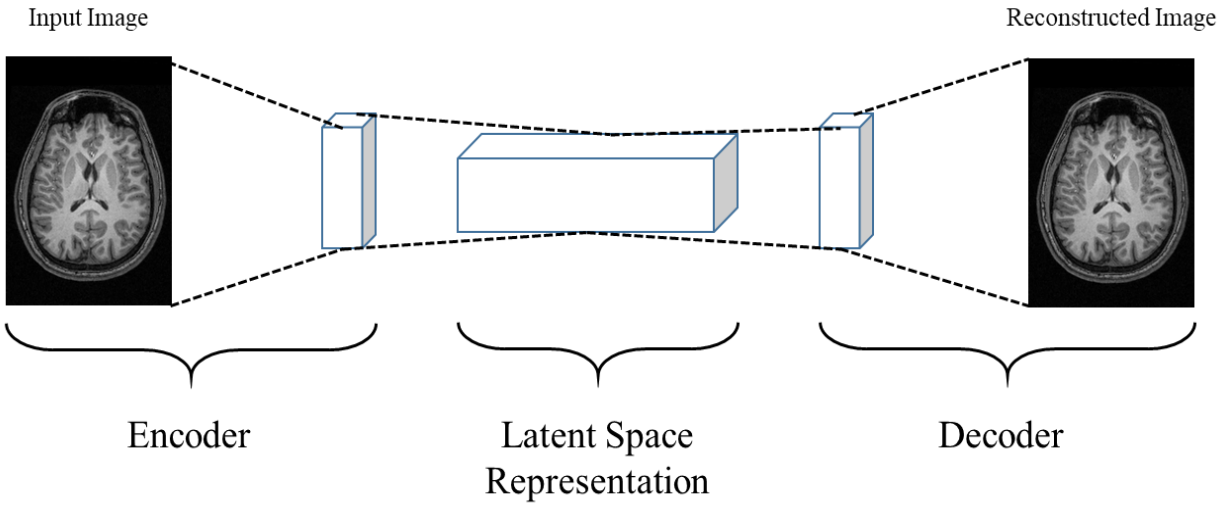


Figure 5-4: *Block diagram of a simple CAE*

images of slices 138 to 141. The similar slices of images are selected for every subject. Figure 5-3 shows the example of the slices of images for a particular subject. I have used the raw images for analysis, and no other pre-processing is applied to the images.

5.2.2 Convolutional Autoencoder

An autoencoder is a type of neural network where the input is reconstructed in the output. This is done by creating a latent space representation by compressing the input data. Whereas a convolutional neural network is a type of deep learning algorithm where the input to the model are images. The model then assigns importance to various objects or aspects in the image, based on which further analysis (such as classification, feature extraction) is done. So, in the CAE input images are reconstructed in the output, and the model is built using convolutional layers. Using convolutional layers rather than fully connected layers allow the model to extract visual features of the images and obtain more accurate latent space representation [124].

Figure 5-4 shows a simplified design diagram of the CAE. The basic building blocks (encoder, latent space representation, and decoder) are the same as the normal autoencoder. However, the input and output are images in the CAE. The main limitation of the normal autoencoder is that it cannot capture the local connectivity in the image [125]. As a result, the normal autoencoders removes local information of the images. However, as this study is focused on analyzing the differences in the structures in the brain images between ASD subjects and HC subjects, removing local structures will have a loss of information. So, to resolve this issue convolution autoencoders

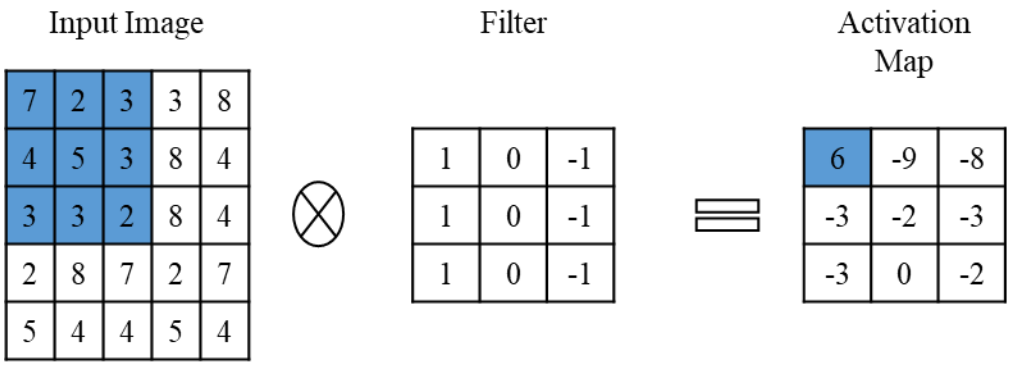


Figure 5-5: *A graphical example of the convolution process*

are used. The main advantage of the convolution autoencoders is that spatial locality is preserved by sharing the weights among all locations in the input, similar to a CNN [126]. For images with r rows and c columns the reconstruction error $L(x, x')$ of the CAE is calculated for every pixel in the image as follows

$$L(x, x') = \sum_{i=1}^{r} \sum_{j=1}^{c} \left\| x_i - x_i' \right\|^2 \qquad (5.1)$$

where x and x' be the input image and the reconstructed image, respectively, and i, j be the position of any pixel in the image.

As in the CNN, CAE also uses convolutional layers, pooling layers, and upsampling layers. The description of the different layers is given in the subsequent sections.

5.2.2.1. Convolutional Layer

A convolutional layer is the main building block of a CNN. It contains a set of filters (or kernel) whose parameters are to be learned by the model throughout the training. The size of the filters is

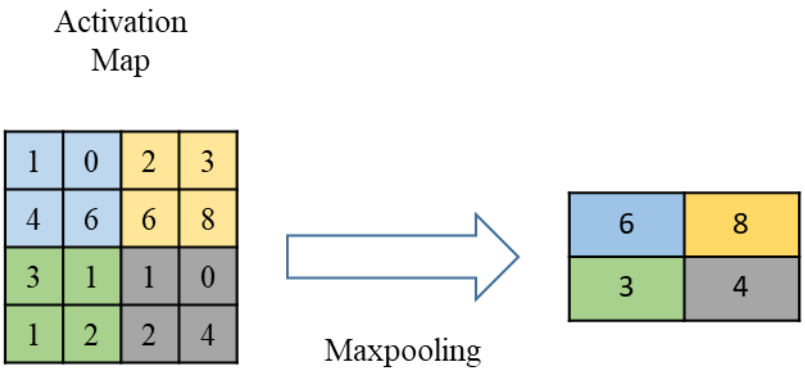


Figure 5-6: *An example of the maxpooling operation using a 2×2 filter*

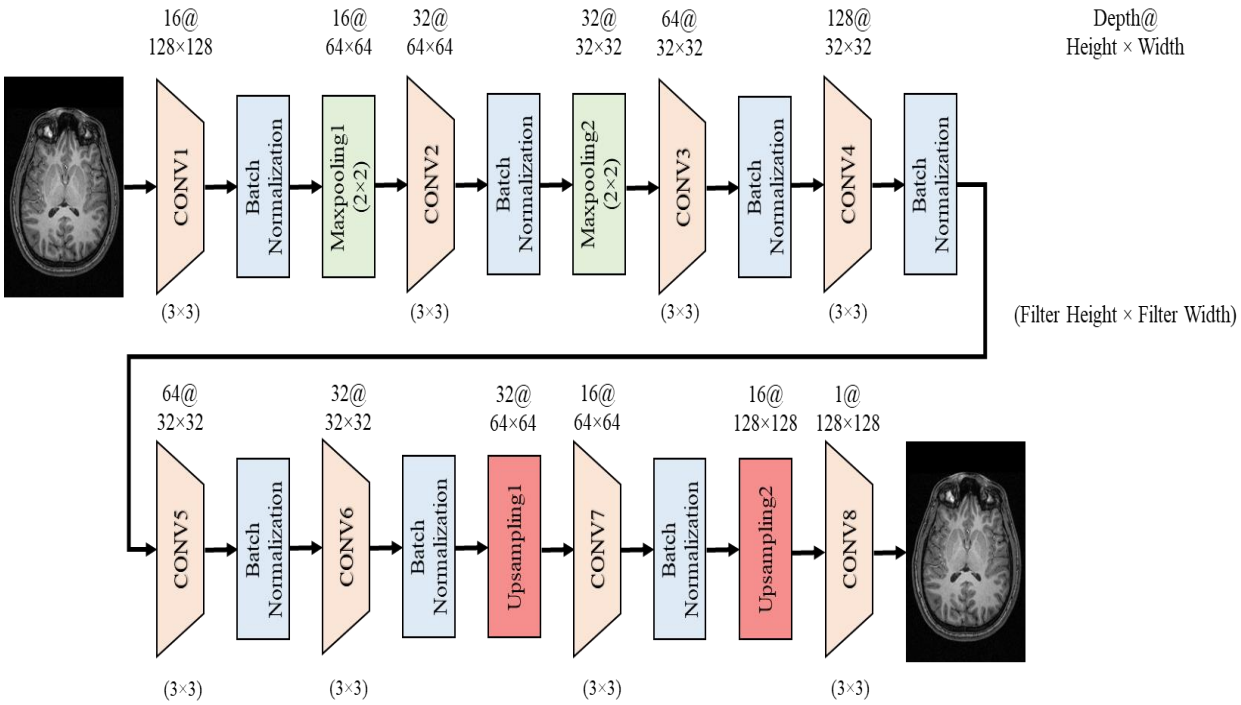


***Figure 5-7**: Architectural details of the proposed CAE*

usually smaller than the actual image. Each filter convolves with the image and creates an activation map. For convolution, the filter slid across the height and width of the image and the dot product between every element of the filter, and the input is calculated at every spatial position. Figure 5-5 shows an example of the convolution process. The first entry of the activation map (marked blue in Figure 5-5) is calculated by convolving the filter with the portion marked blue in the input image. The activation map is generated by repeating this process for every element of the input image. The output volume of the convolutional layer is generated by stacking the activation maps of every filter along the depth dimension. Every component of the activation map can be thought to be the output of a neuron. So, each neuron is connected to a small-local region in the input image, and the size of the area equals the size of the filter. All the neurons in an activation map also share parameters with each other. Because of the local connectivity of the convolutional layer, the network is forced to learn filters which have the maximum response to a local region of the input [127]. The initial convolutional layers capture the low-level features (e.g., lines) of the image, while the later layers extract the high-level features (e.g., shapes, specific objects) [128].

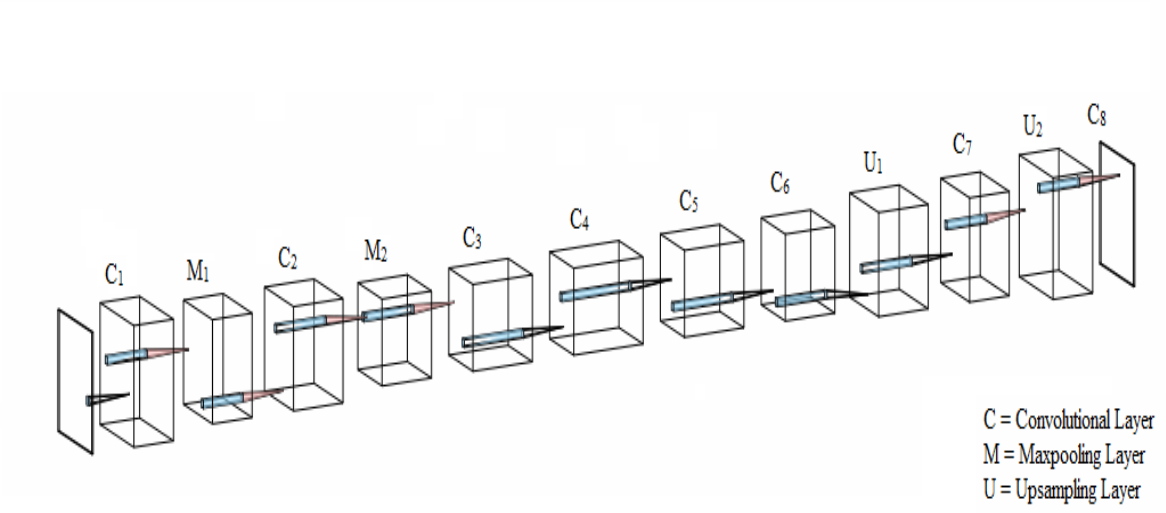


Figure 5-8: *The proposed CAE architecture schematic*

5.2.2.2. Pooling Layer

The convolutional layers effectively outline the features of the input in the activation map by applying learned filters. However, the precise location of a feature is not important. Because, if the position of the feature is changed in the input, the activation map will also change. It will make

Table 5-1: *Description of each layer of the proposed CAE*

	Layer Name	Layer Type	Activation	Filter Dimension	Dimension (Depth×Height×Width)
Encoder	CONV1	Convolutional	Relu	3 × 3	16 × 128 × 128
	MAXPOOLING1	Maxpooling	-	2 × 2	16 × 64 × 64
	CONV2	Convolutional	Relu	3 × 3	32 × 64 × 64
	MAXPOOLING2	Maxpooling	-	2 × 2	32 × 32 × 32
	CONV3	Convolutional	Relu	3 × 3	64 × 32 × 32
	CONV4	Convolutional	Relu	3 × 3	128 × 32 × 32
Decoder	CONV5	Convolutional	Relu	3 × 3	64 × 32 × 32
	CONV6	Convolutional	Relu	3 × 3	32 × 32 × 32
	Upsampling1	Upsampling	-	2 × 2	32 × 64 × 64
	CONV7	Convolutional	Relu	3 × 3	16 × 64 × 64
	Upsampling2	Upsampling	-	2 × 2	16 × 128 × 128
	CONV7 and Output	Convolutional	Relu	3 × 3	1 × 128 × 128

the model ineffective. To overcome this problem, pooling layer is used. The pooling layer reduces the spatial size of the representation. It also reduces the parameters and hence, the amount of computation in the network. Using the pooling layer makes the model translation invariant. The pooling layers operate on each activation map separately. For pooling, the input is divided into non-overlapping rectangular sub-regions, which equals the size of a filter, and information is extracted from every sub-region. In this study, I have used the maxpooling layer. Figure 5-6 shows the example of maxpooling using a 2×2 filter. The filter is applied to the activation map in a non-overlapping manner, and the maximum value in each sub-region is the output of the filter. The output of the maxpooling will be the new activation map on which further operations will be conducted [129].

5.2.2.3. Upsampling Layer

A compressed representation of the input is created in the latent space representation of the autoencoder. The dimension of the input is reduced in the encoder part using the maxpooling layers. However, in the decoder, it is necessary to increase the dimensions again to recreate the input in the output. So, the decoder needs an inverse of the maxpooling layer. I have used the upsampling layer in my model to increase the dimension of the input. It simply repeats the rows and columns provided as input in the output to increase the dimension. Similar to the maxpooling, the upsampling is applied to individual activation maps.

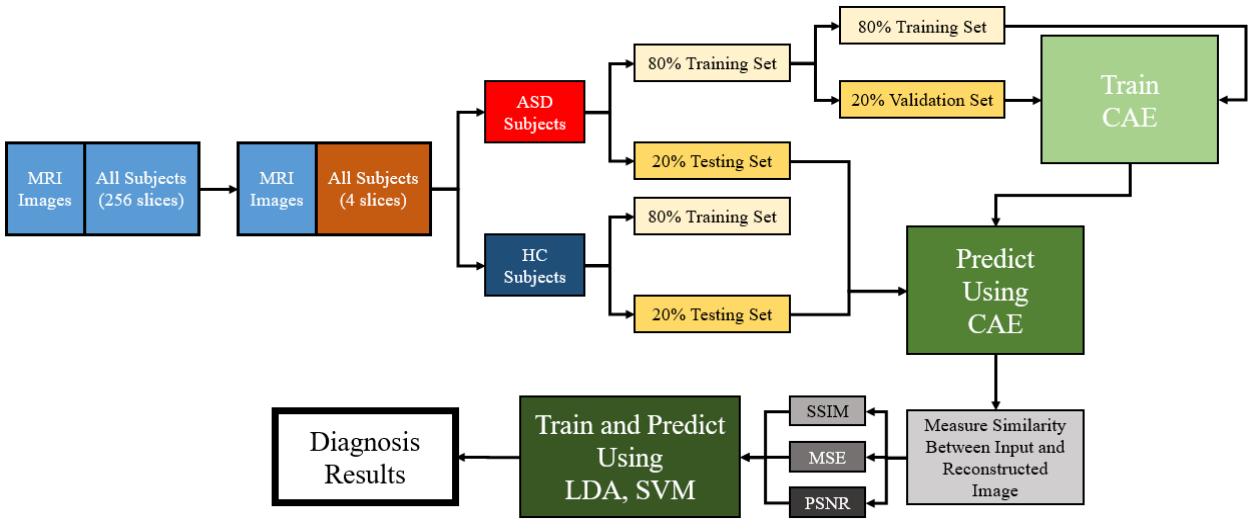


Figure 5-9: *Block representation of the experimental setup of CAE based ASD diagnosis*

Table 5-2: *Performance of the CAE based ASD diagnosis for different training epoch*

Epochs	CAE Trained on ASD Subjects				CAE Trained on HC Subjects			
	ACC (%) (STDV)		AUC (%) (STDV)		ACC (%) (STDV)		AUC (%) (STDV)	
	SVM	**LDA**	**SVM**	**LDA**	**SVM**	**LDA**	**SVM**	**LDA**
10	76.1 (14.0)	79.1 (13.3)	75.6 (13.5)	78.6 (13.0)	80.1 (13.2)	83.6 (15.7)	80.8 (13.2)	83.8 (15.6)
20	79.6 (15.3)	82.1 (14.9)	79.6 (14.4)	82.2 (14.7)	82.9 (19.2)	84.8 (14.3)	83.2 (19.2)	84.7 (14.3)
30	85.4 (10.3)	90.1 (9.3)	85.6 (10.9)	90.5 (9.2)	90.2 (13.1)	92.8 (8.3)	90.2 (12.9)	92.8 (8.3)
40	83.3 (16.0)	88.6 (15.0)	83.4 (15.6)	88.4 (15.1)	85.4 (20.5)	93.1 (14.6)	85.2 (20.7)	93.2 (14.5)
50	84.1 (17.5)	88.8 (16.5)	84.7 (17.3)	88.3 (16.5)	83.9 (15.3)	87.6 (10.7)	83.3 (15.3)	87.5 (10.6)
60	**88.6 (13.5)**	85.3 (17.8)	**88.5 (13.6)**	85.2 (17.7)	79.5 (21.8)	87.0 (16.5)	78.0 (23.9)	86.5 (17.3)
70	76.9 (18.4)	82.1 (24.2)	75.7 (19.4)	82.2 (24.9)	78.4 (15.8)	82.6 (15.1)	78.5 (16.2)	82.7 (15.5)
80	84.8 (17.7)	**90.9 (13.5)**	84.8 (17.7)	**90.8 (13.6)**	**93.2 (12.7)**	**96.6 (8.3)**	**93.1 (12.9)**	**96.5 (8.3)**
90	86.0 (14.8)	90.0 (11.6)	86.0 (14.7)	89.5 (12.3)	86.6 (17.3)	94.0 (6.9)	86.7 (17.4)	93.5 (7.3)
100	85.1 (18.3)	90.4 (13.6)	85.4 (18.1)	90.2 (13.9)	87.1 (18.8)	90.0 (14.3)	88.2 (17.0)	89.8 (14.7)
110	77.2 (24.3)	87.9 (13.3)	77.8 (23.7)	87.8 (13.3)	79.1 (20.7)	89.4 (14.9)	78.2 (20.8)	89.2 (15.0)
120	83.9 (19.1)	88.0 (13.3)	83.8 (18.5)	87.8 (13.2)	85.9 (19.1)	94.9 (12.8)	85.8 (19.3)	94.9 (12.5)
130	83.7 (18.0)	90.6 (13.0)	83.6 (18.6)	90.9 (13.0)	87.1 (17.7)	90.6 (14.0)	86.9 (17.9)	90.4 (14.1)
140	78.9 (20.0)	85.1 (19.4)	78.2 (20.7)	84.9 (19.7)	85.1 (19.7)	92.3 (9.6)	85.0 (19.5)	92.6 (9.4)

5.2.3 Proposed Convolutional Autoencoder

The architecture with details of the proposed CAE is shown in Figure 5-7, and the schematic of the architecture is shown in Figure 5-8. I have experimented with different combinations of layers and hyperparameters. However, the proposed model is chosen as the results are stable and the performance is also better. All images are resized to a dimension of 128×128 before applying

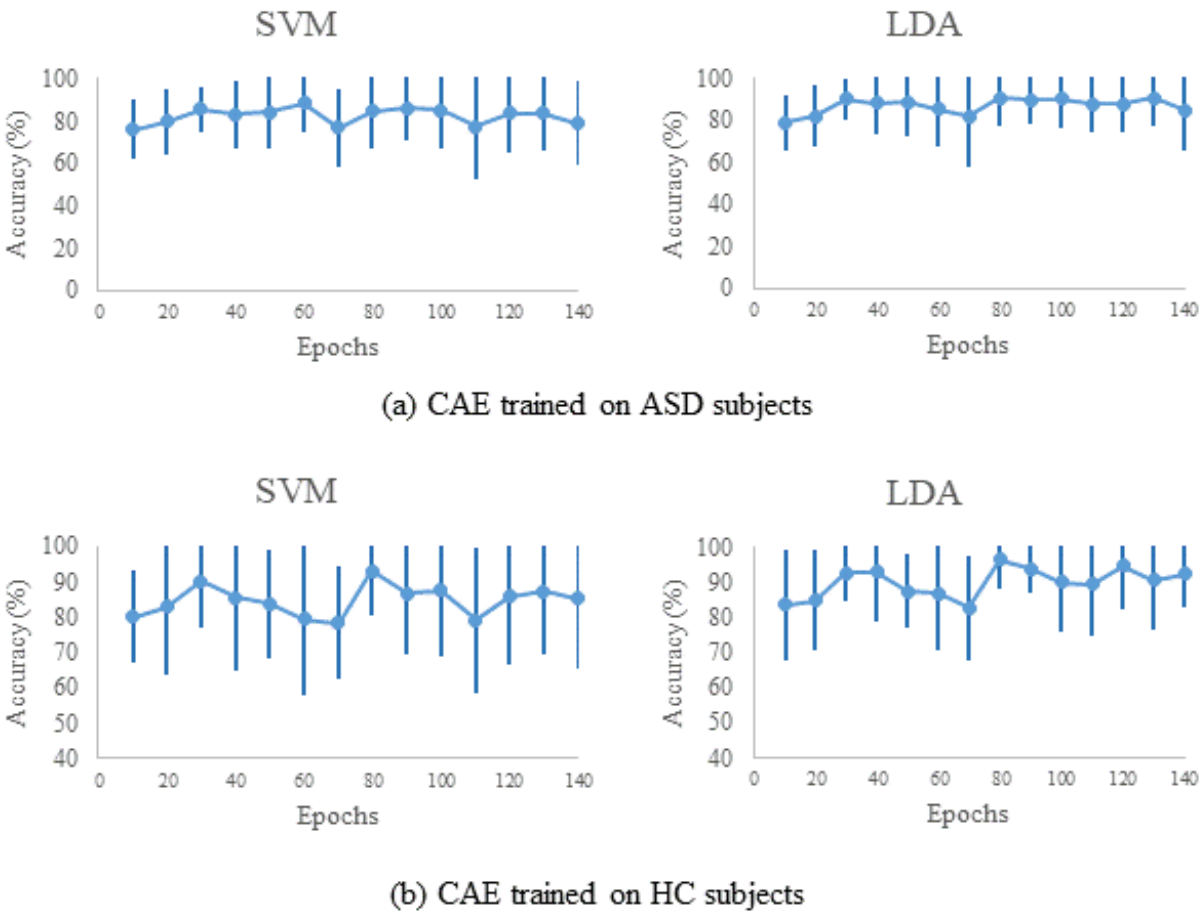


Figure 5-10: *Accuracy curve of the CAE based ASD diagnosis for training the CAE with varying epochs, (a) CAE trained on ASD subjects, (b) CAE trained on HC subjects*

them to the CAE. I have resized the images to decrease the training time. I have also performed the analysis with the actual size of the images. There isn't much difference in the performance due to the resizing. In the proposed CAE each convolutional layer is followed by a batch normalization layer, except the last convolutional layer as it is used to generate the output image. The batch normalization layer ensures all the data are in the same range throughout the network. It also reduces covariance shift (change in the distribution of the input variables) of the hidden units. In the encoder part of the CAE, there are total 4 convolutional layers. The first convolutional layer has 16 filters of size 3×3, followed by a maxpooling layer with a filter size of 2×2. The maxpooling layer downsamples the input by a factor of 2. The second convolutional layer has 32 filters of size 3×3, again followed by a maxpooling layer with a filter size of 2×2. The filter size of the third and fourth convolutional layers is 3×3 with 64 and 128 filters, respectively. However, there is no maxpooling layer after the third and fourth convolutional layers.

The decoder of the CAE reconstructs the input. There is a total of 4 convolutional layers in the decoder. In the first three convolutional layers, the filter size is 3×3, and there are 128, 64, and 32 filters, respectively. After the second and third convolutional layers, there are upsampling layers. The upsampling layers increase the dimension of the input by a factor of two. There is only one filter of size 3×3 in the final convolutional layer to reconstruct the input having a single channel. Table 5-1 summarizes the details of each layer of the CAE.

5.3 Experimental Setup

Out of 256 slices of T1-weighted MRI images from 871 subjects of the ABIDE 1 dataset, I have selected four slices (138 to 141) for each subject. There isn't enough information in the first and last few slices of images. So, those are discarded. The image of the whole brain is only clear in slices 125 to 184. However different regions of the brain are more clearly visible in slices 135 to 156. I have experimented with different numbers of slices and the results are better for slices 138 to 141. After selecting the slices of MRI images, I have trained the proposed CAE. However, when training the autoencoder, rather than using the images of both types of subjects, I have used the images of only one type of subject (for example, HC). This is because, when training the CAE with the images of HC subjects, it is learning to recreate different structures and shapes in the brain images of the HC subject. Now, compared to the images of HC subjects if there is any variation in the structure or shape of the ASD subjects, the CAE will not be able to recreate that variation in the reconstructed image. Due to this, the reconstructed images of the ASD subjects will have more dissimilarity than the reconstructed images of the HC subjects using the same model.

The diagnosis process used in this study is illustrated in Figure 5-9, where the autoencoder is trained using the images of HC subjects. At first, I have randomly divided the images of HC subjects into 80% training set and 20% testing set. Then I have trained the CAE using the data in the training set. After completing the training, I have used the trained CAE model to reconstruct the images in the testing set. I have also randomly selected 20% data from the images of ASD subjects and used the trained CAE model to reconstruct images. So, for every input image, I have a reconstructed image. Next, I have measured the similarity between the reconstructed image and the input image using SSIM, MSE, and PSNR. Then I have used SSIM, MSE, and PSNR as features, and trained LDA and SVM. I have used 10-fold cross-validation for LDA and SVM. The similarity indices (SSIM, MSE, and PSNR) are described in the subsequent sections.

5.3.1 Structural Similarity

SSIM is a measurement of the similarity of the structures in two images. To measure the SSIM the image is divided into different windows of the same shape. The SSIM between two image windows A and B are measured as follows

$$SSIM(A,B) = \frac{(2\mu_A\mu_B + c_1)(2\sigma_{AB} + c_2)}{(\mu_A^2 + \mu_B^2 + c_1)(\sigma_A^2 + \sigma_B^2 + c_2)} \qquad (5.2)$$

where μ_A is the average of window A, μ_B is the average of window B, σ_A^2 is the variance of window A, σ_B^2 is the variance of window B, σ_{AB} is the covariance of two windows A and B, $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$, L is the ratio of the largest and smallest pixel values of the image, $k_1 = 0.01$, and $k_2 = 0.03$. The SSIM of an image is the average of SSIM of all windows.

SSIM can capture the difference in the local luminance and structure. So, if there are any structural dissimilarities, the SSIM will be able to detect it.

5.3.2 Mean Squared Error

The MSE between the input image x and reconstructed image x' with r rows and c columns is measured as follows

$$MSE(x,x') = \frac{1}{rc}\sum_{i=1}^{r}\sum_{j=1}^{c}\left(x(i,j) - x'(i,j)\right)^2 \qquad (5.3)$$

where $i,\ j$ is the position of any pixel in the image.

5.3.3 Peak Signal to Noise Ratio

The PSNR between the input image x and reconstructed image x' is measured as follows

$$PSNR(x,x') = 10\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \qquad (5.3)$$

where MAX_I is the largest possible intensity in the image data type, MSE is the mean squared error between the images.

5.4 Results and Discussion

To evaluate the performance of the CAE based ASD diagnosis process, I have trained the CAE with a single subject type. Then, I have used the trained CAE to reconstruct the images of both

classes. Figure 5-10 shows the accuracy curve of diagnosis process for training the CAE for different epochs. A detailed view of the experiments is in Table 5-2. The results in Table 5-2 and Figure 5-10 are the averages of repeating the experiments ten times. I have collected the results by training the CAE models separately for every mentioned epoch in Table 5-2. Also, all the results are for the validation sets. The validation set is used only once for measuring the performance of the model. It helps to avoid overfitting as much as possible. I have repeated the experiments for the training of the CAE using ASD subjects and HC subjects.

From Table 5-2 and Figure 5-10 it can be seen that the performance is better when the CAE is trained with images of HC subjects. ASD covers a wide range of characteristics. So, due to different behavioral abnormalities, the changes in the structure of the brain might be different. As a result, it is hard to establish a baseline for the ASD. However, for HC subjects the similarities in

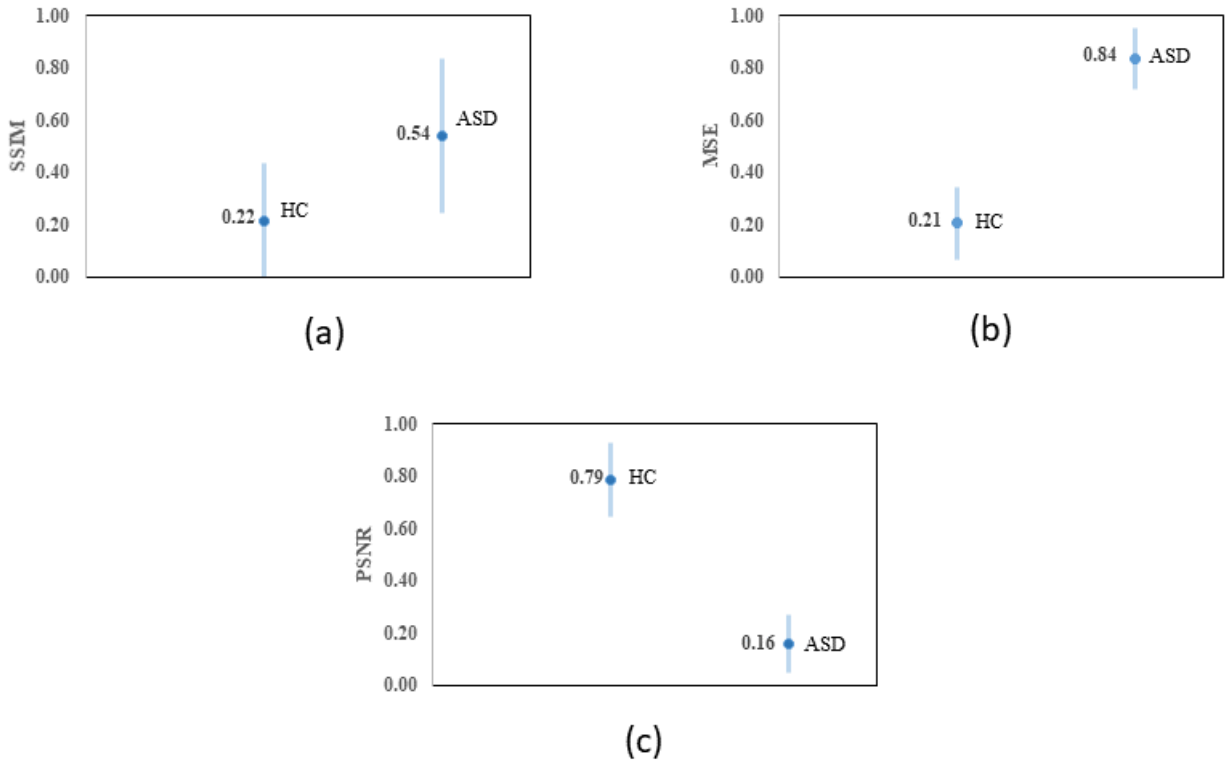


Figure 5-11: *Similarity indices for the reconstructed images of the CAE, (a) SSIM, (b) MSE, (c) PSNR*

Table 5-3: *Comparison of accuracy and AUC for varying the number of slices*

Slice Numbers	ACC (%) (STDV)		AUC (%) (STDV)	
	SVM	LDA	SVM	LDA
138 – 141	**83.9 (15.3)**	**87.6 (10.7)**	**83.3 (15.3)**	**87.5 (10.6)**
138 – 142	68.3 (11.0)	67.3 (5.4)	67.9 (11.1)	66.6 (4.8)
138 – 143	78.6 (10.1)	80.2 (11.4)	76.4 (13.0)	80.6 (11.7)
138 – 144	62.3 (13.8)	71.4 (6.1)	61.9 (13.6)	71.2 (5.9)
138 - 145	71.2 (10.5)	74.3 (5.2)	70.3 (10.0)	73.8 (5.2)

the regions of the brain are high. The results also support the statements, as a higher classification ACC of 96.6% and AUC of 96.5% is achieved when the CAE is trained with images of HC subjects and LDA is used as the classifier. The maximum classification ACC of 90.9% and AUC of 90.8% is achieved when the CAE is trained with images of ASD subjects and LDA is used as the classifier. Analyzing the standard deviation it can be seen that the results are more stable when the CAE is trained with images from HC subjects.

Figure 5-11 shows the plot of the average of different similarity indices along with the standard deviation for training the CAE with HC subjects. Analyzing the similarity indices from Figure 5-11, it can be said the CAE creates a better reconstruction of the images from the HC subjects compared to the images from the ASD subjects. The low MSE and high PSNR indicates there is minimal difference between the input image and the reconstructed image. The range of both MSE and PSNR for HC subjects and ASD subjects are also apart from each other, which indicates the effectiveness of the features. There is some overlap between the SSIM of ASD and HC subjects. However, when the features are combined together, the classifier produces better classification results.

I have also experimented by varying the slices of T1-weighted MRI images. The results are in Table 5-3. The best classification results are acquired when slices 138 to 141 are used. Increasing the number of slices, the accuracy of the diagnosis process using both the classifiers have decreased. As mentioned, the images are similar around a particular slice. Moving away from the slice makes the images more dissimilar. As a result, the structure of different regions in the brain also changes. So, it becomes difficult for the convolutional layers to extract discriminating features

Table 5-4: *Comparison of the accuracy of the proposed studies and state of the art classification methods*

Methods	ACC (%)
Dvornek et al. [11]	70.1
Wong et al. [15]	71.1
Heinsfeld et al. [49]	70.0
Khosla et al. [51]	73.3
Parisot et al. [57]	69.5
Abraham et al. [59]	66.8
Xing et al. [132]	66.8
Traditional machine learning based classifier (**Chapter 3**)	78.4
Autoencoder based DNN classifier (**Chapter 4**)	79.2
Autoencoder based feature selector (**Chapter 4**)	74.6
CAE based classifier (**Chapter 5**)	96.6

from the images. For the experiments in Table 5-3, the CAE is trained for 50 epochs on the HC subjects.

One of the major problems of the ASD studies is the inconsistency of the dataset. There are some studies where a better classification result is acquired for a smaller dataset. The study in [130] has reported a classification accuracy of 85.0%. However, the study only included 74 ASD subjects and 107 HC subjects. Another study [131] has experimented with 24 ASD subjects and 26 HC subjects and reported an accuracy of 96.4%. Due to the small size of the data, methods are not generalizable across datasets [15], as it doesn't cover different age ranges, scanning parameters, and other variables that can introduce an inconsistency in the dataset. So, to make the studies in this thesis comparable, I have used the same 871 subjects mentioned in [15]. This study has the highest accuracy using the ABIDE 1 dataset.

Table 5.4 shows a comparison of the proposed studies and state-of-the-art studies using the ABIDE 1 dataset. For the study of the rs-fMRI based ASD diagnosis, the maximum accuracy of 79.2% is achieved for the autoencoder based DNN classifier, where the DNN classifier is pre-trained using the autoencoder. However, using the T1-weighted MRI images in a CAE a classification accuracy

of 96.6% is acquired. To the best of my knowledge, this is the highest accuracy achieved using the ABIDE 1 dataset.

5.5 Summary

In this chapter, I have looked into the T1-weighted MRI images for the diagnosis of ASD. Rather than extracting features from the structural MRI images, a better way is to use the images directly into the CAE. I have achieved an accuracy of 96.6% using the proposed method. The effect of age for the diagnosis of ASD is important, and the robustness of the proposed method is proved by experimenting with the ABIDE 1 dataset, which covers patients with an age range of 6.5-64 years. The process of training the CAE with a particular subject type and then using it for binary classification is a new approach, and it has proved to be efficient in the experiments. Also, instead of using the whole set of slices of MRI images, it is more convenient to focus on a particular area in the brain and use the relative slices for the analysis.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this thesis, I looked into different machine learning approaches for the diagnosis of ASD using the MRI data. I started by building brain networks from the rs-fMRI image data by dividing the whole brain into 264 ROIs. Then I proposed the spectrum of the Laplacian matrix of the brain networks as a feature for the machine learning classifiers. The classification results achieved for different traditional machine learning classifiers using the proposed feature and the topological centralities are better than the state of the art methods. Then, I looked into DNN classifiers for the diagnosis of ASD. I used the same features (spectrum and topological centralities) for the DNN classifier. However, I have shown that the results of the DNN classifier can be further improved by pre-training the DNN classifier with an autoencoder. I also proposed an autoencoder based feature extractor where I used the latent space of the autoencoder to create a discriminate and compressed representation of the features. Finally, I looked into the T1-weighted MRI images and proposed a CAE based diagnosis process using the T1-weighted images. Rather than extracting features from the images, I used raw images for the diagnosis.

In Chapter 2, I introduced the ABIDE 1 database. I showed why this is the most complicated dataset to work with for the study of ASD. Then, I described the pre-processing steps of the rs-fMRI image along with the software used. After processing the rs-fMRI images, I defined 264 ROIs and extracted the time-series information of the ROIs. I created connectivity matrices by calculating the PCC of the time series measurement between every pair of the 264 ROIs. The ROIs were the nodes and the PCCs were the edges of the brain network. Finally, I described how to calculate the adjacency matrix, the degree matrix, and the Laplacian matrix from the connectivity matrix. I used these matrices to extract features from the brain networks.

In Chapter 3, I proposed the use of the spectrum of the Laplacian matrix of the brain network as a feature for the machine learning classifiers. I also described the topological centralities that I used in the studies. To overcome the problem of overfitting and finding the discriminate features from the available feature set, I used a feature selection algorithm. The feature selection algorithm was explained in this chapter. Using the selected features I trained different traditional machine

learning classifiers. I also discussed the effect of thresholding the brain network and compared the results for different thresholding conditions. To illustrate the efficiency of the proposed feature and feature selection algorithm, I trained the traditional machine learning classifiers on the individual sites of the ABIDE 1 dataset. The classification results for both multi-site and single-site experiments are better than the state-of-the-art studies.

I studied the DNN models in Chapter 4. I built a DNN classifier to diagnose ASD. To improve the classification results of the DNN classifier, I developed an autoencoder. The autoencoder was used to pre-train the classifier. I experimented using the same features defined previously and the same thresholding conditions. I showed that the classification results improved for all the thresholding conditions due to the pre-training. An autoencoder based feature selector was also proposed in this chapter. I used the latent space of the autoencoder to represent the initial 267 features using only 10 features. I pre-trained the feature selector using the DNN classifier and showed that it increases the classification results for most of the traditional machine learning classifiers.

In Chapter 5, I studied the CAE for the diagnosis of ASD using the raw T1-weighted MRI images. I explained the choice of slices of MRI images and described the CAE model in brief. I trained the CAE using the images of a single type of subject (ASD or HC). Then, I used the CAE to reconstruct the images of both types of subjects. I used different similarity indices to measure the similarity between the actual image and the reconstructed image. Then, I used the similarity indices as features to train SVM and LDA for the diagnosis of ASD. Training the CAE on a single class of data allowed it to learn different features of that class. So, when this trained CAE was used to reconstruct the image of a different class, it failed to generate a perfect reconstruction due to the absence of the features it has learned. As a result, the similarity between the input image and the reconstructed image was less for the different classes. The better classification results of this study proved that there is structural dissimilarity in the T1-weighted images of the ASD subjects and HC subjects. In this chapter, I also compared the results of my studies with state-of-the-art methods and showed that the proposed methods did a better classification.

To be consistent with other studies and prove the effectiveness of my techniques, I used the large multisite ABIDE 1 dataset. In the case of the rs-fMRI data, using the spectrum of the Laplacian matrix and the topological centralities, I achieved a classification accuracy of 78.4% using LDA and 79.2% using DNN. I also developed a feature selection algorithm to avoid overfitting in the

traditional machine learning classifiers. However, I achieved a better classification accuracy of 96.6% using the raw T1-weighted images. In summary, comparing the studies of this thesis with other state-of-the-art methods, I can say that the proposed studies can help diagnose ASD more accurately and reliably.

6.2 Future Work

- In Chapter 2, the ROIs of the brain network are defined using 264 ROI based parcellation scheme. However, there are also other brain atlases. A study can be carried to train different machine learning classifiers using the same features but different brain atlases.
- The criterion of the feature selection algorithm in Chapter 3 is the accuracy of the LDA. It means the algorithm tries to find features that will increase the accuracy of the LDA classifier. The feature selection algorithm can be changed to optimize different traditional machine learning classifiers. It will help to find different feature sets for different classifiers.
- I created the feature selection algorithm in Chapter 3 using the wrapper method. However, there are also other feature selection algorithms available such as filter method, embedded method. A feature selection algorithm using different methods can help increase the classification accuracy.
- I did all the experiments in Chapter 3 and Chapter 4 using the ABIDE 1 database for the diagnosis of ASD. The next step of the study will be to use the data from different brain diseases such as AD, schizophrenia. Using the spectrum of the Laplacian matrix of the brain network as a feature to diagnose those diseases will help to improve the efficiency, effectiveness of the feature.
- In Chapter 5, I only experimented with the axial view of the T1-weighted MRI images. Using the neuroimaging software, sagittal and corneal views of the MRI images can also be generated. In future experiments, the data from different views can be studied.
- In Chapter 5, it would interesting to conduct the experiments using a holdout dataset. The dataset should not be used when designing the model. It will be used only once to measure the classification accuracy.
- A major problem of studying medical images is the scarcity of data. There are different data augmentation techniques to create synthetic data. Recently, the use of Generative

Adversial Network (GAN) is becoming popular for data augmentation. Using the GAN to synthesize more data will help to build a better classification model.

REFERENCES

[1] T. Hirvikoski, *et al.*, "Premature mortality in autism spectrum disorder," *Br. J. Psychiatry*, vol. 208, no. 3, pp. 232–238, Mar. 2016.

[2] J. Guan and G. Li, "Injury Mortality in Individuals With Autism.," *Am. J. Public Health*, vol. 107, no. 5, pp. 791–793, May 2017.

[3] "Autism Spectrum Disorder among Children and Youth in Canada 2018 - Canada.ca." [Online]. Available: https://www.canada.ca/en/public-health/services/publications/diseases-conditions/autism-spectrum-disorder-children-youth-canada-2018.html. [Accessed: 13-May-2019].

[4] "About Our Partnership." [Online]. Available: https://www.alphaxidelta.org/about-our-partnership. [Accessed: 13-May-2019].

[5] C. Lord, *et al.*, "Autism Diagnostic Observation Schedule: A Standardized Observation of Communicative and Social Behavior,", *Journal of Autism and Developmental Disorders*, vol. 19, no. 2, pp. 185-212, 1989.

[6] C. Lord, M. Rutter, and A. Le Couteur, "Autism Diagnostic Interview-Revised: A Revised Version of a Diagnostic Interview for Caregivers of Individuals with Possible Pervasive Developmental Disorders," *Journal of Autism and Developmental Disorders*, vol. 24, no. 5, pp. 659-685, 1994.

[7] J. R. Sato, *et al.*, "Inter-regional cortical thickness correlations are associated with autistic symptoms: A machine-learning approach," *J. Psychiatr. Res.*, vol. 47, no. 4, pp. 453–459, Apr. 2013.

[8] E. Varol, *et al.*, "Feature ranking based nested support vector machine ensemble for medical image classification," in *2012 9th IEEE International Symposium on Biomedical Imaging (ISBI)*, pp. 146–149, 2012.

[9] M. Ismail, *et al.*, "A new deep-learning approach for early detection of shape variations in autism using structural mri," in *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 1057–1061, 2017.

[10] K. Al-jabery, *et al.*, “Ensemble statistical and subspace clustering model for analysis of autism spectrum disorder phenotypes,” in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3329–3333, 2016.

[11] N. Dvornek, P. Ventola and J. Duncan, "Combining phenotypic and resting-state fMRI data for autism classification with recurrent neural networks", in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 2018.

[12] O. Altay and M. Ulas, “Prediction of the autism spectrum disorder diagnosis with linear discriminant analysis classifier and K-nearest neighbor in children,” in *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, pp. 1–4, 2018.

[13] T. Iidaka, “Resting state functional magnetic resonance imaging and neural network classified autism and control,” *Cortex*, vol. 63, pp. 55–67, 2015.

[14] A. M. Pagnozzi, *et al.*, “A systematic review of structural MRI biomarkers in autism spectrum disorder: A machine learning perspective,” *Int. J. Dev. Neurosci.*, vol. 71, pp. 68–82, 2018.

[15] E. Wong, *et al.*, “Riemannian Regression and Classification Models of Brain Networks Applied to Autism,” *Connectomics in NeuroImaging*, pp. 78–87, 2018.

[16] F. Pereira, T. Mitchell, and M. Botvinick, “Machine learning classifiers and fMRI: A tutorial overview,” *Neuroimage*, vol. 45, no. 1, pp. S199–S209, 2009.

[17] “Magnetic Resonance Imaging (MRI) | National Institute of Biomedical Imaging and Bioengineering.” [Online]. Available: https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri. [Accessed: 14-May-2019].

[18] D. Mitchell and M. Cohen, *MRI principles*, 2nd ed. Philadelphia: Saunders, 2004.

[19] J. D. Rudie, *et al.*, “Altered functional and structural brain network organization in autism,” *NeuroImage Clin.*, vol. 2, pp. 79–94, Jan. 2013.

[20] K. Murphy, R. M. Birn, and P. A. Bandettini, “Resting-state fMRI confounds and cleanup,” *Neuroimage*, vol. 80, pp. 349–359, Oct. 2013.

[21] M. Greicius, “Resting-state functional connectivity in neuropsychiatric disorders,” *Curr. Opin. Neurol.*, vol. 24, no. 4, pp. 424–430, Aug. 2008.

[22] D. C. Van Essen and K. Ugurbil, “The future of the human connectome,” *Neuroimage*, vol. 62, no. 2, pp. 1299–1310, 2012.

[23] K. Strimbu and J. A. Tavel, “What are biomarkers?,” *Curr. Opin. HIV AIDS*, vol. 5, no. 6, pp. 463–6, 2010.

[24] C. R. Marshall, *et al.*, “Structural Variation of Chromosomes in Autism Spectrum Disorder,” *Am. J. Hum. Genet.*, vol. 82, no. 2, pp. 477–488, 2008.

[25] R. G. Port, *et al.*, “Prospective MEG biomarkers in ASD: pre-clinical evidence and clinical promise of electrophysiological signatures.,” *Yale J. Biol. Med.*, vol. 88, no. 1, pp. 25–36, 2015.

[26] L. Billeci, *et al.*, “On the application of quantitative EEG for characterizing autistic brain: a systematic review.,” *Front. Hum. Neurosci.*, vol. 7, p. 442, 2013.

[27] A. C. Stanfield, *et al.*, “Towards a neuroanatomy of autism: A systematic review and meta-analysis of structural magnetic resonance imaging studies,” *Eur. Psychiatry*, vol. 23, no. 4, pp. 289–299, Jun. 2008.

[28] G. S. Dichter, J. N. Felder, and J. W. Bodfish, “Autism is characterized by dorsal anterior cingulate hyperactivation during social target detection.,” *Soc. Cogn. Affect. Neurosci.*, vol. 4, no. 3, pp. 215–26, Sep. 2009.

[29] E. Redcay, *et al.*, “Atypical brain activation patterns during a face-to-face joint attention game in adults with autism spectrum disorder,” *Hum. Brain Mapp.*, vol. 34, no. 10, pp. 2511–2523, Oct. 2013.

[30] G. B. Karas, *et al.*, “Global and local gray matter loss in mild cognitive impairment and Alzheimer’s disease,” *Neuroimage*, vol. 23, no. 2, pp. 708–716, Oct. 2004.

[31] P. Vemuri, *et al.*, “Alzheimer’s disease diagnosis in individual subjects using structural MR images: Validation studies,” *Neuroimage*, vol. 39, no. 3, pp. 1186–1197, 2008.

[32] A. Raine, *et al.*, "An evaluation of structural and functional prefrontal deficits in schizophrenia: MRI and neuropsychological measures," *Psychiatry Res. Neuroimaging*, vol. 45, no. 2, pp. 123–137, 1992.

[33] L. L. Altshuler, *et al.*, "An MRI study of temporal lobe structures in men with bipolar disorder or schizophrenia," *Biol. Psychiatry*, vol. 48, no. 2, pp. 147–162, 2000.

[34] C. G. Wible, *et al.*, "Prefrontal cortex, negative symptoms, and schizophrenia: an MRI study," *Psychiatry Res. Neuroimaging*, vol. 108, no. 2, pp. 65–78, 2001.

[35] M. A. Just, *et al.*, "Identifying Autism from Neural Representations of Social Interactions: Neurocognitive Markers of Autism," *PLoS One*, vol. 9, no. 12, p. 113879, 2014.

[36] J. A. Nielsen, *et al.*, "Multisite functional connectivity MRI classification of autism: ABIDE results," *Front. Hum. Neurosci.*, vol. 7, p. 599, 2013.

[37] M. Rubinov and O. Sporns, "Complex network measures of brain connectivity: Uses and interpretations," *Neuroimage*, vol. 52, no. 3, pp. 1059–1069, 2010.

[38] E. Bullmore and O. Sporns, "Complex brain networks: graph theoretical analysis of structural and functional systems," *Nat. Rev. Neurosci.*, vol. 10, no. 4, pp. 312–312, 2009.

[39] E. Delbruck, *et al.*, "Functional connectivity in ASD: Atypical pathways in brain networks supporting action observation and joint attention," *Brain Res.*, vol. 1706, pp. 157–165, 2019.

[40] J. D. Rudie, *et al.*, "Reduced functional integration and segregation of distributed neural systems underlying social and emotional information processing in autism spectrum disorders," *Cereb. Cortex*, vol. 22, no. 5, pp. 1025–1037, 2012.

[41] S. E. Schipul, T. A. Keller, and M. A. Just, "Inter-regional brain communication and its disturbance in autism," *Front. Syst. Neurosci.*, vol. 5, p. 10, 2011.

[42] D. P. Kennedy and E. Courchesne, "The intrinsic functional organization of the brain is altered in autism," *Neuroimage*, vol. 39, no. 4, pp. 1877–1885, 2008.

[43] C. Lebel, *et al.*, "Diffusion tensor imaging of white matter tract evolution over the lifespan," *Neuroimage*, vol. 60, no. 1, pp. 340–352, 2012.

[44] J. S. Nomi and L. Q. Uddin, “Developmental changes in large-scale network connectivity in autism,” *NeuroImage Clin.*, vol. 7, pp. 732–741, 2015.

[45] E. Tolan and Z. Isik, “Graph theory based classification of brain connectivity network for autism spectrum disorder,” Springer, Cham, pp. 520–530, 2018.

[46] C. P. Chen, *et al.*, “Diagnostic classification of intrinsic functional connectivity highlights somatosensory, default mode, and visual regions in autism,” *NeuroImage Clin.*, vol. 8, pp. 238–245, 2015.

[47] S. Koyamada, *et al.*, “Deep learning of fMRI big data: a novel approach to subject-transfer decoding,”, ” *arXiv preprint arXiv:1502.00093*, 2015.

[48] S. M. Plis, *et al.*, “Deep learning for neuroimaging: a validation study,” *Front. Neurosci.*, vol. 8, p. 229, 2014.

[49] A. S. Heinsfeld, *et al.*, “Identification of autism spectrum disorder using deep learning and the ABIDE dataset,” *NeuroImage Clin.*, vol. 17, pp. 16–23, 2017.

[50] B. C. Munsell, *et al.*, “Evaluation of machine learning algorithms for treatment outcome prediction in patients with epilepsy based on structural connectome data,” *Neuroimage*, vol. 118, pp. 219–230, 2015.

[51] M. Khosla, *et al.*, “3D convolutional neural networks for classification of functional connectomes.” *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 137-145, 2018.

[52] J. Bruna, *et al.*, “Spectral networks and locally connected networks on graphs.” *arXiv preprint arXiv:1312.6203v3*, 2014,

[53] D. I. Shuman, B. Ricaud, and P. Vandergheynst, “A windowed graph Fourier transform,” *2012 IEEE Stat. Signal Process. Work. SSP 2012*, no. 3, pp. 133–136, 2012.

[54] J. Kawahara, *et al.*, “BrainNetCNN: Convolutional neural networks for brain networks; towards predicting neurodevelopment,” *Neuroimage*, vol. 146, pp. 1038–1049, 2017.

[55] N. C. Dvornek, *et al.*, “Identifying autism from resting-state fmri using long short-term memory networks,” *Springer, Cham*, pp. 362–370, 2017.

[56] R. Anirudh and J. J. Thiagarajan, “Bootstrapping graph convolutional neural networks for autism spectrum disorder classification.”, *arXiv preprint arXiv:1704.07487v2*, *2018.*

[57] S. Parisot, *et al.*, “Spectral graph convolutions for population-based disease prediction,” *Springer, Cham,* 2017, pp. 177–185.

[58] M. R. Arbabshirani, S. Plis, J. Sui, and V. D. Calhoun, “Single subject prediction of brain disorders in neuroimaging: Promises and pitfalls,” *Neuroimage*, vol. 145, pp. 137–165, 2017.

[59] A. Abraham, *et al.*, “Deriving reproducible biomarkers from multi-site resting-state data: An Autism-based example,” *Neuroimage*, vol. 147, pp. 736–745, 2017.

[60] J. Wang, *et al.*, “Parcellation-dependent small-world brain functional networks: A resting-state fMRI study,” *Hum. Brain Mapp.*, vol. 30, no. 5, pp. 1511–1523, 009.

[61] A. Zalesky, *et al.*, “Whole-brain anatomical networks: Does the choice of nodes matter?,” *Neuroimage*, vol. 50, no. 3, pp. 970–983, 2010.

[62] C. T. Butts, “Revisiting the Foundations of Network Analysis.”, *Science*, vol. 325, no. 5939, pp. 414–6, 2009.

[63] N. E. V. Foster, *et al.*, “Structural gray matter differences during childhood development in autism spectrum disorder: A multimetric approach,” *Pediatr. Neurol.*, vol. 53, no. 4, pp. 350–359, 2015.

[64] C. Ecker, S. Y. Bookheimer, and D. G. M. Murphy, “Neuroimaging in autism spectrum disorder: brain structure and function across the lifespan,” *Lancet Neurol.*, vol. 14, no. 11, pp. 1121–1134, 2015.

[65] H. C. Hazlett, *et al.*, “Early brain development in infants at high risk for autism spectrum disorder,” *Nat. Publ. Gr.*, vol. 542, 2017.

[66] G. Wallace, *et al.*, "Age-related temporal and parietal cortical thinning in autism spectrum disorders", *Brain*, vol. 133, no. 12, pp. 3745-3754, 2010.

[67] E. Courchesne, *et al.*, “Unusual brain growth patterns in early life in patients with autistic disorder,” *Neurology*, vol. 57, no. 2, pp. 245 LP – 254, 2001.

[68] T. Hashimoto, *et al.*, "Development of the brainstem and cerebellum in autistic patients," *J. Autism Dev. Disord.*, vol. 25, no. 1, pp. 1–18, 1995.

[69] C. Ecker, *et al.*, "Brain surface anatomy in adults with autism," *JAMA Psychiatry*, vol. 70, no. 1, p. 59, 2013.

[70] H. C. Hazlett, *et al.*, "Magnetic resonance imaging and head circumference study of brain size in autism," *Arch. Gen. Psychiatry*, vol. 62, no. 12, p. 1366, 2005.

[71] N. Lange, *et al.*, "Longitudinal volumetric brain changes in autism spectrum disorder ages 6-35 years," *Autism Res.*, vol. 8, no. 1, pp. 82–93, 2015.

[72] "ABIDE." [Online]. Available: http://fcon_1000.projects.nitrc.org/indi/abide/. [Accessed: 24-May-2019].

[73] A. Di Martino, *et al.*, "The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism," *Mol. Psychiatry*, vol. 19, no. 6, pp. 659–667, 2014.

[74] "Autism Brain Imaging Data Exchange I ABIDE I." [Online]. Available: http://fcon_1000.projects.nitrc.org/indi/abide/abide_I.html. [Accessed: 24-May-2019].

[75] R. W. Cox, "AFNI: software for analysis and visualization of functional magnetic resonance neuroimages.," *Comput. Biomed. Res.*, vol. 29, no. 3, pp. 162–73, 1996.

[76] M. Jenkinson, *et al.*, "FSL," *Neuroimage*, vol. 62, no. 2, pp. 782–790, 2012.

[77] S. Roy and P. Maji, "A simple skull stripping algorithm for brain MRI," in *2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR)*, pp. 1–6, 2015.

[78] M. Jenkinson, *et al.*, "Improved optimization for the robust and accurate linear registration and motion correction of brain images," *Neuroimage*, vol. 17, no. 2, pp. 825–841, Oct. 2002.

[79] M. Jenkinson and S. Smith, "A global optimisation method for robust affine registration of brain images," *Med. Image Anal.*, vol. 5, no. 2, pp. 143–156, Jun. 2001.

[80] L. C. Maas and P. F. Renshaw, "Post-registration spatial filtering to reduce noise in functional MRI data sets," *Magn. Reson. Imaging*, vol. 17, no. 9, pp. 1371–1382, 1999.

[81] J. J. Paakki, *et al.*, "Alterations in regional homogeneity of resting-state brain activity in autism spectrum disorders," *Brain Res.*, vol. 1321, pp. 169–179, 2010.

[82] A. Hahamy, M. Behrmann, and R. Malach, "The idiosyncratic brain: distortion of spontaneous connectivity patterns in autism spectrum disorder," *Nat. Neurosci.*, vol. 18, no. 2, pp. 302–309, 2015.

[83] P. Fransson, *et al.*, "The Functional Architecture of the Infant Brain as Revealed by Resting-State fMRI," *Cereb. Cortex*, vol. 21, no. 1, pp. 145–154, 2011.

[84] N. Yahata, *et al.*, "A small number of abnormal brain connections predicts adult autism spectrum disorder," *Nat. Commun.*, vol. 7, no. 1, p. 11254, 2016.

[85] P. Hagmann, *et al.*, "Mapping the Structural Core of Human Cerebral Cortex," *PLoS Biol.*, vol. 6, no. 7, p. e159, 2008.

[86] F. Megumi, *et al.*, "Functional MRI neurofeedback training on connectivity between two regions induces long-lasting changes in intrinsic functional network," *Front. Hum. Neurosci.*, vol. 9, p. 160, 2015.

[87] J. D. Power, *et al.*, "Functional network organization of the human brain,", *Neuron*, vol. 72, no. 4, pp. 665–678, 2011.

[88] R. Henson, "Analysis of fMRI time series: Linear time-invariant models, event-related fMRI, and optimal experimental design," *Hum. Brain Funct.*, pp. 793–822, Jan. 2004.

[89] S. Hayasaka and P. J. Laurienti, "Comparison of characteristics between region-and voxel-based network analyses in resting-state fMRI data," *Neuroimage*, vol. 50, no. 2, pp. 499–508, Apr. 2010.

[90] J. Wang, X. Zuo, and Y. He, "Graph-based network analysis of resting-state functional MRI," *Front. Syst. Neurosci.*, vol. 4, p. 16, Jun. 2010.

[91] L. Tian, *et al.*, "The relationship within and between the extrinsic and intrinsic systems indicated by resting state correlational patterns of sensory cortices.," *Neuroimage*, vol. 36, no. 3, pp. 684–90, Jul. 2007.

[92] L. Q. Uddin, *et al.*, "Functional connectivity of default mode network components: Correlation, anticorrelation, and causality," *Hum. Brain Mapp.*, vol. 30, no. 2, pp. 625–637, Feb. 2009.

[93] M. Plitt, K. A. Barnes, and A. Martin, "Functional connectivity classification of autism identifies highly predictive brain features but falls short of biomarker standards," *NeuroImage Clin.*, vol. 7, pp. 359–366, Jan. 2015.

[94] Y. Jiao, *et al.*, "Predictive models of autism spectrum disorder based on brain regional cortical thickness," *Neuroimage*, vol. 50, no. 2, pp. 589–599, Apr. 2010.

[95] S. Ghiassian, *et al.*, "Learning to classify psychiatric disorders based on fmr images: autism vs healthy and ADHD vs healthy.", ' in *Proc. 3rd NIPS Workshop Mach. Learn. Interpretation NeuroImag.*, pp. 1–7, 2013.

[96] C. Y. Wee, *et al.*, "Diagnosis of autism spectrum disorders using regional and interregional morphological features.," *Hum. Brain Mapp.*, vol. 35, no. 7, pp. 3414–30, Jul. 2014.

[97] B. Miao and Y. Zhang, "A feature selection method for classification of ADHD," in *2017 4th International Conference on Information, Cybernetics and Computational Social Systems (ICCSS)*, pp. 21–25, 2017.

[98] X. Guo, *et al.*, "Diagnosing Autism Spectrum Disorder from Brain Resting-State Functional Connectivity Patterns Using a Deep Neural Network with a Novel Feature Selection Method," *Front. Neurosci.*, vol. 11, p. 460, Aug. 2017.

[99] C. Bishop, *Pattern recognition and machine learning*. New York: Springer, 2009.

[100] X. Huang, M. Ghodsi, and H. Hassani, "A Novel similarity measure based on eigenvalue distribution," *Trans. A. Razmadze Math. Inst.*, vol. 170, no. 3, pp. 352–362, Dec. 2016.

[101] M. E. J. Newman, "Assortative Mixing in Networks," *Phys. Rev. Lett.*, vol. 89, no. 20, p. 208701, Oct. 2002.

[102] M. Mijalkov, *et al.*, "BRAPH: A graph theory software for the analysis of brain connectivity," *PLoS One*, vol. 12, no. 8, p. e0178798, Aug. 2017.

[103] M. Piraveenan, M. Prokopenko, and A. Zomaya, “Assortative mixing in directed biological networks,” *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, vol. 9, no. 1, pp. 66–78, Jan. 2012.

[104] M. E. J. Newman, “Mixing patterns in networks,” *Phys. Rev. E*, vol. 67, no. 2, p. 026126, Feb. 2003.

[105] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” *Nature*, vol. 393, no. 6684, pp. 440–442, Jun. 1998.

[106] O. Sporns and C. J. Honey, “Small worlds inside big brains,” *Proc. Natl. Acad. Sci.*, vol. 103, no. 51, pp. 19219–19220, Dec. 2006.

[107] K. Supekar, *et al.*, “Network analysis of intrinsic functional brain connectivity in Alzheimer’s disease.,” *PLoS Comput. Biol.*, vol. 4, no. 6, p. e1000100, Jun. 2008.

[108] E. Glerean, *et al.*, “Reorganization of functionally connected brain subnetworks in high-functioning autism,” *Hum. Brain Mapp.*, vol. 37, no. 3, pp. 1066–1079, Mar. 2016.

[109] J. Brownlee, “An Introduction to Feature Selection.” [Online]. Available: https://machinelearningmastery.com/an-introduction-to-feature-selection/. [Accessed: 07-Aug-2019].

[110] H. Liu, H. Motoda, R. Setiono, and Z. Zhao, “Feature Selection: An ever evolving frontier in data mining.” in *Proceedings of the Fourth International Workshop on Feature Selection in Data Mining*, Hyderabad, India, pp. 4–13, 2010.

[111] J. H. Friedman, “On Bias, Variance, 0/1—Loss, and the Curse-of-Dimensionality,” *Data Min. Knowl. Discov.*, vol. 1, no. 1, pp. 55–77, 1997.

[112] L. Jourdan, C. Dhaenens, and E. G. Talbi, “A Genetic Algorithm for Feature Selection in Data-Mining for Genetics,” ” in *Proceedings of the 4th Metaheuristics International ConferencePorto (MIC’2001)*, Porto, Portugal, pp. 29–34, 2001.

[113] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, Jan. 2014.

[114] “Sequential feature selection - MATLAB sequentialfs.” [Online]. Available: https://www.mathworks.com/help/stats/sequentialfs.html. [Accessed: 02-Jun-2019].

[115] “Train Logistic Regression Classifiers Using Classification Learner App - MATLAB & Simulink.” [Online]. Available: https://www.mathworks.com/help/stats/train-logistic-regression-classifiers-in-classification-learner-app.html. [Accessed: 03-Jun-2019].

[116] E. Hosseini-Asl, R. Keynton, and A. El-Baz, “Alzheimer’s disease diagnostics by adaptation of 3D convolutional network,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 126–130.

[117] H. I. Suk, S. W. Lee, D. Shen, and T. A. D. N. Initiative, “Latent feature representation with stacked auto-encoder for AD/MCI diagnosis,” *Brain Struct. Funct.*, vol. 220, no. 2, pp. 841–859, Mar. 2015.

[118] Y. Kong, *et al.*, “Classification of autism spectrum disorder by combining brain connectivity and deep neural network classifier,” *Neurocomputing*, vol. 324, pp. 63–68, Jan. 2019.

[119] K. Han, C. Li, and X. Shi, “Autoencoder Feature Selector.”, *arXiv preprint arXiv:1710.08310*, 2017.

[120] “Train Classification Models in Classification Learner App - MATLAB & Simulink.” [Online]. Available: https://www.mathworks.com/help/stats/train-classification-models-in-classification-learner-app.html. [Accessed: 24-Jul-2019].

[121] X. Li, *et al.*, “2-Channel convolutional 3D deep neural network (2CC3D) for fMRI analysis: ASD classification and feature learning,” in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 1252–1255, 2018.

[122] G. D. Waiter, *et al.*, “Structural white matter deficits in high-functioning individuals with autistic spectrum disorder: A voxel-based investigation,” *Neuroimage*, pp. 455-461, 2005.

[123] R. Chen, Y. Jiao, and E. H. Herskovits, “Structural MRI in Autism Spectrum Disorder,” *Pediatr. Res.*, vol. 69, p. 63R, May 2011.

[124] “Autoencoders — Deep Learning bits #1.”[Online]. Available: https://hackernoon.com/ autoencoders-deep-learning-bits-1-11731e200694?source=post _page. [Accessed: 29-Jul-2019].

[125] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction,” *Springer*, Berlin, Heidelberg, 2011, pp. 52–59.

[126] A. Krizhevsky and G. E. Hinton, “Using Very Deep Autoencoders for Content-Based Image Retrieval.”in *ESANN*, 2011.

[127] Q. Ke, *et al.*, “Computer Vision for Human–Machine Interaction,” *Comput. Vis. Assist. Healthc.*, pp. 127–145, Jan. 2018.

[128] M. Ribeiro, A. E. Lazzaretti, and H. S. Lopes, “A study of deep convolutional auto-encoders for anomaly detection in videos,” *Pattern Recognit. Lett.*, vol. 105, pp. 13–22, Apr. 2018.

[129] “A Gentle Introduction to Pooling Layers for Convolutional Neural Networks.” [Online]. Available: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks. [Accessed: 31-Jul-2019].

[130] N. Yahata, et al., "A small number of abnormal brain connections predicts adult autism spectrum disorder", *Nature Communications*, vol. 7, no. 1, 2016.

[131] T. Watanabe and G. Rees, "Brain network dynamics in high-functioning individuals with autism", *Nature Communications*, vol. 8, no. 1, 2017.

[132] X. Xing, J. Ji, and Y. Yao, “Convolutional Neural Network with Element-wise Filters to Extract Hierarchical Topological Features for Brain Networks,” in Proc. - 2018 IEEE Int. Conf. Bioinforma. Biomed. BIBM 2018, pp. 780–783, 2019.

APPENDIX A
PERFORMANCE ANALYSIS OF AUTOENCODER BASED FEATURE SELECTOR

***Table A-1**: Performance of different machine learning classifier before and after pre-training autoencoder based feature selector with DNN classifier*

Machine Learning Classifier	ACC (%) (STDEV)		AUC (%) (STDEV)	
	Before Pre-training	After Pre-training	Before Pre-training	After Pre-training
Fine TREE	62.2 (3.0)	67.1 (2.6)	64.0 (4.0)	69.3 (3.2)
Medium Tree	62.2 (2.9)	67.3 (2.6)	63.9 (4.3)	69.7 (2.9)
Coarse Tree	68.5 (3.6)	71.3 (2.3)	67.5 (2.8)	71.2 (2.5)
Logistic Regression	66.6 (1.4)	71.3 (2.1)	70.5 (1.4)	76.8 (1.8)
Linear SVM	66.7 (1.7)	72.8 (2.9)	71.0 (1.2)	76.3 (3.4)
Quadratic SVM	66.6 (2.2)	66.9 (3.9)	68.8 (2.9)	70.4 (6.5)
Cubic SVM	64.9 (2.0)	65.2 (2.9)	65.1 (2.2)	65.6 (4.4)
Fine Gaussian SVM	66.5 (1.9)	69.8 (1.8)	71.4 (3.0)	74.2 (3.0)
Medium Gaussian SVM	69.3 (2.2)	73.7 (1.6)	73.3 (0.7)	76.7 (1.6)
Coarse Gaussian SVM	67.6 (3.2)	73.0 (2.5)	71.6 (3.0)	77.9 (1.4)
Fine KNN	63.8 (2.4)	66.7 (2.4)	64.4 (3.7)	66.8 (1.9)
Medium KNN	66.1 (1.8)	73.2 (1.7)	72.9 (1.2)	77.7 (2.1)
Coarse KNN	63.6 (1.7)	68.2 (3.4)	65.1 (3.8)	74.6 (3.2)
Cosine KNN	68.5 (2.5)	74.6 (1.9)	74.7 (2.9)	78.7 (2.1)
Cubic KNN	65.8 (1.5)	72.7 (2.1)	72.4 (1.0)	77.4 (2.0)
Weighted KNN	66.6 (2.5)	72.7 (2.2)	74.3 (1.3)	77.2 (2.0)
Boosted Trees	65.3 (2.9)	68.6 (2.0)	71.2 (3.1)	72.6(2.2)
Bagged Trees	66.6 (2.0)	70.9 (1.9)	73.6 (1.8)	76.6 (2.2)
Subspace Discriminant	68.9 (2.6)	74.4 (1.1)	72.2 (1.6)	77.8 (1.5)
Subspace KNN	65.9 (3.5)	69.8 (2.0)	73.9 (3.3)	75.8 (2.4)
RUSBoosted Trees	64.1 (2.5)	67.5 (2.3)	67.2 (2.7)	70.6 (1.5)

APPENDIX B
LIST OF THE 264 ROIs

***Table B-1**: List of the 264 ROIs with their structural and functional label*

ROI	x, y, z coordinates in MNI152 space			Structural label	Functional label
1	17	-91	-14	Right Occipital Pole	Visual network
2	8	-91	-7	Right Occipital Pole	Visual network
3	-7	-71	42	Left Precuneous Cortex	Default mode network
4	15	-63	26	Right Precuneous Cortex	Visual network
5	-12	-95	-13	Left Occipital Pole	Visual network
6	26	-79	-16	Right Occipital Fusiform Gyrus	Visual network
7	6	-72	24	Right Cuneal Cortex	Visual network
8	-40	-88	-6	Left Lateral Occipital Cortex inferior division	Visual network
9	11	-66	42	Right Precuneous Cortex	Visual network
10	-26	-90	3	Left Lateral Occipital Cortex inferior division	Visual network
11	-25	-98	-12	Left Occipital Pole	Visual network
12	27	-97	-13	Right Occipital Pole	Visual network
13	-24	-91	19	Left Occipital Pole	Visual network
14	37	-81	1	Right Lateral Occipital Cortex inferior division	Visual network
15	-33	-79	-13	Left Occipital Fusiform Gyrus	Visual network
16	-18	-76	-24	Left Occipital Fusiform Gyrus	Visual network
17	6	-81	6	Right Intracalcarine Cortex	Visual network
18	20	-86	-2	Right Occipital Fusiform Gyrus	Visual network
19	43	-72	28	Right Lateral Occipital Cortex superior division	Visual network
20	-8	-81	7	Left Intracalcarine Cortex	Visual network
21	24	-87	24	Right Lateral Occipital Cortex superior division	Visual network
22	-14	-91	31	Left Occipital Pole	Visual network
23	-3	-81	21	Left Cuneal Cortex	Visual network
24	33	-53	44	Right Superior Parietal Lobule	Visual network
25	27	-59	-9	Right Temporal Occipital Fusiform Cortex	Visual network
26	-28	-79	19	Left Lateral Occipital Cortex superior division	Visual network

27	29	-77	25	Right Lateral Occipital Cortex superior division	Visual network
28	-27	-71	37	Left Lateral Occipital Cortex superior division	Visual network
9	-16	-77	34	Left Cuneal Cortex	Visual network
30	37	-84	13	Right Lateral Occipital Cortex superior division	Visual network
31	8	-72	11	Right Intracalcarine Cortex	Visual network
32	43	-78	-12	Right Lateral Occipital Cortex inferior division	Visual network
33	-42	-60	-9	Left Inferior Temporal Gyrus temporooccipital part	Visual network
34	15	-87	37	Right Occipital Pole	Visual network
35	27	-37	-13	Right Parahippocampal Gyrus posterior division	Visual network
36	58	-53	-14	Right Inferior Temporal Gyrus temporooccipital part	Visual network
37	15	-77	31	Right Cuneal Cortex	Visual network
38	20	-66	2	Right Intracalcarine Cortex	Visual network
39	-28	-58	48	Left Superior Parietal Lobule	Visual network
40	-47	-76	-10	Left Lateral Occipital Cortex inferior division	Visual network
41	-18	-68	5	Left Intracalcarine Cortex	Visual network
42	46	-47	-17	Right Temporal Occipital Fusiform Cortex	Visual network
43	42	-66	-8	Right Lateral Occipital Cortex inferior division	Visual network
44	-47	-51	-21	Left Inferior Temporal Gyrus temporooccipital part	Visual network
45	18	-47	-10	Right Lingual Gyrus	Visual network
46	-15	-72	-8	Left Lingual Gyrus	Visual network
47	-16	-52	-1	Left Lingual Gyrus	Visual network
48	-42	-74	0	Left Lateral Occipital Cortex inferior division	Visual network
49	4	-48	51	Right Precuneous Cortex	Default mode network
50	40	-72	14	Right Lateral Occipital Cortex inferior division	Visual network
51	22	-65	48	Right Lateral Occipital Cortex superior division	Visual network

52	42	0	47	Right Precentral Gyrus	Frontal-parietal network
53	25	-58	60	Right Lateral Occipital Cortex superior division	Visual network
54	46	-59	4	Right Middle Temporal Gyrus temporooccipital part	Visual network
55	-38	-27	69	Left Postcentral Gyrus	Somatosensory network
56	-38	-15	69	Left Precentral Gyrus	Somatosensory network
57	-23	-30	72	Left Postcentral Gyrus	Somatosensory network
58	13	-33	75	Right Postcentral Gyrus	Somatosensory network
59	-13	-17	75	Left Precentral Gyrus	Somatosensory network
60	-40	-19	54	Left Precentral Gyrus	Somatosensory network
61	29	-17	71	Right Precentral Gyrus	Somatosensory network
62	2	-28	60	Right Precentral Gyrus	Somatosensory network
63	33	-12	-34	Right Temporal Fusiform Cortex posterior division	Default mode network
64	-16	-46	73	Left Postcentral Gyrus	Somatosensory network
65	-7	-33	72	Left Postcentral Gyrus	Somatosensory network
66	42	-20	55	Right Postcentral Gyrus	Somatosensory network
67	-7	-21	65	Left Precentral Gyrus	Somatosensory network
68	-21	-31	61	Left Postcentral Gyrus	Somatosensory network
69	66	-8	25	Right Postcentral Gyrus	Somatosensory network
70	10	-17	74	Right Precentral Gyrus	Somatosensory network
71	-37	-29	-26	Left Temporal Fusiform Cortex posterior division	Somatosensory network

72	20	-29	60	Right Precentral Gyrus	Somatosensory network
73	-31	-10	-36	Left Temporal Fusiform Cortex anterior division	Default mode network
74	10	-46	73	Right Postcentral Gyrus	Somatosensory network
75	22	-42	69	Right Superior Parietal Lobule	Somatosensory network
76	3	-17	58	Right Precentral Gyrus	Somatosensory network
77	50	-20	42	Right Postcentral Gyrus	Somatosensory network
78	38	-17	45	Right Precentral Gyrus	Somatosensory network
79	-29	-43	61	Left Superior Parietal Lobule	Somatosensory network
80	29	-39	59	Right Superior Parietal Lobule	Somatosensory network
81	-16	-65	-20	Left VI	Visual network
82	52	-34	-27	Right Inferior Temporal Gyrus posterior division	Frontal-parietal network
83	22	-58	-23	Right VI	Somatosensory network
84	1	-62	-18	Vermis VI	Somatosensory network
85	-14	-18	40	Left Precentral Gyrus	Somatosensory network
86	-49	-11	35	Left Precentral Gyrus	Somatosensory network
87	-53	-10	24	Left Postcentral Gyrus	Somatosensory network
88	51	-6	32	Right Precentral Gyrus	Somatosensory network
89	-17	-59	64	Left Lateral Occipital Cortex superior division	Somatosensory network
90	-54	-23	43	Left Postcentral Gyrus	Somatosensory network
91	43	-23	20	Right Parietal Operculum Cortex	Somatosensory network

92	-55	-40	14	Left Planum Temporale	Somatosensory network
93	36	-9	14	Right Insular Cortex	Somatosensory network
94	-56	-45	-24	Left Inferior Temporal Gyrus temporooccipital part	Default mode network
95	-45	-32	47	Left Postcentral Gyrus	Somatosensory network
96	-38	-33	17	Left Planum Temporale	Somatosensory network
97	44	-8	57	Right Precentral Gyrus	Somatosensory network
98	-53	-22	23	Left Central Opercular Cortex	Somatosensory network
99	0	-15	47	Left Cingulate Gyrus posterior division	Somatosensory network
100	11	-39	50	Right Precuneous Cortex	Somatosensory network
101	-49	-26	5	Left Planum Temporale	Somatosensory network
102	-55	-9	12	Left Central Opercular Cortex	Somatosensory network
103	-5	-28	-4	Brain-Stem	Frontal-parietal network
104	58	-16	7	Right Planum Temporale	Somatosensory network
105	32	-26	13	Right Insular Cortex	Somatosensory network
106	56	-5	13	Right Central Opercular Cortex	Somatosensory network
107	10	-62	61	Right Lateral Occipital Cortex superior division	Visual network
108	-7	-52	61	Left Precuneous Cortex	Somatosensory network
109	10	-2	45	Right Cingulate Gyrus anterior division	Somatosensory network
110	-10	-2	42	Left Juxtapositional Lobule Cortex	Somatosensory network
111	-30	-27	12	Left Insular Cortex	Somatosensory network

112	59	-17	29	Right Postcentral Gyrus	Somatosensory network
113	19	-8	64	Right Superior Frontal Gyrus	Somatosensory network
114	-32	-55	-25	Left VI	Visual network
115	-60	-25	14	Left Planum Temporale	Somatosensory network
116	-52	-63	5	Left Lateral Occipital Cortex inferior division	Somatosensory network
117	47	-30	49	Right Postcentral Gyrus	Somatosensory network
118	6	-24	0	Right Thalamus	Frontal-parietal network
119	29	-5	54	Right Precentral Gyrus	Somatosensory network
120	-31	-11	0	Left Putamen	Somatosensory network
121	-16	-5	71	Left Superior Frontal Gyrus	Somatosensory network
122	-50	-34	26	Left Parietal Operculum Cortex	Somatosensory network
123	12	-17	8	Right Thalamus	Somatosensory network
124	-10	-18	7	Left Thalamus	Somatosensory network
125	31	-14	2	Right Putamen	Somatosensory network
126	54	-28	34	Right Supramarginal Gyrus anterior division	Somatosensory network
127	13	-1	70	Right Superior Frontal Gyrus	Somatosensory network
128	65	-33	20	Right Superior Temporal Gyrus posterior division	Somatosensory network
129	-33	-46	47	Left Superior Parietal Lobule	Visual network
130	-45	0	9	Left Central Opercular Cortex	Somatosensory network
131	-3	2	53	Left Juxtapositional Lobule Cortex	Somatosensory network
132	29	1	4	Right Putamen	Somatosensory network

133	37	1	-4	Right Insular Cortex	Somatosensory network
134	7	8	51	Right Juxtapositional Lobule Cortex	Frontal-parietal network
135	49	8	-1	Right Central Opercular Cortex	Frontal-parietal network
136	-51	8	-2	Left Central Opercular Cortex	Frontal-parietal network
137	-34	3	4	Left Insular Cortex	Frontal-parietal network
138	36	10	1	Right Insular Cortex	Frontal-parietal network
139	-1	15	44	Left Paracingulate Gyrus	Frontal-parietal network
140	23	10	1	Right Putamen	Frontal-parietal network
141	-42	38	21	Left Frontal Pole	Frontal-parietal network
142	31	33	26	Right Middle Frontal Gyrus	Frontal-parietal network
143	36	22	3	Right Insular Cortex	Frontal-parietal network
144	-32	-1	54	Left Middle Frontal Gyrus	Frontal-parietal network
145	-35	20	0	Left Insular Cortex	Frontal-parietal network
146	47	10	33	Right Precentral Gyrus	Frontal-parietal network
147	15	5	7	Right Pallidum	Frontal-parietal network
148	-5	18	34	Left Cingulate Gyrus anterior division	Frontal-parietal network
149	-47	11	23	Left Inferior Frontal Gyrus pars opercularis	Frontal-parietal network
150	10	22	27	Right Cingulate Gyrus anterior division	Frontal-parietal network
151	-39	51	17	Left Frontal Pole	Frontal-parietal network
152	38	43	15	Right Frontal Pole	Frontal-parietal network

153	-22	7	-5	Left Putamen	Frontal-parietal network
154	5	23	37	Right Paracingulate Gyrus	Frontal-parietal network
155	37	32	-2	Right Frontal Orbital Cortex	Frontal-parietal network
156	-21	41	-20	Left Frontal Pole	Frontal-parietal network
157	24	32	-18	Right Frontal Orbital Cortex	Frontal-parietal network
158	9	-4	6	Right Thalamus	Frontal-parietal network
159	-23	11	64	Left Superior Frontal Gyrus	Frontal-parietal network
160	43	49	-2	Right Frontal Pole	Frontal-parietal network
161	24	45	-15	Right Frontal Pole	Frontal-parietal network
162	49	-42	45	Right Supramarginal Gyrus posterior division	Frontal-parietal network
163	-41	6	33	Left Middle Frontal Gyrus	Frontal-parietal network
164	-11	26	25	Left Cingulate Gyrus anterior division	Frontal-parietal network
165	48	25	27	Right Middle Frontal Gyrus	Frontal-parietal network
166	48	22	10	Right Inferior Frontal Gyrus pars triangularis	Frontal-parietal network
167	34	16	-8	Right Insular Cortex	Frontal-parietal network
168	-15	4	8	Left Caudate	Frontal-parietal network
169	-42	25	30	Left Middle Frontal Gyrus	Frontal-parietal network
170	34	54	-13	Right Frontal Pole	Frontal-parietal network
171	31	56	14	Right Frontal Pole	Frontal-parietal network
172	26	50	27	Right Frontal Pole	Frontal-parietal network

173	0	30	27	Left Cingulate Gyrus anterior division	Frontal-parietal network
174	-42	45	-2	Left Frontal Pole	Frontal-parietal network
175	55	-45	37	Right Supramarginal Gyrus posterior division	Frontal-parietal network
176	-3	26	44	Left Paracingulate Gyrus	Frontal-parietal network
177	-34	55	4	Left Frontal Pole	Frontal-parietal network
178	-28	52	21	Left Frontal Pole	Frontal-parietal network
179	12	36	20	Right Cingulate Gyrus anterior division	Default mode network
180	-2	38	36	Left Paracingulate Gyrus	Default mode network
181	-53	-49	43	Left Supramarginal Gyrus posterior division	Default mode network
182	40	18	40	Right Middle Frontal Gyrus	Default mode network
183	-3	42	16	Left Cingulate Gyrus anterior division	Default mode network
184	44	-53	47	Right Angular Gyrus	Default mode network
185	-42	-55	45	Left Angular Gyrus	Default mode network
186	34	38	-12	Right Frontal Pole	Frontal-parietal network
187	32	14	56	Right Middle Frontal Gyrus	Frontal-parietal network
188	-44	2	46	Left Precentral Gyrus	Default mode network
189	-10	11	67	Left Superior Frontal Gyrus	Default mode network
190	27	16	-17	Right Frontal Orbital Cortex	Default mode network
191	55	-31	-17	Right Inferior Temporal Gyrus posterior division	Default mode network
192	-49	25	-1	Left Frontal Operculum Cortex	Default mode network
193	53	33	1	Right Inferior Frontal Gyrus pars triangularis	Frontal-parietal network
194	-20	45	39	Left Frontal Pole	Default mode network
195	-11	45	8	Left Cingulate Gyrus anterior division	Default mode network
196	-2	-13	12	Left Thalamus	Default mode network
197	22	39	39	Right Frontal Pole	Default mode network

198	65	-31	-9	Right Middle Temporal Gyrus posterior division	Default mode network
199	8	42	-5	Right Paracingulate Gyrus	Default mode network
200	49	35	-12	Right Frontal Pole	Default mode network
201	54	-43	22	Right Angular Gyrus	Frontal-parietal network
202	2	-24	30	Right Cingulate Gyrus posterior division	Default mode network
203	37	-65	40	Right Lateral Occipital Cortex superior division	Default mode network
204	-8	48	23	Left Paracingulate Gyrus	Default mode network
205	-2	-37	44	Left Cingulate Gyrus posterior division	Default mode network
206	-31	19	-19	Left Frontal Orbital Cortex	Default mode network
207	-16	29	53	Left Superior Frontal Gyrus	Default mode network
208	13	30	59	Right Superior Frontal Gyrus	Default mode network
209	9	54	3	Right Paracingulate Gyrus	Default mode network
210	-7	51	-1	Left Paracingulate Gyrus	Default mode network
211	56	-46	11	Right Middle Temporal Gyrus temporooccipital part	Somatosensory network
212	-20	64	19	Left Frontal Pole	Default mode network
213	65	-24	-19	Right Middle Temporal Gyrus posterior division	Default mode network
214	-35	20	51	Left Middle Frontal Gyrus	Default mode network
215	35	-67	-34	Right Crus I	Default mode network
216	23	33	48	Right Superior Frontal Gyrus	Default mode network
217	6	54	16	Right Paracingulate Gyrus	Default mode network
218	51	-29	-4	Right Middle Temporal Gyrus posterior division	Default mode network
219	13	55	38	Right Frontal Pole	Default mode network
220	-46	31	-13	Left Frontal Orbital Cortex	Default mode network
221	52	-33	8	Right Superior Temporal Gyrus posterior division	Somatosensory network
222	47	-50	29	Right Angular Gyrus	Default mode network
223	-56	-50	10	Left Middle Temporal Gyrus temporooccipital part	Somatosensory network
224	52	-59	36	Right Lateral Occipital Cortex superior division	Default mode network
225	-68	-41	-5	Left Middle Temporal Gyrus posterior division	Default mode network

226	-58	-30	-4	Left Middle Temporal Gyrus posterior division	Default mode network
227	-10	39	52	Left Superior Frontal Gyrus	Default mode network
228	6	64	22	Right Frontal Pole	Default mode network
229	-39	-75	44	Left Lateral Occipital Cortex superior division	Default mode network
230	-2	-35	31	Left Cingulate Gyrus posterior division	Default mode network
231	-18	63	-9	Left Frontal Pole	Default mode network
232	8	41	-24	Right Frontal Medial Cortex	Default mode network
233	-49	-42	1	Left Middle Temporal Gyrus posterior division	Default mode network
234	-41	-75	26	Left Lateral Occipital Cortex superior division	Default mode network
235	17	-80	-34	Right Crus II	Default mode network
236	-58	-26	-15	Left Middle Temporal Gyrus posterior division	Default mode network
237	-10	55	39	Left Frontal Pole	Default mode network
238	-3	44	-9	Left Paracingulate Gyrus	Default mode network
239	8	48	-15	Right Frontal Medial Cortex	Default mode network
240	-44	-65	35	Left Lateral Occipital Cortex superior division	Default mode network
241	-68	-23	-16	Left Middle Temporal Gyrus posterior division	Default mode network
242	28	-77	-32	Right Crus I	Default mode network
243	-34	-38	-16	Left Temporal Fusiform Cortex posterior division	Default mode network
244	6	67	-4	Right Frontal Pole	Default mode network
245	-46	-61	21	Left Lateral Occipital Cortex superior division	Default mode network
246	-13	-40	1	Left Hippocampus	Default mode network
247	52	7	-30	Right Temporal Pole	Default mode network
248	49	-3	-38	Right Inferior Temporal Gyrus anterior division	Default mode network
249	17	-28	-17	Right Parahippocampal Gyrus	Default mode network
				posterior division	
250	65	-12	-19	Right Middle Temporal Gyrus posterior division	Default mode network
251	52	-2	-16	Right Superior Temporal Gyrus anterior division	Default mode network

252	46	16	-30	Right Temporal Pole	Default mode network
253	6	-59	35	Right Precuneous Cortex	Default mode network
254	-26	-40	-8	Left Lingual Gyrus	Default mode network
255	8	-48	31	Right Cingulate Gyrus posterior division	Default mode network
256	-3	-49	13	Left Cingulate Gyrus posterior division	Default mode network
257	-21	-22	-20	Left Parahippocampal Gyrus anterior division	Default mode network
258	-50	-7	-39	Left Inferior Temporal Gyrus anterior division	Default mode network
259	-53	3	-27	Left Middle Temporal Gyrus anterior division	Default mode network
260	11	-54	17	Right Precuneous Cortex	Default mode network
261	-56	-13	-10	Left Middle Temporal Gyrus posterior division	Default mode network
262	-7	-55	27	Left Precuneous Cortex	Default mode network
263	-11	-56	16	Left Precuneous Cortex	Default mode network
264	-44	12	-34	Left Temporal Pole	Default mode network