Automated Sleep Stage Scoring from EEG data using Deep Learning techniques

Master's Thesis im Fach Informatik

vorgelegt
von

Nidhi Joshi

Geboren am 05.02.1993 in Dharwad

Angefertigt am

Lehrstuhl für Mustererkennung (Informatik 5)
Department Informatik
Friedrich-Alexander-Universität Erlangen-Nürnberg.

Betreuer: Dr.Andreas Maier, Dr.Patrick Krauss, Dr.Achim Schilling

Beginn der Arbeit: 10.06.2019

Abgabe der Arbeit: 10.12.2019

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien des Lehrstuhls für Studien- und Diplomarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Erlangen, den 5. Dezember 2019

Übersicht

Der Schlaf ist üblicherweise in 5 Stadien unterteilt: Wake, REM (Schnelle Augenbewegung),N1, N2, N3 auf der Basis von EEG- (Elektroenzephalogramm), EOG- (Elektrookulogramm) und EMG(Elektromyogramm) daten . Es ist üblich, dass ein Somnologe nach den Regeln für die visuelle Klassifizierung die nächtliche Aufzeichnung polysomnografischer Signale manuell durchläuft. Tiefe neuronale Netze, die aus einer Kombination von CNN (Convolutional Neural Network) und LSTM (Long Short-Term Memory) bestehen, können dazu beitragen, diesen langwierigen Prozess in kürzerer Zeit abzuschließen. In dieser Arbeit werden ein traditionelles CNN-LSTM basiertes neuronales Netzwerk mit vielen Schichten sowie ein transferlernbasiertes VGG16 Netzwerk mit den EEG daten von 54 Patienten trainiert und wiederum mit den EEG daten von 14 Patienten evaluiert. Multidimensional Scaling (MDS) - Diagramme werden verwendet, um die internen Darstellungen von Daten anzuzeigen, die zu jeder Klasse auf jeder wichtigen Ebene des neuronalen Netzwerks gehören. Eine als Generalized Discrimination Value (GDV) bezeichnete Metrik wird berechnet und gegen die Schichten des neuronalen Netzwerks aufgetragen, um die Klassentrennung zu messen. Eine weitere wichtige Analyse in der Arbeit ist die Darstellung der Hypnodichte, die im Gegensatz zum Hypnogramm, das nur die wahrscheinlichsten Schlafstadien anzeigt, die Wahrscheinlichkeiten für alle Schlafstadien zu einem bestimmten Zeitpunkt liefert. Die Pearson-Korrelation wird verwendet, um die Abhängigkeit zwischen den Kanälen zu überprüfen..

Abstract

Sleep is commonly divided into 5 stages: Wake, REM (rapid eye movement), N1, N2 and N3 on the basis of EEG(electroencephalogram), EOG(electroocculogram) and EMG (electromyogram) data. Sleep scoring usually involves a somnologist who manually goes through the entire night's recordings of polysomnographic signals according to the visual classification rules. Deep neural networks comprising of a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) can help complete this tedious process in less time. In this work, a traditional CNN-LSTM based neural network containing eleven layers, as well as a transfer learning based VGG16 network are trained with the EEG data of 54 patients, and in turn evaluated with the EEG data of 14 patients. Multidimensional Scaling (MDS) plots are used to view the internal representations of data belonging to each class at every important layer of the neural network. A metric called Generalized Discrimination Value(GDV) is calculated and plotted against certain layers of the network in order to measure class separation. Another important analysis in the thesis is the 'hypno-density' plot, which provides the probabilities for all sleep stages at a given time. Pearson correlation is used to check for inter-channel dependency.

Contents

1 Introduction **1**

1.1 Overview 1

1.2 Biological aspects and Electroencephalogram 2

1.3 How is the EEG recorded for Patients? 3

2 Background **5**

2.1 Rosenblatt Perceptron 5

2.2 Multi-layer Perceptron (MLP) 6

2.3 Artificial Neural Network (ANN) 7

2.4 Convolutional Neural Network (CNN) 8

2.5 VGG16 10

2.6 Recurrent Neural Networks (RNN) 11

2.7 Vanishing Gradient Problem 12

2.8 Long Short-Term Memory Networks (LSTMs) 13

2.9 Hypnogram 14

2.10 Hypnodensity plots 15

2.11 Pearson correlation 17

2.12 Multidimensional Scaling (MDS) 18

2.13 Generalized Discrimination Value (GDV) 19

2.14 Error bars 20

2.15 Cohen's Kappa Coefficient (K) 20

3 Existing approaches of Sleep Stage Classification **21**

3.1 Guideline based approaches 21

3.1.1 Rechtschaffen and Kales (R&K) Rules 21

3.1.2 American Academy of Sleep Medicine (AASM) Rules 22

3.1.3 Genetic Fuzzy Inference System 23
3.2 Machine Learning based approaches 23
3.2.1 Using Bayesian Classifier 24
3.2.2 Using Support Vector Machines (SVM) 24
3.2.3 Using Decision Tree Learning (DT) 25
3.2.4 Decision tree based SVM using ECG data 26
3.2.5 Wavelet transform and Random Forest Classifier 27
3.2.6 Linear Discriminant Analysis (LDA) 27
3.2.7 Hidden Markov Model (HMM) 27
3.2.8 Summary of Machine Learning based approaches 28
3.3 Deep Learning based approaches 29

4 Methodology 31
4.1 CNN-LSTM network trained from scratch 31
4.1.1 Data pre-processing and CNN-LSTM network 31
4.1.2 Hypnodensity Plots 36
4.1.3 Increased temporal resolution 36
4.1.4 Pearson correlation 38
4.2 VGG16 41
4.2.1 Data-preprocessing and VGG16 network design 41
4.2.2 Hypnodensity plots 43
4.2.3 Increased Temporal Resolution 43
4.2.4 Pearson correlation 44
4.2.5 Multidimensional Scaling (MDS) 44
4.2.6 Generalized Discrimination Value (GDV) 44
4.3 Other networks 44

5 Results and Discussion 47
5.1 CNN-LSTM network trained from scratch without dropout 47
5.1.1 Accuracy - Loss curve 47
5.1.2 Confusion matrix 48
5.1.3 Classification Report 49
5.2 CNN-LSTM network trained from scratch with dropout 49
5.2.1 Accuracy - Loss curve 49
5.2.2 Confusion Matrix 50

5.2.3 Classification Report 50
5.2.4 Ground truth versus Predicted labels 51
5.2.5 Results for channel 2 52
5.2.6 Results for channel 3 54
5.2.7 Hypnodensity plots for 30s and 5s 55
5.2.8 Pearson correlation 57
5.2.9 MDS Plots 58
5.2.10 GDV plots with error bars 59

6 Conclusion 63
6.1 Conclusion from different networks 63
6.2 Conclusion from hypnodensity plots 64
6.3 Conclusions from temporal resolutions 65
6.4 Conclusions from Pearson correlations 65
6.5 Conclusions from MDS and GDV plots 66

7 Future Work 67
7.1 Use the left cerebral hemisphere data 67
7.2 Use the remaining 18 EEG placements of 10-20 system 67
7.3 Data preprocessing could be explored more 67
7.4 Give Wavenet a shot 68
7.5 Wavelet transform 68

A Network pseudocode 69

B Hypnodensity plots for the test set 71

C VGG16 Network 75
C.1 Accuracy - Loss curve 75
C.2 Confusion Matrix 75
C.3 Classification Report 76
C.4 Ground truth versus Predicted labels 76
C.5 Hypnodensity plots for non overlap and 5s overlap cases 77
C.6 Pearson Correlation 80
C.7 MDS plots 80
C.8 GDV plot with error bars 80

List of Figures **83**

List of Tables **87**

Bibliography **89**

Chapter 1

Introduction

1.1 Overview

Sleep is a naturally recurring state of the mind and body. It is represented by changing consciousness, slightly slowed down sensory activity, reduced muscle activity and inhibition of nearly all voluntary muscles during rapid eye movement (REM) sleep [Fer08] and relatively reduced interactions with environment [Bet17]. On the contrary, wakefulness is the increased ability to react to stimuli. During a wake state, a human being engages in cognitive and behavioural responses to his environment. Sleep displays very different brain patterns. However, the brain is still active during sleep. On an average, mammals sleep for 2-20 hours every day. Humans require 7.5 hours of sleep on an average per night [Cam84]. Human sleep can be largely divided into two phases: rapid eye movement (REM) sleep and non-rapid eye movement (NREM) sleep. REM comprises 20% and NREM comprises 80% of the total sleep time. NREM is further divided into N1, N2 and N3 based on the lightness and deepness of sleep. From N1 to N3, the deepness of the sleep increases. All these phases alternate cyclically after every 90 minutes. Typically, 3-5 such cycles are completed throughout the night. Figure 1.1 shows an idealized scoring throughout a night, a so-called hypnogram.

For a long time, electroencephalogram (EEG) signals have been used to divide sleep into different sleep stages. The differentiation of these stages of sleep is helpful to the clinicians for the diagnosis of sleep related disorders, psychiatric disorders, cardiovascular diseases, diabetes or obesity, as well as assists the field of sleep research. The manual method of sleep stage scoring where a trained sleep physician goes through an entire night's recordings of the polysomnographic signals and classifies sleep stages by visual inspection is very labour intensive and a time-consuming process. Hence, computerised methods based on applying machine Learning

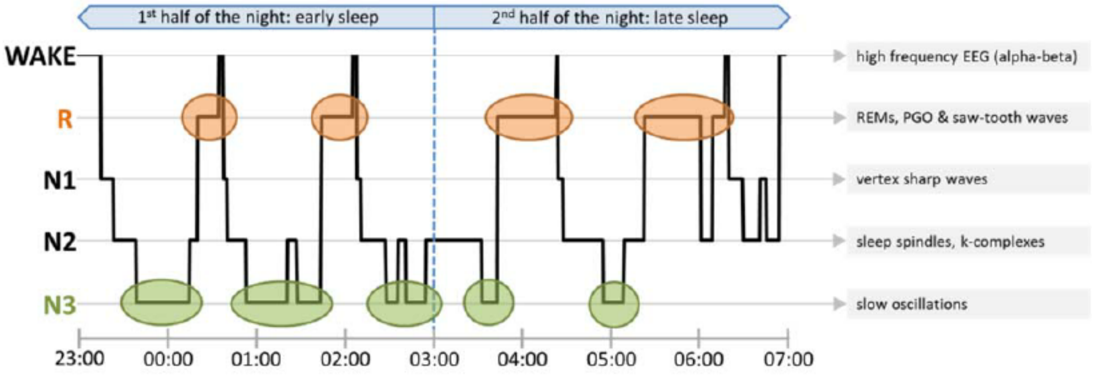


Figure 1.1: Hypnogram for healthy human sleep overnight [Wis14]

techniques on hand crafted features were inspected. These were better than the manual techniques, though still not totally efficient. Therefore, there was a need for automatic sleep stage classification without even having to bother about designing the hand crafted features. This thesis proposes a method to automatically classify sleep stages into Wake, REM, N1, N2 and N3 labels from single channel EEG data using deep neural networks. It further dives into more complex sleep research like analysing the hypnodensity plots and temporal course of sleep stages, and improving the temporal resolutions, visualization of class separation using Multidimensional Scaling (MDS) and Generalized Discrimination Value (GDV) which will all be discussed later in detail.

1.2 Biological aspects and Electroencephalogram

The human brain is a complex dynamic system made up of millions of neurons, interconnected by axons and dendrites. These neurons are responsible for communicating information to and from the brain. Figure 1.2 shows the general structure of a neuron.

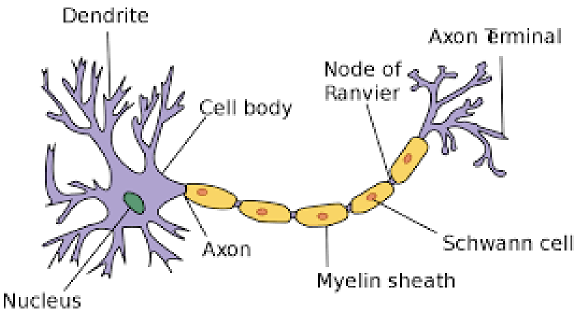


Figure 1.2: Structure of a neuron [Com19]

Brain signals are synchronised electrical pulses obtained from millions of neurons communicating with each other. The brain is divided anatomically into three primary structures: cerebrum, cerebellum and the brain stem. The cerebrum is divided into two hemispheres: left and right hemispheres. The cerebral cortex is divided into four sections called ”lobes”: the frontal lobe, parietal lobe, occipital lobe and temporal lobe [Abo16], as shown in Figure 1.3.

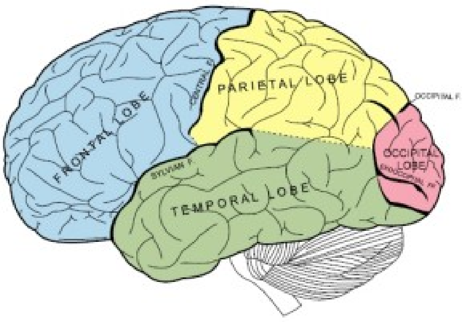


Figure 1.3: Parts of the brain [Wik19a]

Electroencephalogram (EEG) is a non-invasive method to measure the electrical activity of the cerebral cortex.

1.3 How is the EEG recorded for Patients?

Electrodes are fixed on the head of the patient at three different points, to record the brain signals from those points. These three points correspond to the three channels. So, three channel data has been considered in this thesis. Electrode placement for EEG scoring rules follows the international 10-20 system, which assigns a number to each EEG electrode in order to specify the location in the left or right hemisphere, as illustrated in Figure 1.4. The ”10” and ”20” names come from the fact that the actual distances between adjacent electrodes are either 10% or 20% of the total front-back or right-left length of the skull.

The sleep physician (a somnologist) or sleep lab nurse manually goes through the entire night’s EEG recording and rates the sleep stage for each 30 second epoch of the polysomnographic signals according to the visual classification rules. This takes around 30- 90 minutes for a 8 hour recording. Since sleep laboratories have many patients per night, this entire process becomes very time consuming. Furthermore, it is also very expensive. Additionally, there is a huge rate of disagreement (inter-rater disagreement) close to 80% between the scoring obtained from any two somnologists [Dh09]. All this calls for the development of an **Automated Sleep Stage classification system**.

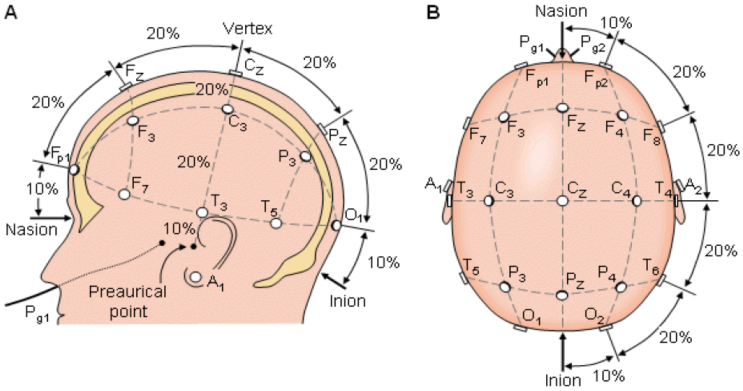


Figure 1.4: Electrode placement system for EEG [Cam12]

The input to this method is a sequence of 30-s EEG epochs. In order to extract the epochs from a given EEG signal, two simple steps are followed:

1. Segmenting the continuous raw single-channel EEG data into a sequence of 30 second epochs and assigning a label to each epoch (i.e., sleep stage) based on the annotation file from a human scorer.

2. Normalizing 30 second EEG epochs such that each one has a zero mean and unit variance (z-scoring).

It is worth mentioning that, these preprocessing steps for the sleep stage extraction are very simple and do not involve any form of filtering or noise removal techniques [Mou19].

Chapter 2

Background

2.1 Rosenblatt Perceptron

Perceptron was the first step to artificial neural network. This allowed the artificial neuron to update its own weights using a simple update rule. It could classify linearly separable data. The main objective of the perceptron update rule was to find the decision boundary (rather the parameters a and a_0) such that after every update, the cardinality of the misclassified data samples (M) becomes smaller and smaller.

The decision rule of perceptron is given by:

$$y* = sign(a^T x + a_0)$$

The objective function which needs to be minimised in this case is:
Minimize:

$$D(\boldsymbol{a}, a_0) = -\sum_{x_i \in M} y_i.(\boldsymbol{a}^T \boldsymbol{x} + a_0)$$

The updated a and a_0 for misclassified samples is calculated as follows:

$$\begin{pmatrix} a_0^{k+1} \\ \boldsymbol{a}^{k+1} \end{pmatrix} = \begin{pmatrix} a_0^{k} \\ \boldsymbol{a}^{k} \end{pmatrix} + \lambda. \begin{pmatrix} y_i \\ y_i.\boldsymbol{x}_i \end{pmatrix}$$

Hence the final decision boundary is given by:

$$F(x) = (\sum_{i \in C} y_i.\boldsymbol{x}_i)^T.\boldsymbol{x} + \sum_{i \in C} y_i = \sum_{i \in C} y_i.\langle \boldsymbol{x}_i, \boldsymbol{x} \rangle + \sum_{i \in C} y_i$$

From the above equation, we know that the parameter a of the decision boundary is a linear combination of feature vectors. A non-linear discrete optimization problem should be solved to calculate the parameters a and $a_0]$. This is achieved using a gradient descent method. The decision boundary after all updates is dependent on initialization conditions.

The algorithm ends when $y_i.(a^T\boldsymbol{x} + a_0)$ which means all data samples fall in their appropriate side of the decision boundary.

The main aim here was to compute a hyperplane (decision boundary) such that the distance of misclassified feature vectors from the decision boundary is minimized.

2.2 Multi-layer Perceptron (MLP)

A Multi-layer Perceptron, as the name suggests consists of several layers of perceptrons. Its structure is the closest to an artificial neural network. It consists of at least one input layer, one hidden layer and one output layer, as shown in Figure 2.1. The hidden layer nodes and the output layer nodes use a non-linear activation function. It is a feed-forward neural network that is trained by backpropagation algorithm. The major advantage of MLP in comparison to the Rosenblatt Perceptron is its ability to classify non-linearly separable data.

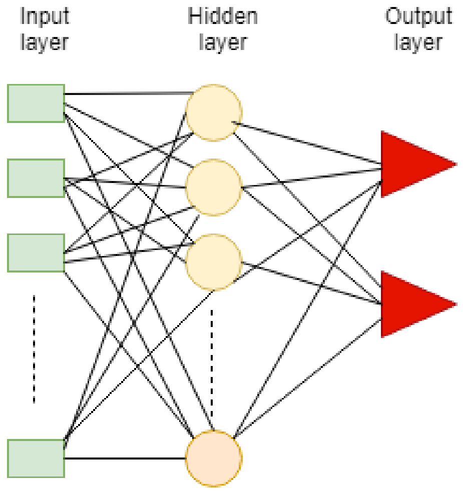


Figure 2.1: Vanilla Neural Network

2.3 Artificial Neural Network (ANN)

An Artificial Neural Network mimics the temporal lobe of the cerebrum. It is designed in such a way that it can replicate a human brain and a human's ability to learn. Like MLP, it has one input layer, several hidden layers and one output layer. The input layer is supplied with the data, the hidden layers constitute various representations of the data and the output layer gives predictions. Activation functions are applied to the input to control a specific behaviour of the output. ANN learns the optimal weight values after several iterations.

It is a good practise to initialize weights following a standard normal distribution. Training the ANN goes on until the specified number of epochs are completed. The input data is propagated inside the neural network through a process called forward propagation. We compare that prediction to the target (the real value which we have because we are dealing with the training set), and we compute the loss error between the prediction and the target.

This loss error is back-propagated inside the neural network. Through Gradient Descent or Stochastic Gradient Descent, the weights are updated in the directions that reduce the loss error the most (since in fact the loss error is a cost function of the weights). We thus get new weights, ready to make a new prediction at the next iteration. Then the same whole process repeats until the loss error stops reducing (early stopping) or until the training reaches the last epoch.

Activation functions apply non-linear complex properties to the network so that the network does a good job in predicting the output. There are a variety of activation functions which can be chosen by the programmer while designing a model. It is as simple as passing an argument to a function. Figure 2.2 illustrates the activation functions in detail.

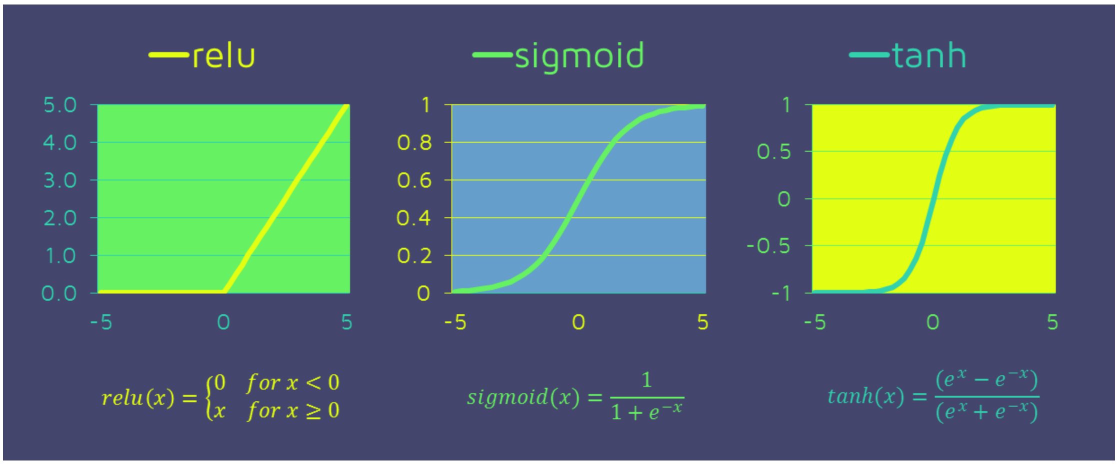


Figure 2.2: Most common activation functions [xA19]

It is a common practise to have one training dataset, one validation dataset and one completely unseen test dataset. The model is trained on the training dataset and validated on the validation dataset. These two processes can be simultaneously performed. A good model would exhibit the following behaviour with advancement in epochs: the training accuracy will increase, training loss will decrease, validation accuracy will increase, and validation loss will decrease. We often observe that the training accuracy will be increasing, but suddenly the validation loss also starts increasing. This is an indication that the model has begun to memorize the training data and is hence unable to perform well on validation data. K-fold cross validation is a good way to make sure that the model has been trained correctly. In this method, the training dataset is divided into 'K' equal parts. 'K-1' parts are used as training data and the remaining 1 part is used for validation. Each of the 'K' parts gets to be a part of the validation dataset. The loss function is a numerical estimate of how wrong our neural network predictions are. If we reduce the loss function, it means we are reducing the mistakes made by the neural network and in-turn making the predictions more accurate. Binary cross entropy and categorical cross entropy are the two most commonly used loss functions for two-class and multi-class classification problems respectively.

The test set is kept completely untouched and unmodified. Once the validation works well, the model is used to make predictions on the test dataset. True values and the values predicted by the model are then compared. A confusion matrix, classification report (recall, precision and F-1 score) and area under the curve(AUC) are some popular metrics used for evaluating the model.

ANNs can be used widely for a variety of applications such as supervised learning (CNNs classifying images), unsupervised learning (Boltzmann Machines, AutoEncoders, GANs, DC-GANS, VAE, SOMs, etc.) and Reinforcement learning (Deep Convolutional Q-Learning that plays videogames from pixel input, AlphaGO, etc.).

2.4 Convolutional Neural Network (CNN)

CNN mimics the occipital lobe of the cerebrum which deals with vision. CNN is a type of ANN which is commonly used with image data. This is because the pre-processing required for image data using CNN is very less as compared to any other network. CNNs can best extract image related features like edges and corners in the initial layers, and other high level features in further layers. A CNN can capture the spatial and temporal dependencies in an image through the application of relevant filters called convolution kernels. Convolution and pooling are the two most important steps in a CNN. Convolution filter or kernel extracts features in every layer it has been used in. Commonly, this is given as an input to the pooling layer which down samples the feature information provided by its preceding convolution layer, yet still maintaining the context information. This helps in reducing the computational costs.

Figure 2.3 gives an intuition of how a dog image would be learnt by CNN. Figure 2.4 illustrates the convolution operation in detail.

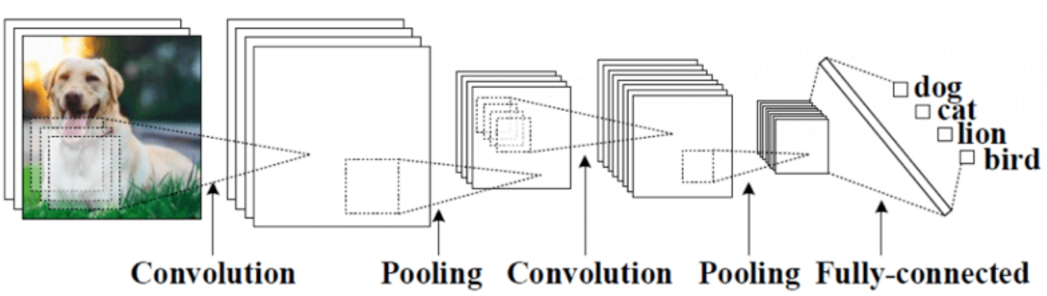


Figure 2.3: Working of a typical CNN [Des16]

In fully connected/ Dense layers unlike convolutional layers, every neuron of a layer is connected to every neuron of the next layer. Fully connected layers are used as end layers because they can encode final global high-level features of the image instead of extracting just the limited local space features. A sigmoid/softmax activation function is used with Dense layers for classification. Sigmoid is used for binary (two class) classification while softmax is used for multi-class classification. This returns the confidence/ probabilities for each class label in the final layers. CNNs can be trained alone or end-to-end with some other networks like LSTMs.

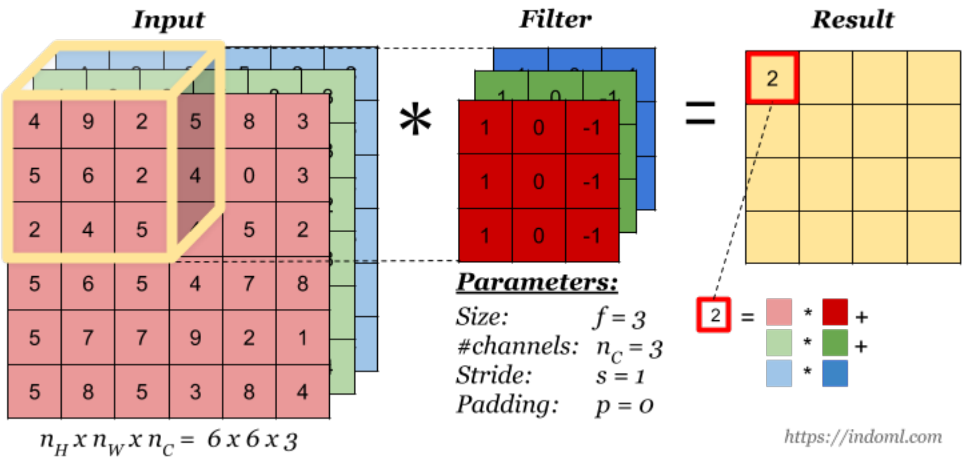


Figure 2.4: Convolution Operation on Volume [Gas18]

Pooling is of two types: (1) max-pooling which returns the maximum value found in the image covered by the filter kernel; (2) average-pooling which returns average of all the values found in the image covered by the filter kernel.

Transfer learning is another major useful addition to the CNNs. It allows the network to learn from some pre-trained weights downloaded from ImageNet. Thus, the CNN would not be trained from scratch in this case. Various transfer learning networks exist such as VggNet, Resnet, Inception, Xception, etc.

2.5 VGG16

VGG stands for ”Very Deep Convolutional Networks for Large-Scale Image Recognition”. Vgg16 model gives 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes [Qas18].

The images provided to this network have to be reshaped to a fixed resolution of 256*256. The Cov1d layer input is of fixed size: 224 X 224 of an RGB image. The image is inputed to a stack of convolutional layers, along with filters with a very small receptive field of 3X3. It also utilizes a 1X1 convolution filter in one of the configurations. The convolution stride is fixed to 1 pixel, and the spatial padding of convolutional layer input is such that the spatial resolution is preserved even after a convolution operation. There are five max-pooling layers, following some convolutional layers. There is a 2X2 max-pooling pixel window, with stride 2. Figure 2.5 illustrates the architecture of this network.

Three Dense layers follow a stack of convolutional layers. First two have 4096 channels, the third has 1000 channels due to 1000 class labels. The final layer is the soft-max layer.

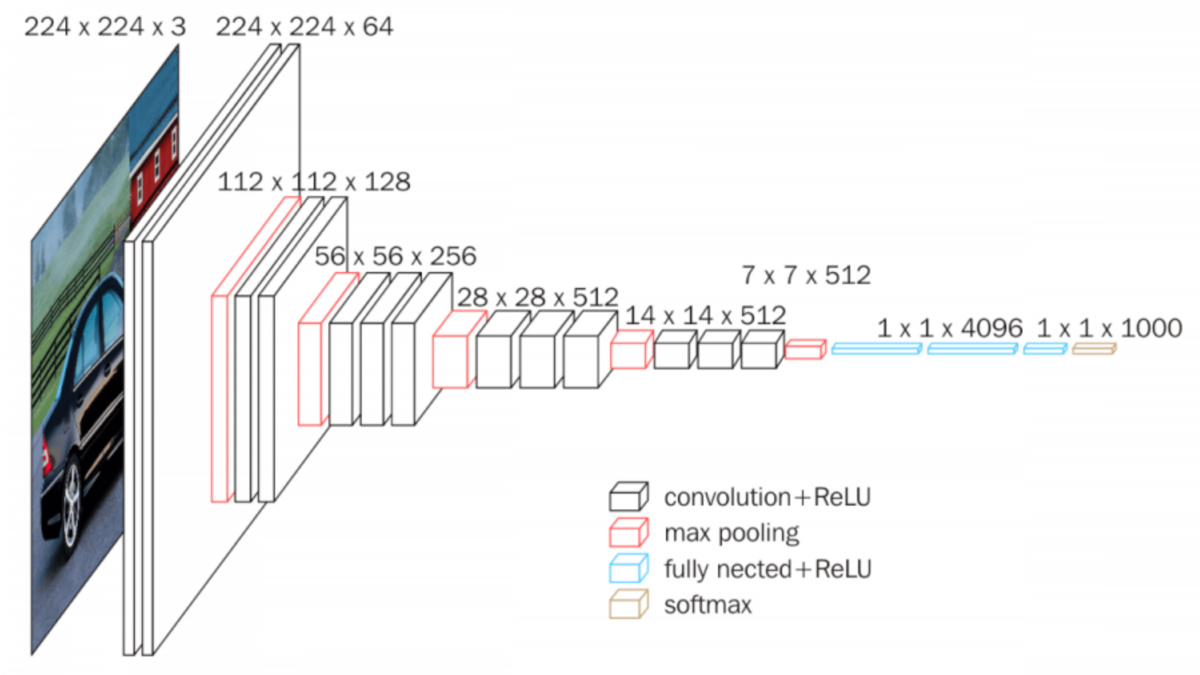


Figure 2.5: Vgg16 layer architecture [uH18]

2.6 Recurrent Neural Networks (RNN)

RNN mimics the Frontal lobe of the cerebrum. It deals with short term memory operations. RNN also has input, hidden and output layers. They are networks with loops in them, allowing information to persist. A recurrent neural network has multiple copies of the same network, each passing a message to a successor. For a better representation of the RNN, these layers are squashed and stacked one behind the other as shown in Figure 2.6.

This chain-like nature reveals that RNNs are closely related to sequential data learning. RNNs have a variety of applications: speech recognition, language modeling, translation and image captioning.

Types of RNNs based on input and outputs are as follows:

1. **One to Many**

The input is one image (Figure 2.7) and output is description of the image by the system. The CNN model extracts features and the RNN has memory of various features in the image. Hence, this model describes the image as: ’A cheetah sprinting on green grass on a sunny day’.

2. **Many to One**

This is commonly observed in sentiment analysis. The input is a set of words such as ’Apple

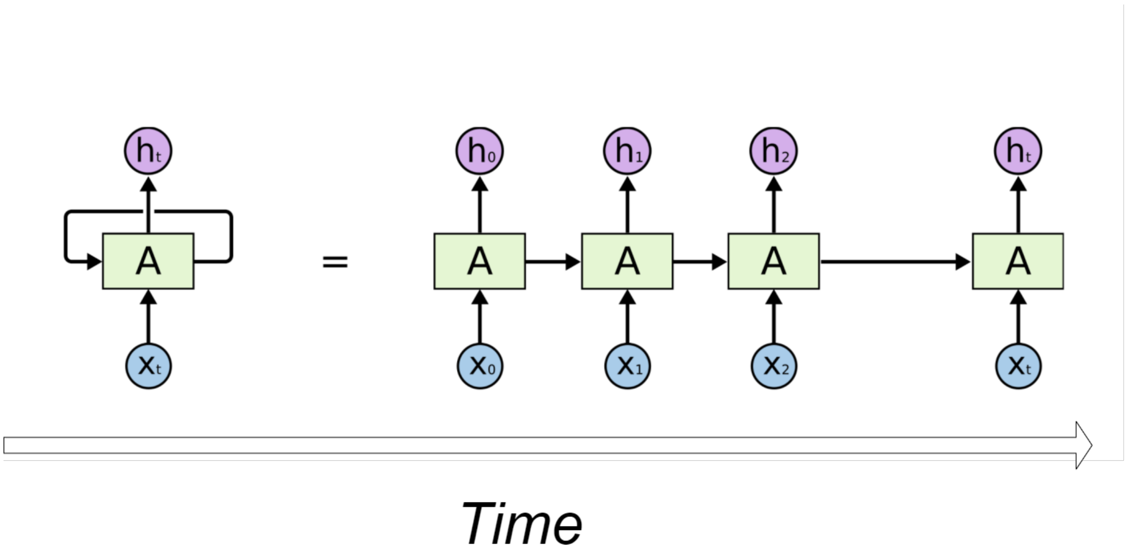


Figure 2.6: RNN architecture [Ola15]

Figure 2.7: One to many RNN [Wik19b]

products are awesome!', which would return a positive sentiment as the output.

3. **Many to Many**

Used in applications like translating a sentence from one language to another and describing a video which is a series of image frames.

2.7 Vanishing Gradient Problem

The loss propagates backwards from the output to the input layer, propagating the input error gradient. With deeper neural networks, issues that can arise from back propagation are vanishing and exploding gradients. As we go back to the lower layers, the gradient often get smaller and smaller. Eventually, the weights never change at deeper layers. As a result, shallow weights in the network get trained but deeper weights remain untrained. This is referred to as the 'Vanishing Gradient' problem.

The three techniques that can solve Vanishing Gradient problem are: 1. Weight Initialization 2. Echo State Networks 3. Long Short-Term Memory Networks (LSTMs)

2.8 Long Short-Term Memory Networks (LSTMs)

In Figure 2.8, the straight line on the top that passes through all the blocks has to go through only two simple point-wise operations: 'X' means some subtraction and '+' means some addition.

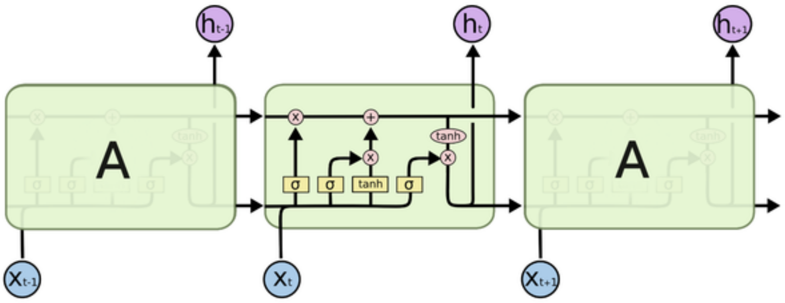


Figure 2.8: LSTM architecture [Ola15]

The cell state, hidden state, input, output, next cell state and the next hidden state (also called output) are all vectors. Cell state and the Next cell state refer to the long and short term memory respectively. Every gate/valve extracts and directs some information onto the next state.

In the LSTM diagram (Figure 2.9), the top line is called the memory pipeline as discussed before. The input is a vector of the old memory. The first cross 'X' it passes through, called the forget valve, is an element-wise multiplication operation. So if the old memory (Cell state) is multiplied with a vector that is close to 0, it will get erased. On the contrary, forget valve equals 1 implies that we want to retain the old memory.

The next operation is the memory flow through the '+' operator. This is a piece-wise summation operator. New memory and the old memory will be added piece-wise by this operation. How much of the new memory is added to the old memory is controlled by another valve, the 'X' below the '+' sign.

Now, the old memory has been changed to the new memory. The input of the neural network is the output of the previous LSTM block, and the input for the current LSTM block is the memory of

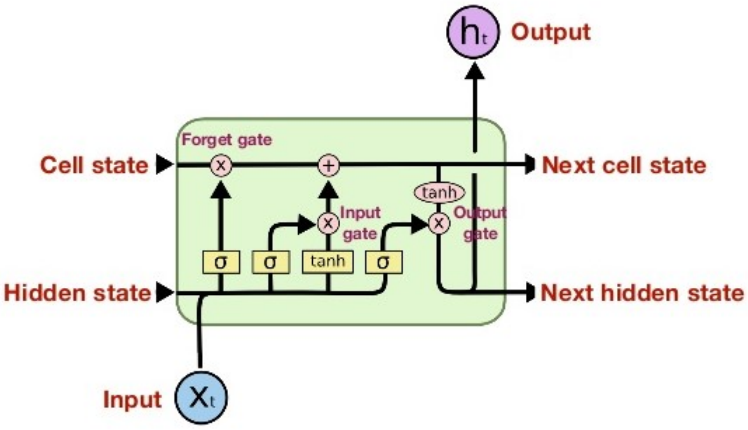


Figure 2.9: LSTM architecture compressed version [Ola15]

the previous block. This neural network has a sigmoid function as activation, and its output vector is the forget valve, which will be applied to the old memory via element-wise multiplication.

The second valve is called the new memory valve. This takes the same inputs as the forget valve. This valve controls how much the new memory should influence the old memory. The new memory itself, is generated by another neural network which uses tanh as the activation function. The output of this network is element-wise multiplied with the new memory valve, and added to the old memory to form the new memory.

Eventually, we need to get the output of this LSTM unit. This step has an output valve that is controlled by the new memory, the previous output and the input. This valve controls how much new memory should go to the next LSTM unit. Hence, we have got the final output of this LSTM block [Yan16].

Gated Recurrent Unit (GRU) is a variation of LSTM. It combines the forget and input gates into a single gate called 'update gate'. It also merges the cell state and hidden state. There is no output gate. GRU is simpler than standard LSTM models, and has become popular. Gated RNN is also a variant of LSTM introduced in 2015.

2.9 Hypnogram

Hypnogram is a plot of stages of sleep versus time. It gives us information about which sleep stage has been predicted for a particular time interval sleep data, as well as details of irregular sleeping patterns associated with sleep disorders. Typically, a hypnogram is recorded per 30 second epoch (since underlying EEG data is also recorded for 30 second epoch).

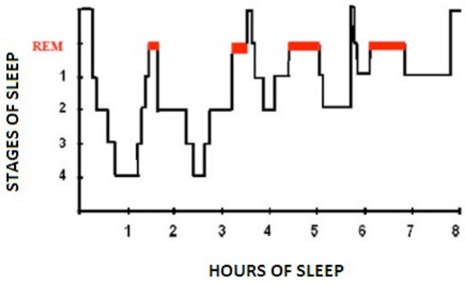


Figure 2.10: Typical hypnogram recorded overnight [Gor11]

Figure 2.10 illustrates an overnight recorded hypnogram. For now it is enough to know that REM, 1,2,3 and 4 refer to the sleep stages. Detailed plots for our work will be provided in the next sections.

2.10 Hypnodensity plots

Hypnodensity plot is a probability distribution conveying more information than classical hypnograms [Ste18]. It is a hypnogram that deals with not only one sleep stage label, but contains the probability values of occurrence of each sleep stage label, thereby allowing more information about other candidate sleep stages also to be conveyed. This is typically useful since a human scorer is also not capable of giving accurate information about other sleep stages for a particular epoch.

Hence, the model results are displayed more informatively as hypnodensity graphs, which represent not only the discrete singular sleep stage output, but also the probability of occurrence of each sleep state for each 30 second epoch (Figure 2.11).

Hypnodensity graphs can be created by various methods: First, automatically from raw sleep data, and check how the resulting interpretable features can be used to generate a diagnosis probability for sleep related disorders. Second, by classifying the sleep stages directly from the neural network by optimizing the performance not only for sleep staging, but also for direct diagnosis by plotting the softmax layer output with respect to time, thereby creating a multitask classifier. The second approach has been used in this thesis.

We have plotted hypnodensity plots for all our models and each of our patients in this thesis work. They have been very helpful in analyzing and validating our results with the biological consequences for each of our channel data that comes from electrodes placed on different points

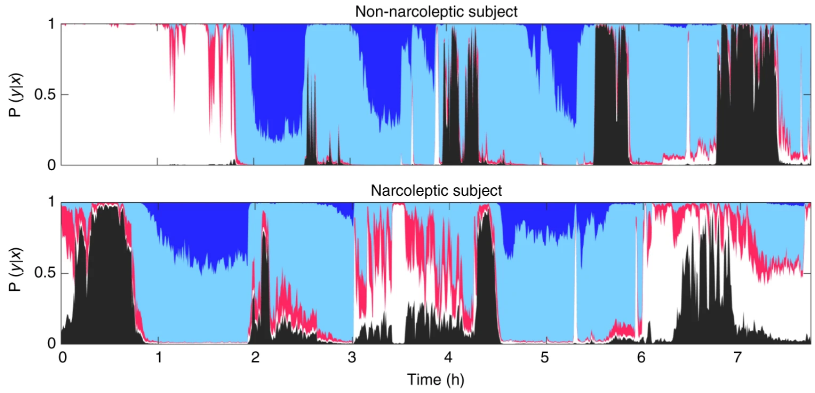


Figure 2.11: Typical hypnodensity plot obtained for nacroleptic and non-nacroleptic subject [Ste18]

on the patient's head.

2.11 Pearson correlation

Pearson correlation is a measure of the linear correlation between two variables X and Y.

Its value can vary between -1 and +1. -1 stands for anti-correlation, 0 stands for no correlation and +1 stands for complete positive correlation. All these correlations are linear in nature.

Correlations are never lower than -1. A correlation of -1 indicates that the data points in a scatter plot lie exactly on a straight descending line; the two variables are perfectly negatively linearly related.

A correlation of 0 means that the two variables don't have any linear relation. However, some non linear relation may exist between them.

Correlation coefficients are never higher than 1. A correlation coefficient of 1 means that two variables are perfectly positively linearly related; the dots in a scatter plot lie exactly on a straight ascending line. Figure 2.12 illustrates these concepts.

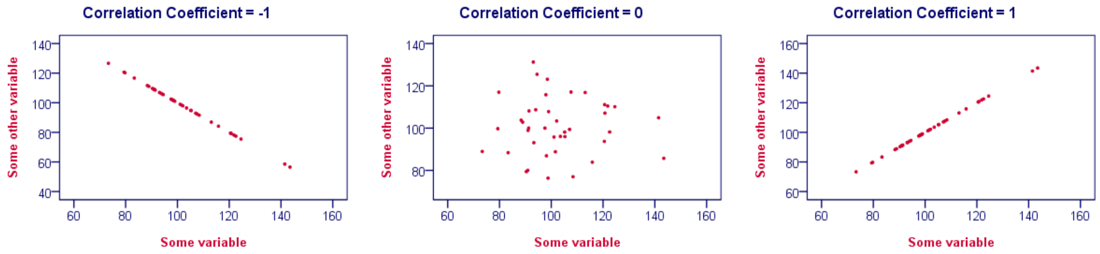


Figure 2.12: Pearson Correlation coefficient significance [Tut19]

According to the formula, Pearson correlation coefficient r_{xy} is the covariance of the two variables X and Y divided by the product of their standard deviations.

$$r_{xy} = \frac{cov(X,Y)}{\delta_X \delta_Y}$$

where:

'cov' is the covariance

'δ_X' is the standard deviation of X

'δ_Y' is the standard deviation of Y

Formally, r_{xy} is given by:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where:

’n’ is the sample size

x_i and y_i are the individual sample points indexed with i

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i.$

Please note that in our case, X and Y are not two scalars but 5 dimensional vectors.

2.12 Multidimensional Scaling (MDS)

Multidimensional scaling (MDS) is a method of visualizing the level of similarity of each of the classes of a dataset. MDS is used to translate information about the pairwise ’distances’ among a set of n classes into a configuration of n points mapped into an abstract Cartesian space.

Figure 2.13 shows class separations for several data points separated by different colors.

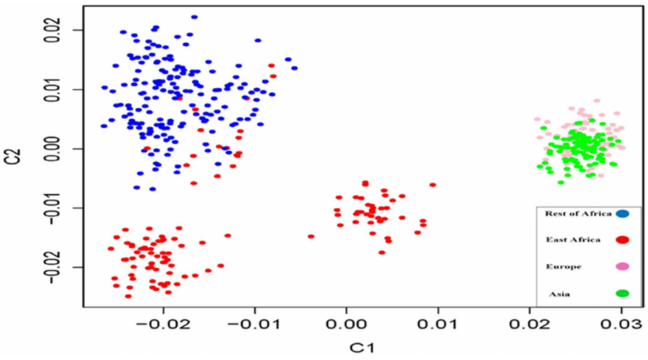


Figure 2.13: Typical MDS plot [Elh14]

MDS is a non-linear dimensionality reduction technique. It is a common practice to visualize the class separations on a two dimensional plane, since it is more intuitive. However, more the dimensions allowed, better is the separation. A researcher has to specify the desired dimensionality. Factor analysis is a powerful technique to choose the right number of dimensions onto which the high dimensional data can be projected. With MDS, best results are achieved if the visualization shows high inter-class/cluster separation with a minimum intra-class/cluster separation.

In our work, we plot MDS after every important neural network layer to visualize how well the classes separate at different stages of the neural network.

2.13 Generalized Discrimination Value (GDV)

GDV measures how well different data classes separate within each given network layer in a non-invasive manner [Sch18]. GDV is capable of indicating self organized data transformations during the process of neural network training. GDV proves to be useful in accessing not just supervised learning techniques but also the unsupervised training with the Contrastive Divergence method. GDV can be used in optimizing neural network performance by hyper-parameter tuning such as varying the width and depth of the network. A good GDV value would mean increased inter-class separation and decreased intra-class separation.

We consider 'N' points $X_{n=1,...,N} = (X_{n,1}, X_{n,2}, \ldots ,_{n,D})$ distributed within a D-dimensional space. A label l_n assigns each point to one of L distinct classes $C_{l=1,2,...,L}$. Each dimension is separately z-scored and multiplied by $\frac{1}{2}$ to achieve invariance against scaling and translation.

$$s_{n,d} = \frac{1}{2}.\frac{x_{n,d} - \mu_d}{\delta_d}$$

.

Here, $\mu_d = \frac{1}{N}.\sum_{n=1}^{N} x_{n,d}$ denotes the mean and $\sigma_d = \sqrt{\frac{1}{N}.\sum_{n=1}^{N}(x_{n,d} - \mu_d)^2}$ is the standard deviation of dimension d. Based on the rescaled data points $s_n = (s_{n,1}, \ldots s_{n,D})$, we calculate the mean intra-class distances:

$$\bar{d}(C_l) = \frac{2}{N_l.(N_l - 1)}.\sum_{i=1}^{N_l - 1}\sum_{j=i+1}^{N_l} d(s_l^i, s_l^j)$$

and the mean inter-class distances:

$$\bar{d}(C_l, C_m) = \frac{1}{N_l.N_m}.\sum_{i=1}^{N_l - 1}\sum_{j=i+1}^{N_l} d(s_l^i, s_l^j)$$

Here, N_k is the number of points in class k, and s_i^k is the i^th point of class k. The quantity d(a,b) is the Euclidean distance between a and b. Finally, the GDV value is calculated from the mean intra-class and inter-class distances as follows :

$$GDV = \frac{1}{\sqrt{D}}.\left[\frac{1}{L}.\sum_{l=1}^{L}\bar{d}(C_l) - \frac{2}{L.(L-1)}.\sum_{l=1}^{L-1}\sum_{m=l+1}^{L} d(C_l, C_m)\right]$$

Ideal value for GDV is -1. It is 0 for randomly shuffled labels.

2.14 Error bars

Error bars are graphical representations of the variability of data and are used on graphs to indicate the error or uncertainty in a reported measurement. They give a general idea of how precise a measurement is, or in other words, how far from the reported value the true (error free) value might be.

In our work, we have plotted average GDV plots along with the error bars.

2.15 Cohen's Kappa Coefficient (K)

Cohen's Kappa Coefficient is a statistical measure of inter-rater reliability for categorical items. It tells as how much do any two human sleep scoring experts agree with each other's scores. This is an important measure in evaluating any sleep stage scoring algorithm along with the F1 score and Accuracy.

Chapter 3

Existing approaches of Sleep Stage Classification

The upcoming section throws some light on all the approaches in the direction of Sleep Stage Classification, accomplished using different datasets (like electroencephalogram, electrocardiogram, electrooculogram, electromyogram etc) and different techniques dating back to history. For simplicity, this section has been organized into three major sub-sections:

1. Guideline based approaches
2. Machine Learning based approaches
3. Deep Learning based approaches

The third approach based on Deep Learning techniques has been implemented in this thesis.

3.1 Guideline based approaches

3.1.1 Rechtschaffen and Kales (R&K) Rules

For the initial 40 years, only the manual sleep scoring rules by Rechtschaffen and Kales (R&K) dominated sleep research. According to the R&K rules, sleep recordings are divided into seven stages : wake, stage 1, stage 2, stage 3, stage 4, stage REM, and movement time. Though this standard is widely used, the rules of Rechtschaffen and Kales have also been unsuccessful for leaving plenty of opportunities for subjective interpretation, which led to a great variability in the visual evaluation of sleep stages. Moreover, the R&K rules were developed for young healthy adults' data and do not apply to elderly patients [Mos09].

3.1.2 American Academy of Sleep Medicine (AASM) Rules

R&K rules were modified by the American Academy of Sleep Medicine (AASM) [Ibe07], and a new guideline for terminology, recording method, and scoring rules for sleep-related phenomena was developed (Figure 3.1). The AASM manual revolves around digital analysis and parameters reporting, visual scoring, arousal, respiratory and cardiac events, movements and pediatric scoring. There were the following terminology changes from the R&K system: in the AASM classification, sleep stages S1 to S4 are referred to as N1, N2, and N3, with N3 denoting slow wave sleep (SWS, R&K stages S3 + S4); stage REM is referred to as stage R. According to the AASM manual, a minimum of 3 EEG derivations, sampling activity from the frontal, central, and occipital regions, have to be recorded. The recommended derivations are F4-M1, C4-M1, and O2-M1 (right-sided active electrodes and a reference over the left mastoid, rather than the ear). This new manual also defines the sleep-wake transition, K-complexes, sleep spindles, slow wave sleep, REM sleep, as well as arousals and major body movements. To summarize, the major changes of the new manual comprise the merging of stages 3 and 4 into N3, EEG derivations introduction, the elimination of ”movement time”, the simplification of many context rules, as well as the recommendation of sampling rates and filter settings for polysomnographic (PSG) reporting and for user interfaces of computer-assisted sleep analysis [Mos09].

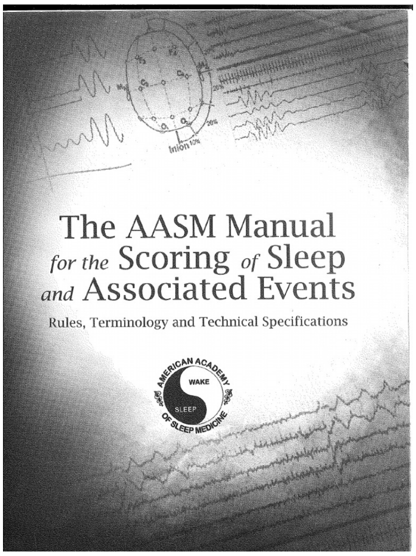

Inion
The AASM Manual
for the Scoring of Sleep
and Associated Events
Rules, Terminology and Technical Specifications
AMERICAN ACADEMY OF SLEEP MEDICINE
WAKE
SLEEP

Figure 3.1: AASM Manual [Ibe07]

While light and deep sleep increased (S1 vs. N1 [+10.6 min, (+2.8%)]: P <0.01; S3+S4 vs. N3 [+9.1 min (+2.4%)]: P <0.01), stage 2 sleep decreased significantly according to AASM rules (S2 vs. N2 [-20.5 min, (-4.9%)]: P <0.01). Moreover, wake after sleep onset was significantly prolonged by approximately 4 minutes ($P < 0.01$) according to the AASM standard. Interestingly, the effects on stage REM were age-dependent (intercept at 20 years: -7.5 min; slope: 1.6 min for 10-year age increase). No effects of sex and diagnosis were observed.

3.1.3 Genetic Fuzzy Inference System

Another approach was based on some straightforward fixed set of fuzzy rules that classified sleep stages. This genetic fuzzy inference system was based on expert knowledge. Eight features, including temporal and spectrum analyses of the EEG and EMG signals, were utilized as the input variables. The fuzzy rules and the fuzzy sets were constructed based on expert knowledge and the distributions of feature values at different sleep stages. The overall inter-rater agreement and kappa coefficient of this integrated system applied to Polysomnography(PSG) data from 8 subjects with good sleep efficiency, 8 subjects with poor sleep efficiency and 8 subjects with insomnia were 86.44% and 0.81 respectively [Lia15].

3.2 Machine Learning based approaches

Hand crafted features defined by the expert are fed to Machine Learning algorithms. The classifier tries to find an optimal mapping from the input features to the class labels. Typical features consist of the spectral bands of brain activity or mean power of the signal. Features can belong to one of the following categories:

1. Spectral features: These are the frequency components of the EEG signal obtained from Fourier/ Wavelet transform. Delta (0.5-4 Hz),theta (4-8 Hz), alpha (8-12 Hz), beta (12-40 Hz) and gamma (40-100 Hz) and sometimes an additional band for sleep spindles (12-16 Hz) account for the spectral features.

2. Temporal features: Indicate change in the brain activations with time. For example, the onset of stage N2 depends on whether K complex or sleep spindles occurs early or in the last half of the previous epoch or change in peak amplitude as an indicator of SWS or relative change of frequencies.

3. Statistical features: Using statistical signal parameters like mean, median, standard deviation, kurtosis and skewness of the signal.

4. Perceptual features: Assigned definite values based on visual inspection of signal structure.

5. Fractal and Entropy features: Since the EEG signals behave irregularly, fractal and entropy methods are suitable to measure the amount of the randomness inside the signal. As the entropy of a signal is increased, its information content is inclined accordingly. In other words, as a signal

behaves more irregularly, fractal dimension and entropy of the signal are increased. When a sleep stage is changed, the efficient frequency bandwidth along with the signal amplitude vary accordingly. Therefore, it is expected that fractal and entropy measures can finely describe each sleep stage [Boo17].

6. Complex features: Based on how each stage is represented in the system memory.

3.2.1 Using Bayesian Classifier

Rytkönen and Kirsi-Marja described a simple MATLAB-based method for automated scoring of rat and mouse sleep using the naive Bayes classifier. This method resulted in an overall auto-rater agreement of 93%, comparable to an inter-rater agreement between two human scorers (92%), with high sensitivity and specificity values for wake (94% and 96%), NREM sleep (94% and 97%) and REM sleep (89% and 97%) states. In addition to baseline sleep-wake conditions, the performance of the naive Bayes classifier was assessed in sleep deprivation and drug infusion experiments, as well as in aged and transgenic animals using multiple EEG derivations. 24-hour recordings from 30 different animals were used, with approximately 5% of the data manually scored as training data for the classification algorithm [Ryt11].

3.2.2 Using Support Vector Machines (SVM)

Shelly Crisler in his Journal of Neuroscience methods [Cri08] mentions the scoring system for rat sleep stages based on electrocorticographic (ECoG) signals recorded from electrodes implanted epidurally and electromyographic (EMG) signals from the temporalis or nuchal muscle. Numerous spectral and frequency domain features combined with statistical analysis were extracted. A hand crafted feature based sleep scoring system was developed using a support vector machine (SVM) to discriminate between waking, nonrapid eye movement sleep, and paradoxical sleep. The SVM kernel function was chosen to be a Gaussian radial basis function and the kernel parameters were varied to examine the effectiveness of optimization methods. Tests indicated that a common set of features could be chosen which resulted in an overall agreement between the automated scores and the expert consensus of greater than 96%.

In another recent approach by Alickovic [Ali18], a robust system, consisting of three modules, was proposed for automated classification of sleep stages from the single-channel electroencephalogram (EEG). In the first module, signals taken from Pz- Oz electrode were denoised using multiscale Principal Component Analysis (PCA). In the second module, the most informative features were extracted using Discrete Wavelet Transform (DWT), and then, statistical values of

DWT sub-bands were calculated. In the third module, extracted features were fed into an ensemble classifier, Rotational Support Vector machine (RotSVM). The proposed classifier combined advantages of the PCA and SVM to improve classification performances of the traditional Support Vector Machine (SVM). The sensitivity and accuracy values across all subjects were 84.46% and 91.1%, respectively, for the five-stage sleep classification with a Cohen's kappa coefficient of 0.88. Obtained classification performance results indicate that it is possible to have an efficient sleep monitoring system with a single-channel EEG which can be used effectively in medical and home-care applications.

3.2.3 Using Decision Tree Learning (DT)

Hanaoka in 2001 published a waveform recognition method that extracted characteristic features from waveforms and a method of sleep stage scoring using decision tree learning [Han01]. In the waveform recognition, the characteristic parameters were extracted in a form that resembled the description for R&K rules. First the characteristics of EEG, EOG and EMG were compared with characteristic features of alpha waves, delta waves, sleep spindles, K-complexes and REMs. Then, several features that are necessary for sleep stage scoring were extracted. The extracted features were transformed into few discrete variables using canonical discriminant analysis and the discretization method based on a random walk, and then a committee that consists of several small decision trees was formed from a small number of training instances. Final sleep stages were decided by a majority voting of the committee. Learning and classification were carried out five times by cross-validation, the average learning time was 39.54 sec and the average classification time was 3.23 sec. With decision tree classification approach, the sleep stage scoring was made faster with over 70% correct answers, which coincided with the percentage obtained by specialists.

Similar work using decision trees was also carried by Martin [Mar72] with EOG and EEG data. Their results indicated that sleep scorers agreed among themselves approximately 89% of the time. The overall percent agreement between the computerized recognition system and the consensus formed by a majority vote of the three human judges was 7% less than the average pairwise agreement among human scorers. The consensus scoring showed slightly better agreement with the automatic scoring system than the individual scorer versus computer comparison. In addition to this, the conventional sleep stages were replaced with a sleep scoring system where the stages N1 and N2 were combined and this yielded a better performance for both the human and computerised scoring systems. The sleep scorers agreed among themselves approximately 91% of the time. The computer's average percent agreement was only 6% lower than the human agreement.

A decision tree based algorithm was also used by Louis [Lou04] in 2004 for the real time scoring of sleepâwake state in Wistar rats. EEG amplitude (μVrms) was measured in the following frequency bands: delta (δ; 1.5 to 6 Hz), theta (θ; 6 to 10 Hz), alpha (α; 10.5 to 15 Hz), beta (β; 22 to 30 Hz), and gamma (γ; 35 to 45 Hz). Electromyographic (EMG) signals (μVrms) were recorded from the levator auris longus (neck) muscle, as this yielded a significantly higher algorithm accuracy than the spinodeltoid (shoulder) or temporalis (head) muscle EMGs (ANOVA; P=0.009). Data were obtained using either tethers (n=10) or telemetry (n=4). They developed a simple three-step algorithm that categorizes behavioural state as wake, non-rapid eye movement (NREM) sleep, rapid eye movement (REM) sleep, based on spectral features characterising the thresholds set during a manually-scored 90 minute preliminary recording. The behavioural state was assigned in 5 second epochs. EMG amplitude and ratios of EEG frequency band amplitudes were measured and compared with empirical thresholds in each animal. The algorithm was as follows:

STEP 1: EMG amplitude greater than threshold? Yes: ”active” wake, no: sleep or ”quiet” wake.

STEP 2: EEG amplitude ratio ($\delta \times \alpha$)/($\beta \times \gamma$) greater than threshold? Yes: NREM, no: REM or ”quiet” wake.

STEP 3: EEG amplitude ratio θ2/($\delta \times \alpha$) greater than threshold? Yes: REM, no: ”quiet” wake.

The overall accuracy in discriminating wake and sleep (NREM and REM combined) using step one alone was found to be 90.1%. The overall accuracy using the first two steps was 87.5% in scoring wake, NREM and REM sleep. The overall accuracy in scoring wake, NREM and REM sleep was 87.9%. All accuracies were derived from comparisons with unequivocally scored epochs from four 90-min recordings as defined by an experienced human rater. The algorithms were as reliable as the agreement between the three human scorers (88%).

3.2.4 Decision tree based SVM using ECG data

Wang’s strategy [Wan12] consisted of a sequential forward selection (SFS) feature selection method and a Decision Tree based Support Vector Machines (DTB-SVM) classifier for discriminating three types of sleep based on Electrocardiogram (ECG) signals. Each 5-minute epoch of ECG signal data collected during sleep was used to generate 24 features using Heart Rate Variability (HRV) analysis. A SFS feature selection method was then employed to determine which significant features should be selected to improve the classification accuracy. A DTB-SVM was then trained using selected features in order to discriminate between three sleep stages, including pre-sleep wakefulness, NREM sleep and REM sleep. The average classification accuracy of the proposed strategy was 73.51%. Their experimental results demonstrated that the proposed strategy provided moderate accuracy for detecting sleep stages in sleep apnea patients.

3.2.5 Wavelet transform and Random Forest Classifier

According to Boostani's [Boo17] work on Sleep Stage classification, the entropy of wavelet coefficients was chosen as a hand crafted feature and the random forest classifier was chosen for classification. They applied it to single-channel EEG of 20 subjects containing equal number of healthy and unhealthy individuals. This was one of the few unique studies where healthy and patient datasets were analysed separately. This study gave 88.66% accuracy on healthy subjects and 66.96% on patient group. Although the accuracy on healthy subjects was remarkable, the results for the patient group by the quantitative methods were not promising yet. It was believed that adopting non-matched sleep EEG features from other signal processing fields such as communication was a reason for this. Hence, they concluded that developing sleep pattern-related features were necessary to enhance the performance of this process.

3.2.6 Linear Discriminant Analysis (LDA)

[Abo16] proposed a machine learning based method where three features obtained from the Cross-Frequency-Coupling (CFC), average power method and preferential frequency band method, using a single-channel EEG were fed as inputs to a LDA classifier for sleep stage classification. The proposed method correctly classified up to an average of 75% of the stages using a combination of both average power and CFC features on the PhysioNet Sleep European Data Format (EDF) database. The greatest misclassification problem of this scheme was related to epochs in REM which were classified as N1.

3.2.7 Hidden Markov Model (HMM)

Doroshenkov in his work on ”Usage of Hidden Markov Models for automatic sleep stages classification” [Dor07] used signals from two EEG channels with 30 second interval segments. To extract and evaluate delta, theta, alpha and beta rhythms, each segment was windowed by a Hamming window and then filtered in correspondent frequency range by a FFT-based band pass filter. Rhythm index was set using thresholding method. If the amplitude of some signal portion exceeded the threshold, then that part was considered as containing the rhythm and its duration was added to index calculation. EEG features were used for initialization and training of HMM.

Baum-Welch algorithm was employed for training. Following cluster analysis which revealed the groups of similar elements, the number of such groups was set equal to number of stages to be identified. Cluster centers were stored in the model and used to label observations.

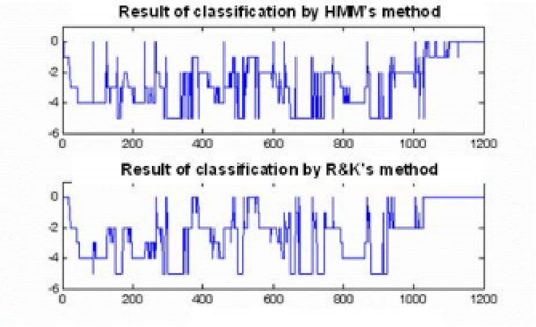


Figure 3.2: Hypnogram plot for HMM approach [Dor07]

Classification was performed by utilizing Viterbi algorithm which inputs a set of signal characteristics for all EEG segments. The resulting sequence of state changes was considered as a sequence of sleep stages. Finally, the sequence of the obtained stages was filtered by a median filter to eliminate short-term transients. Such transients were related to non stationary EEG portions which had features from two different sleep stages. Value averaging for 2-3 minutes helped clean up the graphics and revealed better sleep structure. Based on this post-filtration, the hypnogram was finally plotted. Their experimental results of hypnogram plots were as per Figure 3.2.

3.2.8 Summary of Machine Learning based approaches

Approximately 31% of all the machine Learning methods for sleep stage classification use classification schemes that are based on SVM classifiers, 22% based on Nearest Neighbor (NN) classifiers, 11% based on LDA, 10% based on KNN, 5% based on DT, 5% based on Naïve Bayes (NB) classifiers and less than 5% based on other types, such as HMM, Adaboost, Bagging, quadratic, RF and K-means. Approximately 14% are also based on other types, such as fuzzy classification and combined classification [Abo16]. Among all the techniques random forest has performed the best with 97% accuracy for one-versus-all classification. But this is not a good method of evaluation since it means we are training 5 different classifiers with 5 different optimal set of features in each case for classifying 5 classes individually, which is highly resource intensive and time consuming.

Major limitations of using hand-crafted features are:

1. They are less convenient since finding what the most optimal set of features (spectral, temporal, entropy, statistical, linear and non-linear) are leaves a large room for discussion.

2. This makes the entire process very time-consuming.

3. Most of the times, an individual classifier does not provide good results in terms of classifying all sleep stages simultaneously. E.g. sleep stage N1 might be best identified by Random forest classifier while N2 might be best represented by SVM classifier.

3.3 Deep Learning based approaches

These approaches differ from the hand-crafted feature based approaches in a way that the features are automatically chosen by the neural network. The neural network learns by itself with weight adjustment (whose concept will be described later), regarding what features should be learnt to classify to the correct category. ”Which features are chosen” is a black box for classification. However, with techniques like Deep Dreaming, the chosen features can be discovered.

Zhang applied Sparse Deep Belief Network (SDBN) [Zha16] to extract features from the EEG, EOG and EMG signals. These signals were down-sampled at 64Hz and band pass filtered: EEG(0.5 to 32Hz), EOG(0.5 to 32Hz) and EMG(5 to 32Hz). SDBN was used for automatic feature extraction. Next, a combination of multiple classifiers namely SVM, KNN and HMM was used to classify sleep stages for every 30 second epoch of the signals . A unique aspect of their work was that they proposed a new voting principle based on classification entropy to enhance the classification performance further by harnessing the complementary information provided by the individual classifier. The results demonstrated that the SDBN was an efficient unsupervised feature extraction method for sleep data, and the combination of multiple classifiers based on classification entropy performed well on sleep stages. Although this approach gave fairly good results, it was still too complex because a combination of many classifiers was used which was both time and resource consuming.

Mousavi proposed an automatic sleep stage classification algorithm called SleepEEGNet [Mou19]. The algorithm utilized a single EEG channel with a 30s epoch interval. The dataset utilized here was Physionet Sleep-EDF dataset. It was trained using 61 polysomnograms (PSGs) in 2013, and evaluated using 197 PSG in 2018. Like our approach, they also used a network architecture consisting of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). CNN was used for time-invariant feature extraction and RNN was used to encode temporal information. Although the results were promising, the amount of data used was not enough for generalization. Also, the major time gap (between 2013 and 2018) that occurred between the experiments was not desirable.

Yildirim in an international journal of environmental research and public health [Yil19] published his study on automated sleep stage classification using EEG and EOG signals. Public datasets: sleep-edf and sleep-edfx were used for this study. In the sleepedf database, there are 15188 samples of six sleep stages. The second database sleep-edfx contains 127512 samples. 30 second segments of PSG signals were fed to a 19-layer 1D CNN model.

Chapter 4

Methodology

68 patients' data obtained from Uniklinikum Erlangen has been utilized. We have broadly implemented two neural networks in our work. The first is a network trained from scratch and the second one is using a pre-trained network (Vgg16). Apart from these two, several other networks are also tested for their behaviour on our data. The workflow pipeline for both the networks is shown in Figure 4.1.

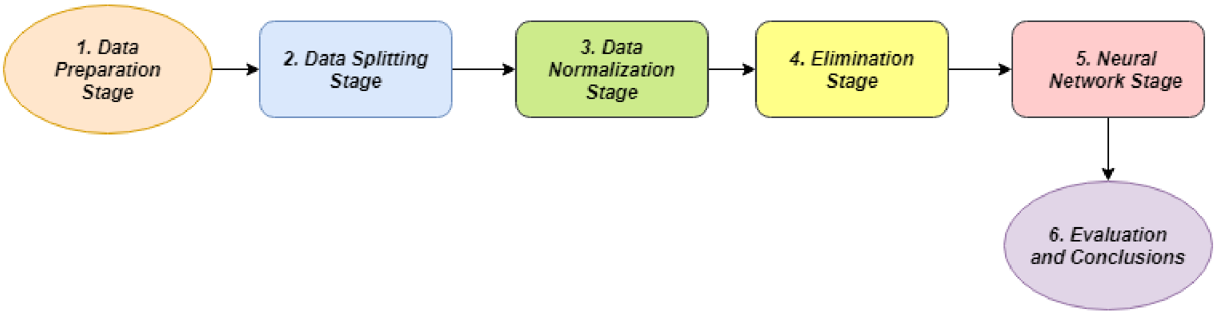

1. Data Preparation Stage
2. Data Splitting Stage
3. Data Normalization Stage
4. Elimination Stage
5. Neural Network Stage
6. Evaluation and Conclusions

Figure 4.1: Workflow pipeline

4.1 CNN-LSTM network trained from scratch

4.1.1 Data pre-processing and CNN-LSTM network

Before we delve into the network configurations (5. Neural Network Stage), let us understand the first four stages in our Workflow Pipeline in Figure 4.1.

Figure 4.2 shows a plot of the real data used during this thesis. The given data for each patient consists of three .txt files for the corresponding three channels, each with channel information and one .txt file with the corresponding label information. An array of 256 values in the channel information has EEG reading of 1s. Each label in the labels.txt file has the manually scored label for a period of 30s. Hence, every 7680 (30*256) values from the first three .txt files (comprising of channel1.txt, channel2.txt and channel3.txt) corresponds to one label value in the labels.txt file. Figure 4.2 shows channel information in green and the corresponding label information in red for first three labels of Patient 1.

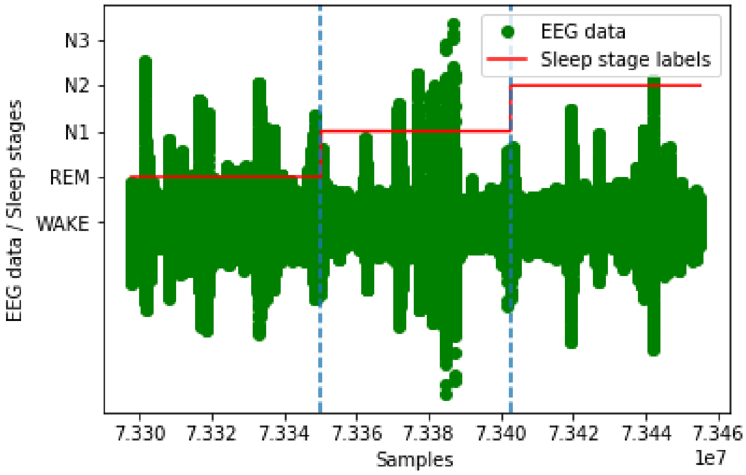


Figure 4.2: Real data depiction

Now that we have an intuition of how the given data looks like, let us understand the different pre-processing steps applied on this data before it is finally fed to the neural network for training and testing purposes.

As already discussed before, Figure 4.3 illustrates that for each of the 68 patients, channel data and label information is available. An important thing to note here is that we are working on one channel data at a time. Hence, Figure 4.3 shows 1 channel EEG data.

Figure 4.4 is for the easy visualization of the data. It states that for each patient, there is a set of data values and corresponding label values.

During the data splitting stage (Figure 4.5), the first 54 patients' data goes to training set and the remaining 14 patients' data goes to test set. This split has been performed in such a way that 80% of the overall patients are a part of training dataset and the rest are a part of test dataset.

Training and test datasets are concatenated in the form of long numpy arrays as shown in Figure 4.6.

In the next stage, the training and test data are normalized. Z- scoring has been used for normalization. Z-scores are linearly transformed data values having a mean of zero and a standard

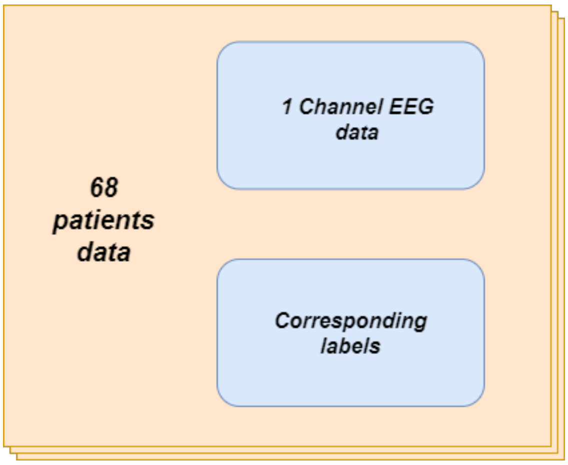


Figure 4.3: Data Preparation stage

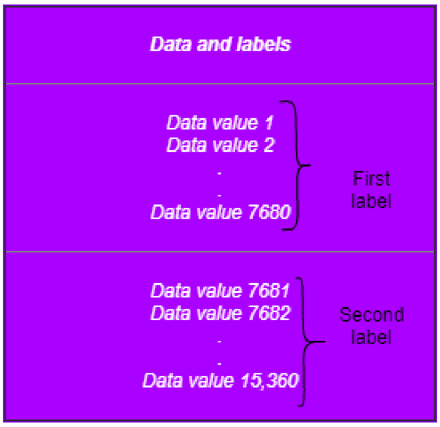


Figure 4.4: Data visualization for patient 1

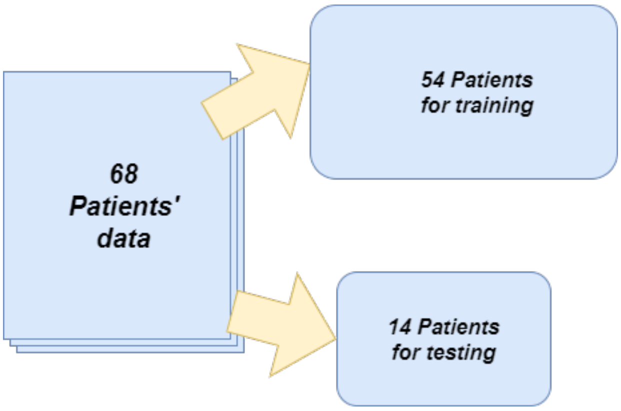


Figure 4.5: Data splitting stage

deviation of 1. This has been performed to bring data to a common standard.

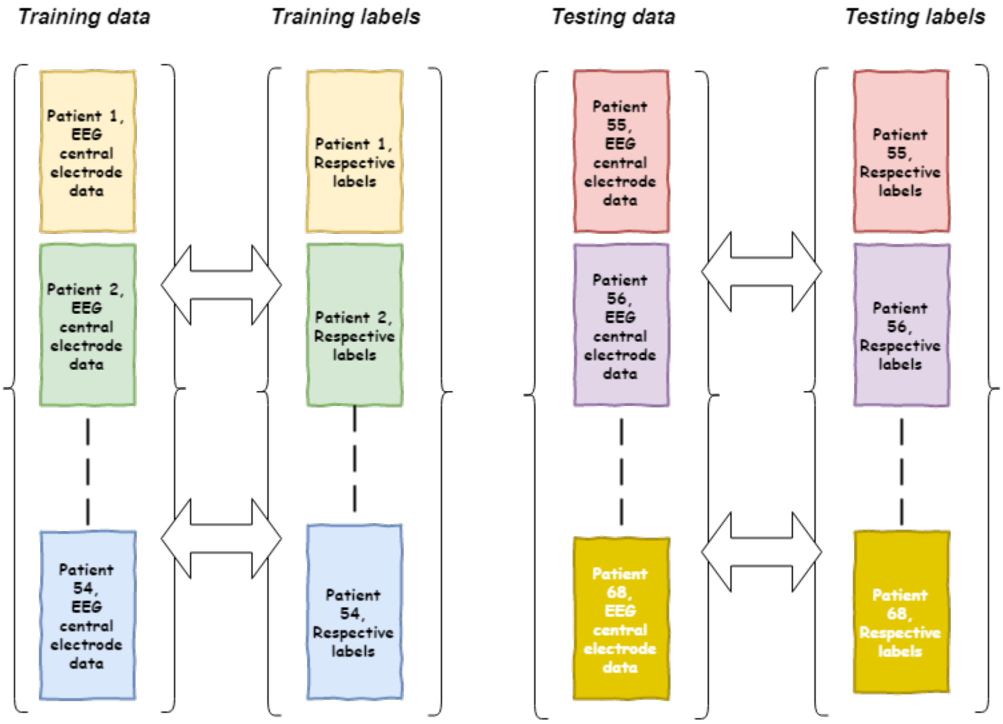

Training data
Training labels
Testing data
Testing labels
Patient 1, EEG central electrode data
Patient 2, EEG central electrode data
Patient 54, EEG central electrode data
Patient 1, Respective labels
Patient 2, Respective labels
Patient 54, Respective labels
Patient 55, EEG central electrode data
Patient 56, EEG central electrode data
Patient 68, EEG central electrode data
Patient 55, Respective labels
Patient 56, Respective labels
Patient 68, Respective labels

Figure 4.6: Final data representation

The elimination stage (Figure 4.7) follows this. The label values in the above data are -1, 0, 1, 2, 3, 4 (artefact, wake, REM, N1, N2, N3). In this stage, we eliminate all the channel data values that correspond to label -1 and also the label -1 itself. Please note that the label -1 corresponds to the sleep stage ”artefact” which we are not using during our experiments.

Now the above pre-processed data is finally fed to the neural network which contains a cascade of CNN and LSTM blocks. Figure 4.8 illustrates why each of these blocks are required.

The network in Figure 4.9 is fed with the pre-processed data obtained from the elimination step. The first layer uses 1D convolution (since the data is 1D), 32 filters, kernel size of 4, rectified linear unit for activation and a stride leap of 1. Layers 2 to 10 are also 1D convolution layers with 64 filters, kernel size of 4, stride leap of 1 and pooling by the maximum value in the kernel. The 11th layer is a 1D convolution layer comprising of 128 filters, kernel size 5 and stride leap of 5. The next two layers are bidirectional stateful LSTM layers with 32 neuronal units each. Finally we add a fully connected/ dense layer with 5 neuronal units and a softmax activation function. The Softmax function returns the prediction probability values for each of the five sleep stage labels (Wake, REM, N1, N2, N3). The label corresponding to the maximum probability value is the final predicted label for that particular case. This neural network did not utilize any hyper-parameter

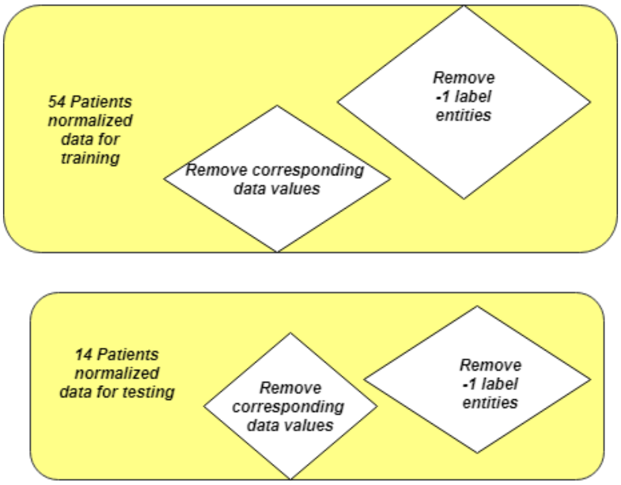

54 Patients normalized data for training
Remove corresponding data values
Remove -1 label entities
14 Patients normalized data for testing
Remove corresponding data values
Remove -1 label entities

Figure 4.7: Data elimination stage

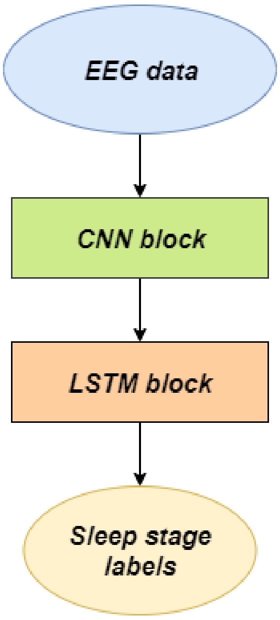


Figure 4.8: Why CNN-LSTM network ?

tuning.

Another network exactly the same as previous one except for the additional 0.2 dropout and batch normalization (as underlined in Figure 4.10) has also been designed for testing purposes. The results of both the networks will be discussed in detail in the Results and Discussion section.

All the above steps have been carried out for the first channel EEG data. The same data preprocessing steps and neural network building have also been carried out for the remaining

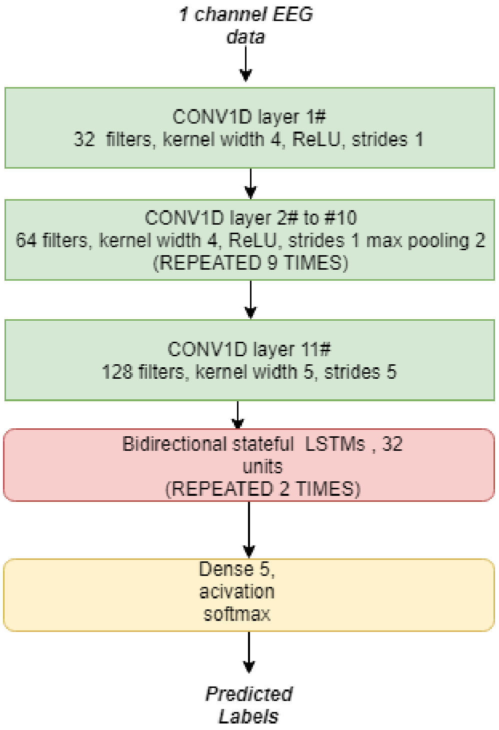

1 channel EEG data
CONV1D layer 1#
32 filters, kernel width 4, ReLU, strides 1
CONV1D layer 2# to #10
64 filters, kernel width 4, ReLU, strides 1 max pooling 2
(REPEATED 9 TIMES)
CONV1D layer 11#
128 filters, kernel width 5, strides 5
Bidirectional stateful LSTMs , 32 units
(REPEATED 2 TIMES)
Dense 5, acivation softmax
Predicted Labels

Figure 4.9: Neural network stage without dropout

two channel data. Finally, the models for all the three channels have been saved.

4.1.2 Hypnodensity Plots

In the next step, we plot the hypnodensity plots for each of the three channels separately. As discussed before, these plots visualize the probabilities for all sleep stages at each instant of time. The output of the softmax layer (probabilities) is plotted against time in seconds. Each sleep stage label's probability has been stacked one over the other with different color code. Hypnodensity is plotted for all test dataset patients together for every channel.

4.1.3 Increased temporal resolution

Hypnodensity plots are again plotted for broadly two cases : first considering samples without overlap and second with an overlap of 25 second samples.

Figure 4.11 depicts the style of concatenation of the numpy arrays of EEG data of all the test patients (patient 55-patient 68). Please note that here we have separate numpy arrays saved for each patient from 55 to 68. We already have the final models (.h5 files) saved for all the three channels individually. These three models will now be utilized to make predictions on every 30s

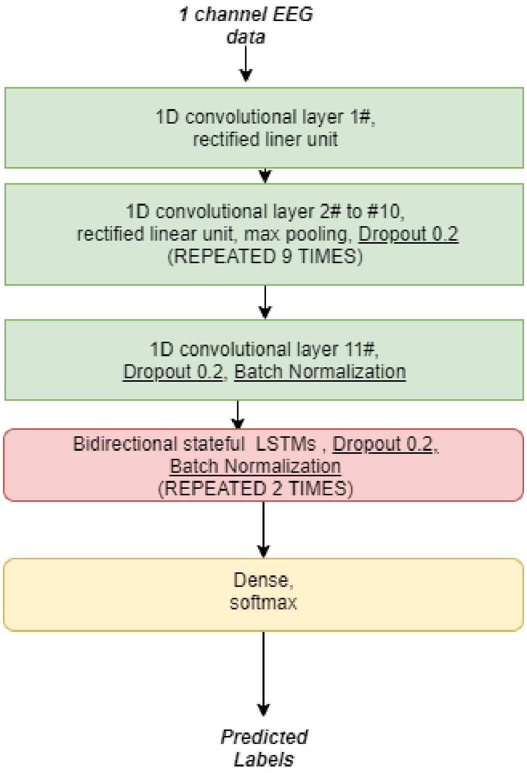

1 channel EEG data
1D convolutional layer 1#, rectified liner unit
1D convolutional layer 2# to #10, rectified linear unit, max pooling, Dropout 0.2 (REPEATED 9 TIMES)
1D convolutional layer 11#, Dropout 0.2, Batch Normalization
Bidirectional stateful LSTMs , Dropout 0.2, Batch Normalization (REPEATED 2 TIMES)
Dense, softmax
Predicted Labels

Figure 4.10: Neural network stage with dropout

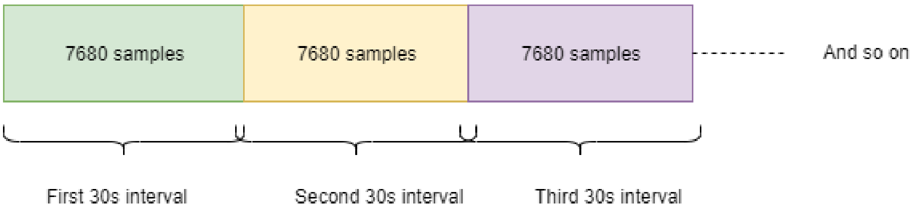


Figure 4.11: Numpy array concatenation without overlap for Patient 1

samples from Figure 4.11. The predicted probabilities for each class (obtained from softmax activation layer), for each of the 7680 samples per patient are saved in numpy arrays. Table 4.1 depicts how the numpy arrays are stored.

Now that since we have all the required class probabilities for all classes (sleep stages) and all channels (1,2,3), we are fully equipped with all information to plot the hypnodensities. The x-axis for these plots is time in seconds. Since every label is valid for a period of 30 seconds, we have plotted the class probabilities such that their color remains spread over that interval of 30 seconds.

Figure 4.12 shows the numpy array concatenation pattern for the first patient. As indicated, there is an overlap of 6400 samples everywhere. It is clear that every 30 second interval here has

Channel 1 (C4)	**Channel 2 (F4)**	**Channel 3 (O2)**
class probabilities of class Wake	class probabilities of class Wake	class probabilities of class Wake
class probabilities of class REM	class probabilities of class REM	class probabilities of class REM
class probabilities of class N1	class probabilities of class N1	class probabilities of class N1
class probabilities of class N2	class probabilities of class N2	class probabilities of class N2
class probabilities of class N3	class probabilities of class N3	class probabilities of class N3

Table 4.1: Numpy arrays of class probabilities for each patient

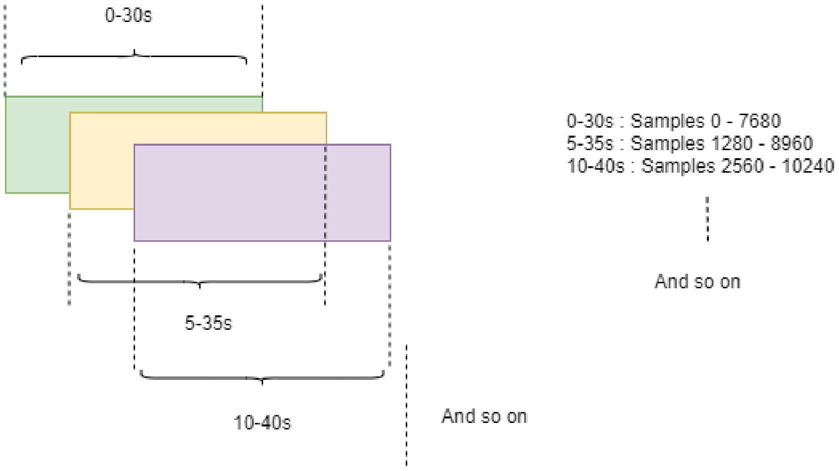


Figure 4.12: Numpy array concatenation with overlap for Patient 1

6 times the number of class probability values of the previous case without overlap.

Hypnodensity plots are again drawn for every test class patient and every channel.

4.1.4 Pearson correlation

The five probabilities for each sleep stage at a given time, can be interpreted as a 5-dimensional vector. The standard two-point Pearson correlation has been generalized to allow for correlating vectors. Thus, two 5-dimensional vectors are correlated with each other. Pearson correlation has been calculated between each pair of channels (as named below), resulting in cross-correlations. In addition, each channel has been correlated with itself, resulting in the auto-correlation:

channel 1 (C4) and channel 2 (F4)

channel 2 (F4) and channel 3 (O2)

channel 3 (O2) and channel 1 (C4)

The following formula is used for calculation:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Note that x and y are 5 dimensional vectors of class probabilities belonging to any two channels. $\bar{x}$ and $\bar{y}$ are the mean of these vectors. This is used to access the relationship between any two time series data (class probabilities of any two channels in our case).

In the above case, there was no time lag introduced between the class probabilities obtained from the three channels. It means the class probabilities coming from all the three channels starting at a time t=0 was utilized to calculate the correlations.

Next we introduce a time lag between any two channels. E.g. if suppose we consider the class probabilities from channel 1 at time t=0, we find its correlation with the channel 2 class probabilities obtained at time T= t+τ where τ varies from -150s to +150s.

$$r_{xy}(\tau) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_{i+\tau} - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_{i+\tau} - \bar{y})^2}}$$

The above formula has been used to find correlation between time lagged class probability time series from the two channels x and y. The channels used for each patient are mentioned in Table 4.2.

channel 1 and channel 1	channel 1 and channel 2	channel 1 and channel 3
channel 2 and channel 1	channel 2 and channel 2	channel 2 and channel 3
channel 3 and channel 1	channel 3 and channel 2	channel 3 and channel 3

Table 4.2: Channels for Pearson correlation with lag time

The results of both cases: with time lag and without time lag will be discussed in the Results and Discussion section.

Multidimensional Scaling (MDS)

As discussed before, MDS plots are of great help to figure out if the trained model is actually able to separate the 5 sleep stages on a neat, easy to visualize two dimensional plane. Previously saved models for Channel 1, Channel 2 and Channel 3 are used to get outputs from several layers of the network trained from scratch. We have plotted MDS for the following layers in the above network : 1,4,7,10,13,16,19,22,25,28,31,34,37,38. Pseudo code is provided in Appendix A section to show which layers activations have been utilized to plot the MDS. MDS has been plotted for all patients in both training and test dataset.

Generalized Discrimination Value (GDV)

GDV values have been averaged over a channel for all patients in both the training and test datasets. Thus we have one GDV plot per channel. The GDV calculation requires the input parameters: data points and their respective labels. For each layer, similar to MDS, GDV calculated for each patient is averaged over the same layer for all the patient.

Algorithm for Average GDV:

- Average GDV for layer 1, Channel 1= Average(GDV for layer 1, Patient 1, Channel 1), (GDV for layer 1, Patient 2, Channel 1),...., (GDV for layer 1, Patient 68, Channel 1)
- Average GDV for layer 4, Channel 1= Average(GDV for layer 4, Patient 1, Channel 1) , (GDV for layer 4, Patient 2, Channel 1),....,(GDV for layer 4, Patient 68, Channel 1)

  .

  .

  .
- Average GDV for layer 38, Channel 1= Average(GDV for layer 38, Patient 1, Channel 1) , (GDV for layer 38, Patient 2, Channel 1),...., (GDV for layer 38, Patient 68, Channel 1)

  Similarly for Channel 2:
- Average GDV for layer 1, Channel 2= Average(GDV for layer 1, Patient 1, Channel 2) , (GDV for layer 1, Patient 2, Channel 2),....,(GDV for layer 1, Patient 68, Channel 2)
- Average GDV for layer 4, Channel 2= Average(GDV for layer 4, Patient 1, Channel 2) , (GDV for layer 4, Patient 2, Channel 2),...., (GDV for layer 4, Patient 68, Channel 2)

  .

  .

  .
- Average GDV for layer 38, Channel 2= Average(GDV for layer 38, Patient 1, Channel 2) , (GDV for layer 38, Patient 2, Channel 2),...., (GDV for layer 38, Patient 68, Channel 2)

  Finally for Channel 3 also:

- Average GDV for layer 1, Channel 3= Average(GDV for layer 1, Patient 1, Channel 3) , (GDV for layer 1, Patient 2, Channel 3),....,(GDV for layer 1, Patient 68, Channel 3)
- Average GDV for layer 4, Channel 3= Average(GDV for layer 4, Patient 1, Channel 3) , (GDV for layer 4, Patient 2, Channel 3),...., (GDV for layer 4, Patient 68, Channel 3)

.

.

.

- Average GDV for layer 38, Channel 3= Average(GDV for layer 38, Patient 1, Channel 3) , (GDV for layer 38, Patient 2, Channel 3),...., (GDV for layer 38, Patient 68, Channel 3).

Ideally, the average GDV value plot should fall to as low as -1 in the final layer for each channel. But practically, the GDV just has be minimum at the last layer, as compared to other layers.

4.2 VGG16

4.2.1 Data-preprocessing and VGG16 network design

In contrary to the network trained from scratch, Vgg16 is a transfer learning framework. Some pre-trained network weights are downloaded from the imagenet website and our model has been trained on this. Vgg16 has been more commonly used with images until now but in this thesis we have made an attempt to use this pre-trained network to classify non-image data.

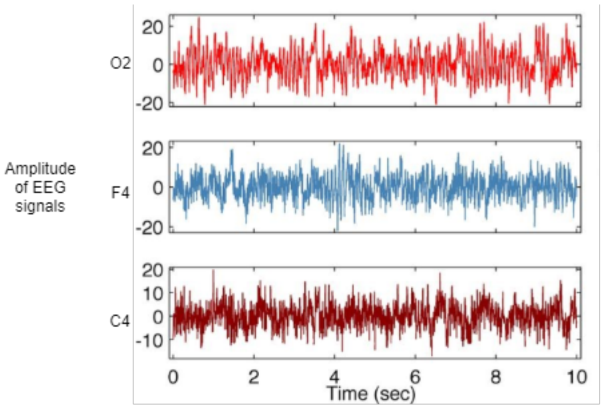


Figure 4.13: Three channel EEG data illusion

Figure 4.13 is an illustration of how our three channel data plotted against time looks like. Vgg16 requires all the three channel data to be fed simultaneously along with the respective label assigned every 30s channels data.

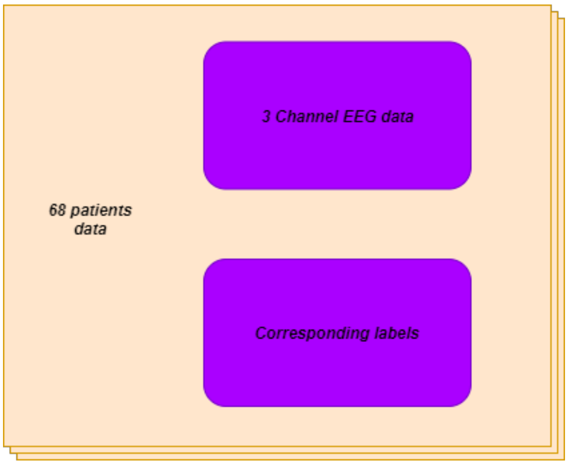


Figure 4.14: Data preparation for VGG16 network

Figure 4.14 indicates data preparation with all three channel data simultaneously.

In the next step, the 68 patients' data has been split into training and test dataset such that patients 1-54 go to the training dataset and patients 55-68 go to the test dataset. Table 4.3 shows how the numpy arrays are created for each channel. Again, we follow the elimination step as before to remove all the -1 labels and corresponding data values from all the three channels.

Numpy array data (A)	**Numpy array data (B)**	**Numpy array data (C)**	**Label**
Data value 1 to 7680, Channel 1	Data value 1 to 7680, Channel 2	Data value 1 to 7680, Channel 3	Label 1
Data value 7681 to 15360, Channel 1	Data value 7681 to 15360, Channel 2	Data value 7681 to 15360, Channel 3	Label 2
Data value 15361 to 23040, Channel 1	Data value 15361 to 23040, Channel 2	Data value 15361 to 23040, Channel 3	Label 3
And so on....	And so on....	And so on....	And so on....

Table 4.3: Numpy array arrangement for VGG16 network

Next, we train and evaluate the network (Figure 4.15) on the training and test dataset respectively. Results are recorded and discussed in the Results and Discussion section.

Please note that this network has no LSTM layers. It means only the features are learnt, not the time series sequence dependency.

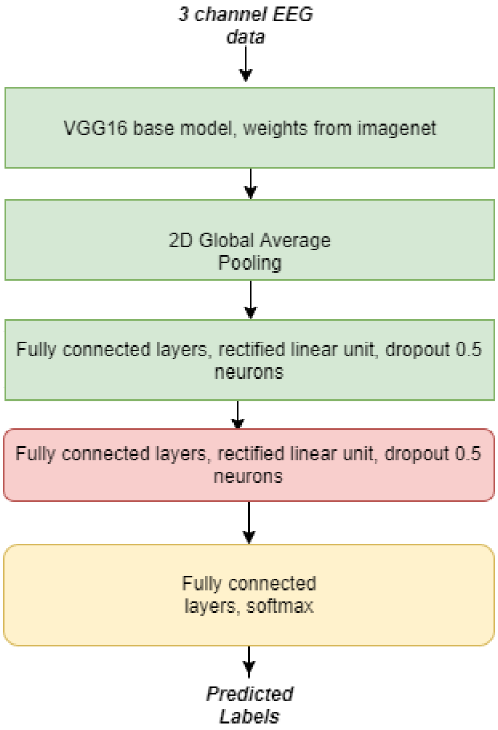

3 channel EEG data
VGG16 base model, weights from imagenet
2D Global Average Pooling
Fully connected layers, rectified linear unit, dropout 0.5 neurons
Fully connected layers, rectified linear unit, dropout 0.5 neurons
Fully connected layers, softmax
Predicted Labels

Figure 4.15: VGG16 network

4.2.2 Hypnodensity plots

Same as before, one hypnodensity plot is plotted for all the test data patients put together. This time it is just one plot because we have combined the information from all the three channels and trained on a single network.

4.2.3 Increased Temporal Resolution

Overlap and non-overlap case hypnodensity plots are again plotted.This is again done for all the patients in both training and test dataset. This time we have one hypnodensity plot per patient.

Numpy arrays used in this case are :

class probabilities of class WAKE

class probabilities of class REM

class probabilities of class N1

class probabilities of class N2

class probabilities of class N3

4.2.4 Pearson correlation

Pearson correlation cannot be calculated in this case because we are dealing with all the three channel data simultaneously.

4.2.5 Multidimensional Scaling (MDS)

MDS plots are again plotted for the same layers as before : 1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 38. This time we have only 14 MDS plots per patient.

4.2.6 Generalized Discrimination Value (GDV)

One average GDV plot along with error bars is plotted considering all layers of all patients.

Algorithm for average GDV:

- Average GDV for layer 1 = Average(GDV for layer 1, Patient 1), (GDV for layer 1, Patient 2),...., (GDV for layer 1, Patient 68)
- Average GDV for layer 4 = Average(GDV for layer 4, Patient 1), (GDV for layer 4, Patient 2),...., (GDV for layer 4, Patient 68)

  .

  .

  .
- Average GDV for layer 38 = Average(GDV for layer 38, Patient 1), (GDV for layer 38, Patient 2),...., (GDV for layer 38, Patient 68)

4.3 Other networks

Apart from the above two networks, few other networks have also been trained :

- Densenet121
- Inception V3
- InceptionResnet V2
- Resnet 50

- VGG19
- Xception

The results of these networks will be discussed in the Results and Discussion section.

Chapter 5

Results and Discussion

5.1 CNN-LSTM network trained from scratch without dropout

All the relevant results obtained in the case of CNN-LSTM network trained from scratch without any hyperparameter tuning for central channel (channel 1) will be discussed in this section.

5.1.1 Accuracy - Loss curve

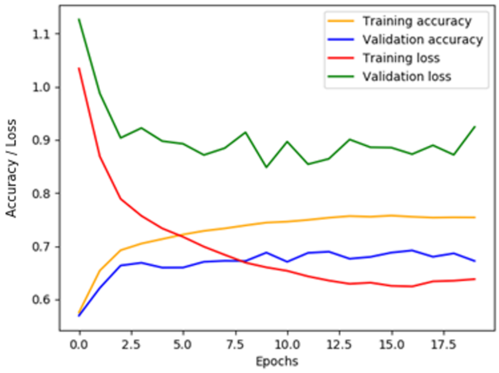
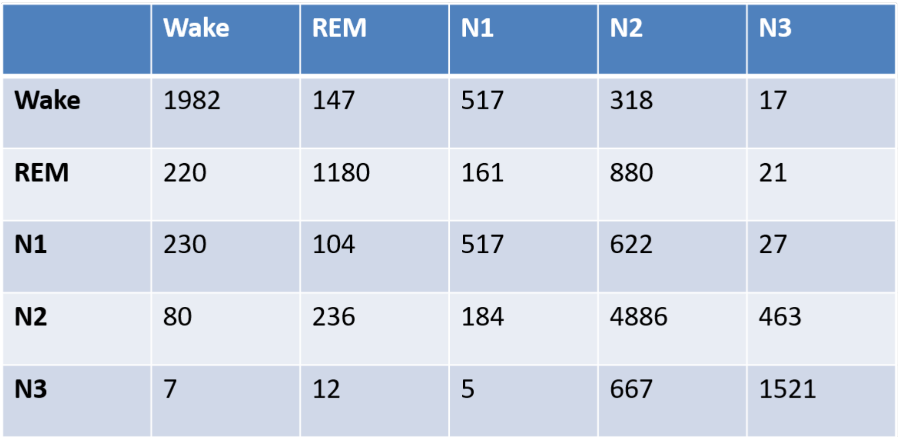


Figure 5.1: Accuracy versus epochs for CNN-LSTM network without dropout

From Figure 5.1, we can see that CNN-LSTM network without dropout gives a validation accuracy of 67%. Training accuracy reached 76%. Overfitting starts post 15 epochs when the training accuracy continues to increase but validation accuracy keeps decreasing slightly. This moment onwards, the model does not learn anymore but starts memorizing.

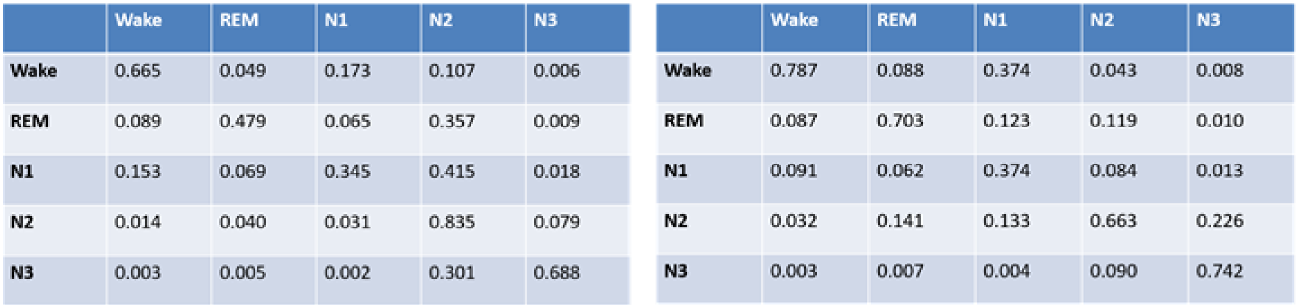
5.1.2 Confusion matrix

Figures 5.2 and 5.3 illustrate 5×5 confusion matrices. Labels in black (column wise) are true labels and the one's in white (row wise) are predicted labels. Here, the sleep stage N2 is often mistaken for N3 and vice versa. Also, the sleep stage N1 is majorly misclassified as N2. The only way for human annotators to discriminate between N1 and N2 is by identifying sleep spindles. N2 has sleep spindles, while N1 does not. Additionaly, whenever a sleep spindle occurs, the annotator has a tendency of assigning the next 30s interval entirely as N2. Therefore, this behaviour of the network is not unexpected.



	Wake	REM	N1	N2	N3
Wake	1982	147	517	318	17
REM	220	1180	161	880	21
N1	230	104	517	622	27
N2	80	236	184	4886	463
N3	7	12	5	667	1521

Figure 5.2: Raw confusion matrix without normalization



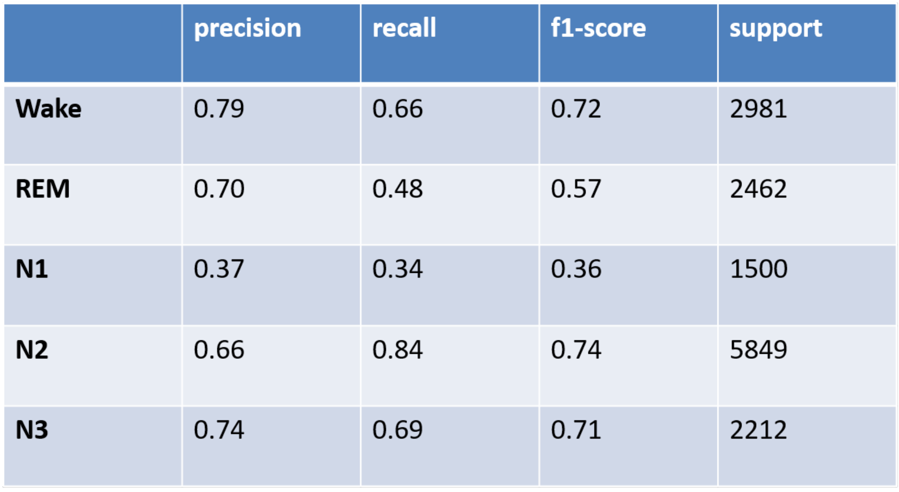
	Wake	REM	N1	N2	N3
Wake	0.665	0.049	0.173	0.107	0.006
REM	0.089	0.479	0.065	0.357	0.009
N1	0.153	0.069	0.345	0.415	0.018
N2	0.014	0.040	0.031	0.835	0.079
N3	0.003	0.005	0.002	0.301	0.688

A

	Wake	REM	N1	N2	N3
Wake	0.787	0.088	0.374	0.043	0.008
REM	0.087	0.703	0.123	0.119	0.010
N1	0.091	0.062	0.374	0.084	0.013
N2	0.032	0.141	0.133	0.663	0.226
N3	0.003	0.007	0.004	0.090	0.742

B

Figure 5.3: (A) Row-wise normalized (B) Column-wise normalized confusion matrices



	precision	recall	f1-score	support
Wake	0.79	0.66	0.72	2981
REM	0.70	0.48	0.57	2462
N1	0.37	0.34	0.36	1500
N2	0.66	0.84	0.74	5849
N3	0.74	0.69	0.71	2212

Figure 5.4: Classification report for CNN-LSTM network without dropout

5.1.3 Classification Report

The Classification report in Figure 5.4 indicates that the weighted average of precision and recall i.e. f1-score value is the best in case of the sleep stage N2. However, the best precision value is in the case of sleep stage Wake which means that the false positive alarm for it is less. Recall is the best for N2 which implies that the false negative alarm for N2 is very less. This is also verified from the fact that N2 is the most predicted sleep stage label amongst all the other stages.

5.2 CNN-LSTM network trained from scratch with dropout

Now, we discuss all the results associated with CNN-LSTM network trained from scratch with dropout layers as discussed in Figure 4.10. We will be able to discover differences from our previous network performance. (a): Channel 1 (C4) (b) Channel 2 (F4) (c): Channel 3 (O2).

5.2.1 Accuracy - Loss curve

With this network trained and tested on the C4 channel data, we obtain a validation accuracy of 70% (Figure 5.5). This validation accuracy of 70% is already very promising (considering the fact that we have used only a single channel raw EEG data) because the maximum possible inter-rater reliability of a human annotator for sleep stage classification is 80%. It means that this model has achieved a test accuracy value close to the 80% mark without even using EOG (electrooculogram) or EMG (electromyogram) data, which would significantly improve the learning.

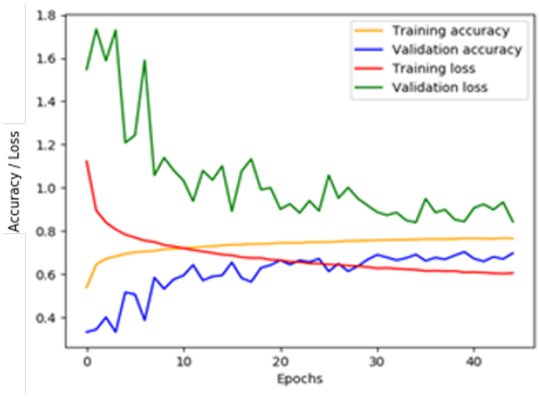


Figure 5.5: Accuracy versus epochs for CNN-LSTM network with dropout, channel 1

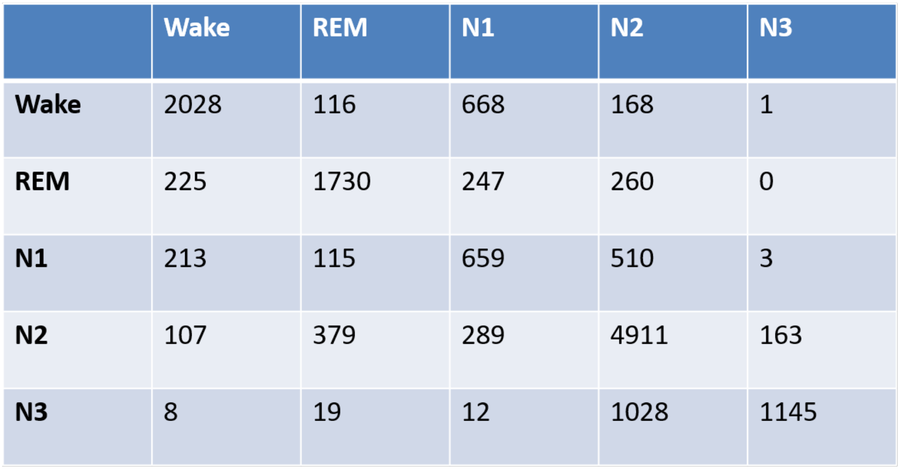
In comparison to the previous neural network without dropout and batch normalization layers, this network did not face overfitting problem in the initial layers.

5.2.2 Confusion Matrix

The Figures 5.6 and 5.7 present 5×5 confusion matrices obtained for CNN-LSTM network trained with dropout and batch normalization layers. This network clearly classifies the sleep stages Wake, REM, N1 and N2 with a higher accuracy than the previous network. For the sleep stage N3, the previous network performs better. Specially for the sleep stage REM, the event of wrongly classifying REM to N3 is reduced from 21 (in previous network) to nil. Overall the false positives for sleep stage N3 are reduced significantly but N3 is more often wrongly classified as N2. In addition to this, the misclassification of N1 to N2 drops noticeably. However, N2 sleep stage continues to be misunderstood as N1 for the reasons discussed before (section 5.1.2). To summarize, this network performs better overall for all other sleep stages except for its misclassification of N3 to N2.

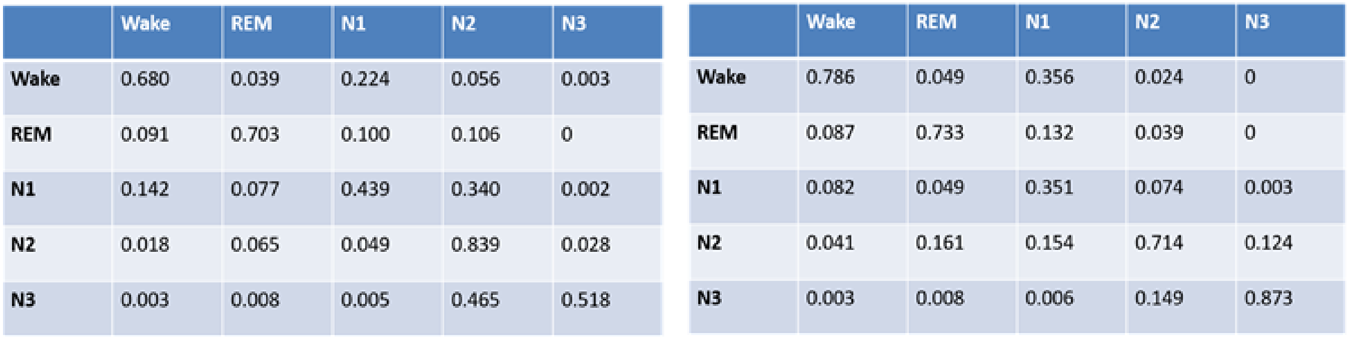
5.2.3 Classification Report

The precision values for REM, N2 and N3 as shown in Figure 5.8 improve here, the value for Wake remains the same while it drops slightly in case of N1 as compared to the previous network without dropout and batch normalization. The recall value for REM has improved significantly while it has improved slightly in the case of Wake, N1 and N2, and has dropped in case of N3. F1-scores have improved for all the sleep stages except N3.



	Wake	REM	N1	N2	N3
Wake	2028	116	668	168	1
REM	225	1730	247	260	0
N1	213	115	659	510	3
N2	107	379	289	4911	163
N3	8	19	12	1028	1145

Figure 5.6: Raw confusion matrix without normalization, channel 1



	Wake	REM	N1	N2	N3
Wake	0.680	0.039	0.224	0.056	0.003
REM	0.091	0.703	0.100	0.106	0
N1	0.142	0.077	0.439	0.340	0.002
N2	0.018	0.065	0.049	0.839	0.028
N3	0.003	0.008	0.005	0.465	0.518

A

	Wake	REM	N1	N2	N3
Wake	0.786	0.049	0.356	0.024	0
REM	0.087	0.733	0.132	0.039	0
N1	0.082	0.049	0.351	0.074	0.003
N2	0.041	0.161	0.154	0.714	0.124
N3	0.003	0.008	0.006	0.149	0.873

B

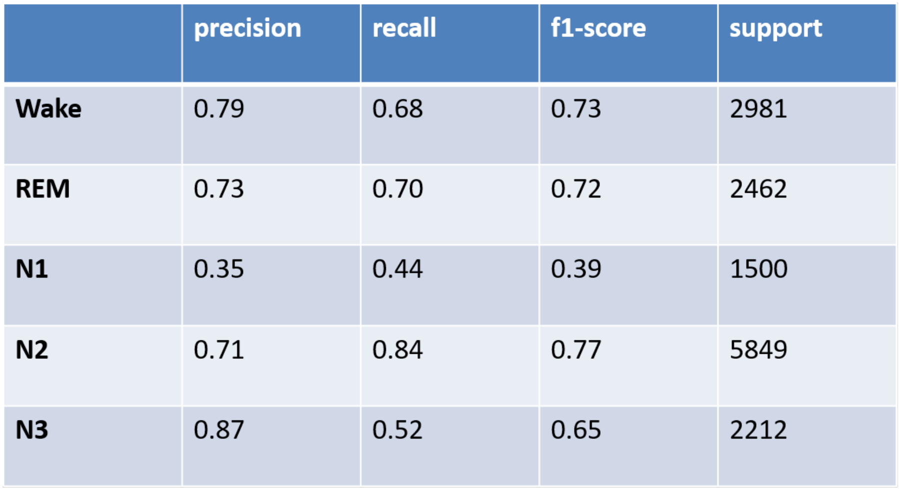
Figure 5.7: (A) Row-wise normalized (B) Column-wise normalized (channel 1) confusion matrices

5.2.4 Ground truth versus Predicted labels

We plot actual versus predicted labels (Figure 5.9) for the results of this network.

Since we now know that the network with dropout is performing better on our data, we proceed with all our further analysis with the second network (with dropout).

As a part of further analysis, we first perform hypnodensity plot for the entire test patient dataset as shown in Figure 5.9. This is a plot of probability of occurrence of each sleep stage against time in seconds. Patient 55 is plotted leftmost while patient 68 is at the rightmost in the above plot. The plot covers almost 15,000 labels each comprising of 7680 samples. N2 sleep stage is the most predicted/dominant sleep stage while Wake sleep stage is least popular among



	precision	recall	f1-score	support
Wake	0.79	0.68	0.73	2981
REM	0.73	0.70	0.72	2462
N1	0.35	0.44	0.39	1500
N2	0.71	0.84	0.77	5849
N3	0.87	0.52	0.65	2212

Figure 5.8: Classification report for CNN-LSTM network with dropout, channel 1

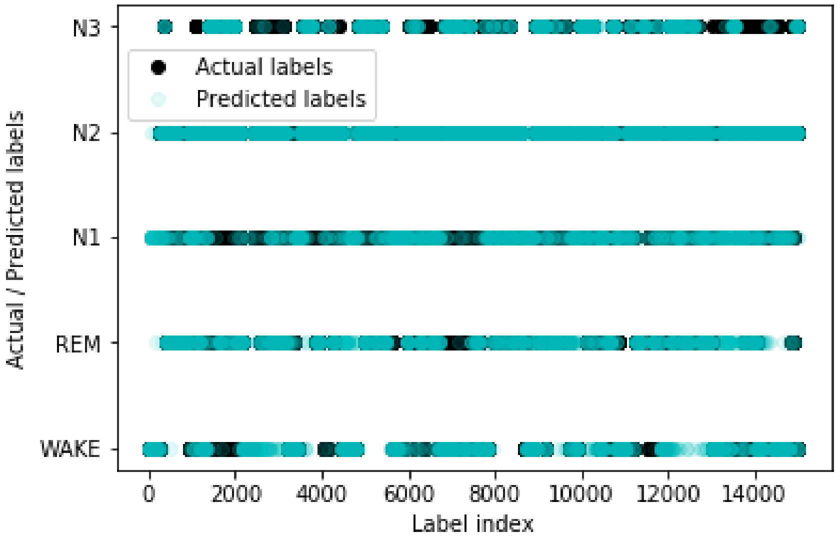


Figure 5.9: Actual and predicted labels for all test set samples, channel 1

our data. Similarly, below are the results for channel 2 and channel 3.

5.2.5 Results for channel 2

Accuracy - Loss curve

The validation accuracy obtained in case of the same network with dropout and batch normalization trained on channel 2 data is 68%, as illustrated in Figure 5.11. This is slightly lower than the first channel accuracy. The model faces overfitting problem (gap between training and validation

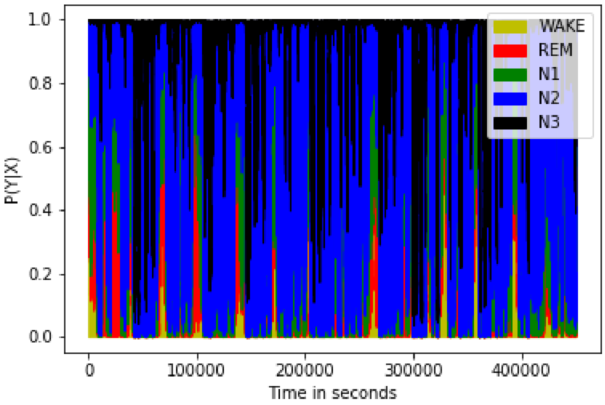


Figure 5.10: Hypnodensity plot for patient 55-68 combined, channel 1

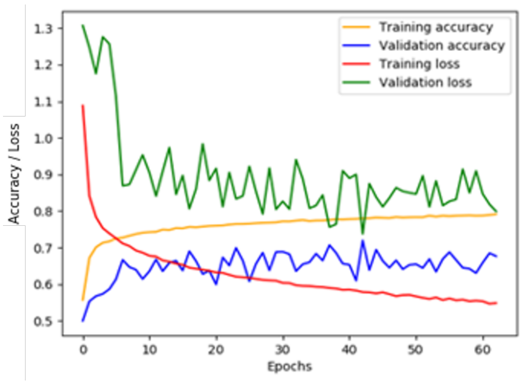


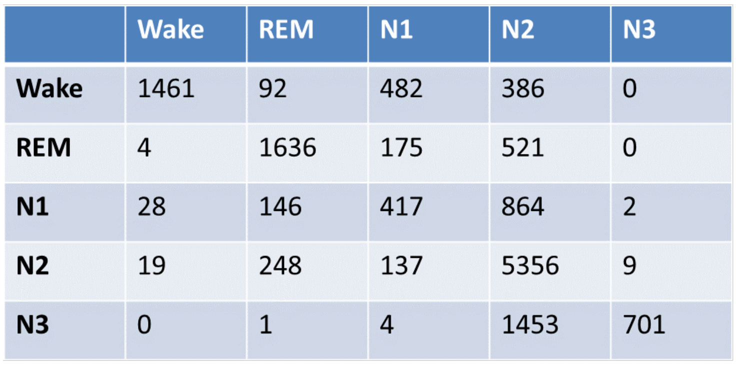
Figure 5.11: Accuracy versus epochs for CNN-LSTM network with dropout, channel 2

accuracy increases post 40 epochs).

Confusion Matrix

The confusion matrix in Figure 5.12 indicates that N3 is most often misclassified as N2 in case of channel 2 as compared to channel 1. The sleep stage N1 is also strongly mistaken for N2. But, Wake and REM are never falsely classified as N3. N2 is the best classified label here as compared to the network trained on the other two channels (C4, O2).

Similarly, we plot test dataset hypnodensity plot for channel 2 (Figure 5.13). In this plot as well, N2 dominates. REM is the least predicted sleep stage with channel 2 (F4) electrode.



	Wake	REM	N1	N2	N3
Wake	1461	92	482	386	0
REM	4	1636	175	521	0
N1	28	146	417	864	2
N2	19	248	137	5356	9
N3	0	1	4	1453	701

Figure 5.12: Raw confusion matrix without normalization, channel 2

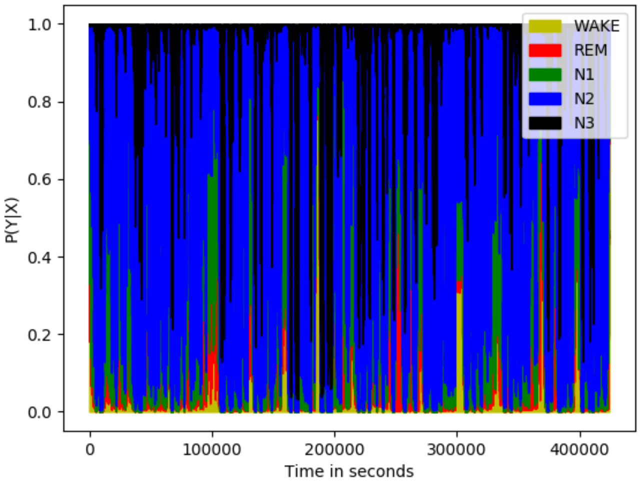


Figure 5.13: Hypnodensity plot for patient 55-68 combined, channel 2

5.2.6 Results for channel 3

Accuracy - Loss curve

The validation accuracy in case of the third channel (O2) is 67% as plotted in Figure 5.14. This is the least amongst all the three channels. Contrary to the F4 channel, here the model's training accuracy does not improve beyond 79%. Thus there is no increasing gap (overfitting) between training and validation accuracies as before.

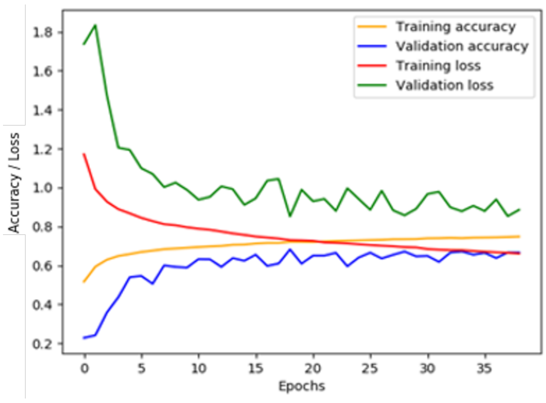
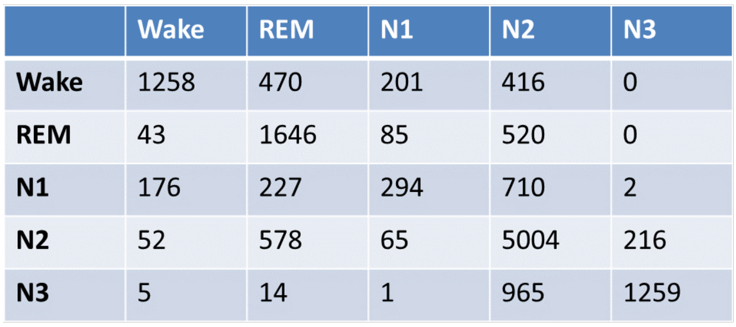


Figure 5.14: Hypnodensity plot for patient 55-68 combined, channel 3



	Wake	REM	N1	N2	N3
Wake	1258	470	201	416	0
REM	43	1646	85	520	0
N1	176	227	294	710	2
N2	52	578	65	5004	216
N3	5	14	1	965	1259

Figure 5.15: Raw confusion matrix without normalization, channel 3

Confusion Matrix

With the O2 channel, the Wake is most often misclassified as REM. N3 is best classified with O2 channel as compared to the other two channels. The instances of misclassification of N2 to N1 are the least in this channel. This is illustrated in Figure 5.15.

In the hypnodensity plot for O2 channel (Figure 5.16), REM has slightly more relevance as compared to its relevance in the other two channels. Wake stage is almost negligible.

5.2.7 Hypnodensity plots for 30s and 5s

Table 5.1 shows hypnodensity plots for Patient 58 and Patient 61 for all the three channels without overlap of time intervals. Patient 58 has a dominant REM especially in the third channel. This makes sense because both the O2 channel and rapid eye movement sleep are related to vision senses. In the other two channels, N2 is dominant as expected. In the case of Patient 61, there is an unusual behaviour observed, i.e., Wake stage is heavily dominant. It can mean that the patient

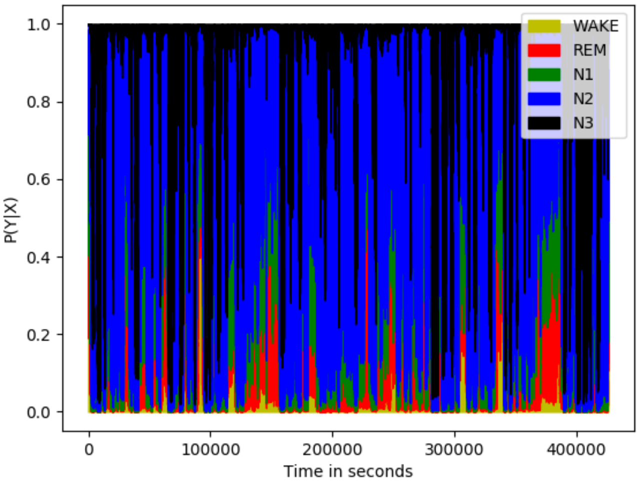


Figure 5.16: Hypnodensity plot for patient 55-68 combined, channel 3

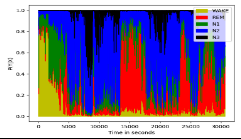
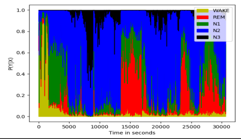
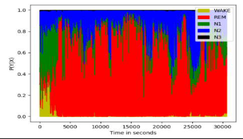
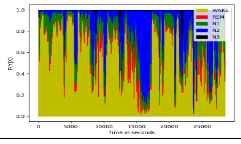
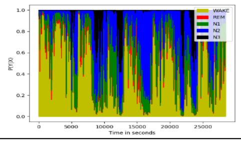
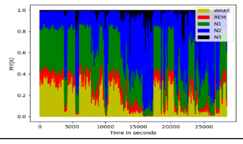
Patient	Channel 1(C4)	Channel2 (F4)	Channel3 (O2)
58			
61			

Table 5.1: Hypnodensity plots without overlap

suffers from sleep apnea. The hypnodensity plots for all the test set patients are provided in the appendix section for reference.

We now discuss hypnodensity plots for 5s temporal resolution.

Table 5.2 shows the similar hypnodensity plots for the overlap case. These plots are very much similar to their non-overlap counterparts in their general structure. They have 6 times increased temporal resolution as compared to their counterparts in Table 5.1. Here, since the resolution is high, finer details from plots can be better visualized.

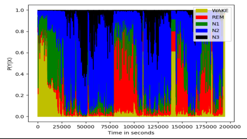
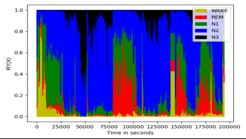
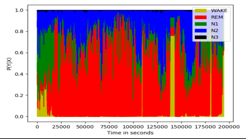
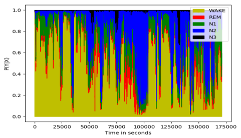
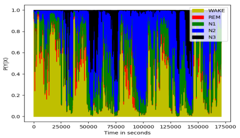
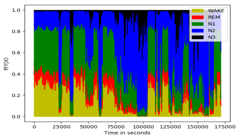
Patient	Channel 1(C4)	Channel2 (F4)	Channel3 (O2)
58			
61			

Table 5.2: Hypnodensity plots with 5 second overlap

5.2.8 Pearson correlation

As discussed before, we first provide a table of vectorial pearson correlation values between any two channels without lag time.

Patient	Channel 1-2 Correlation	Channel 2-3 Correlation	Channel 1-3 Correlation
55	0.184373933	0.218314148	0.23832961
56	0.127254266	0.17068805	0.177811556
57	0.149541949	0.18389718	0.19569853
58	0.136983925	0.142665644	0.128460181
59	0.235199701	0.21412203	0.193892689
60	0.230949007	0.214901397	0.195311334
61	0.09153098	0.168660245	0.185697112
62	0.136991305	0.155605077	0.136248182
63	0.166407985	0.24484901	0.266273656
64	0.125783789	0.148958166	0.140146258
65	0.102773602	0.157762457	0.15638723
66	0.146744348	0.168794445	0.17542335
67	0.177263544	0.179284297	0.220182963
68	0.158313424	0.165832631	0.159273146

Table 5.3: Pearson correlations without lag time

Firstly, it is very clear that there is no anti-correlation between channels (no negative values). They are always positively correlated in the range of 0-0.2 values. Also from Table 5.3, we can roughly say that the correlation values between channel 1-3 > channel 2-3 > channel 1-2 in most cases, when there is no lag time introduced.

We now discuss the results of Pearson correlations with lag time.

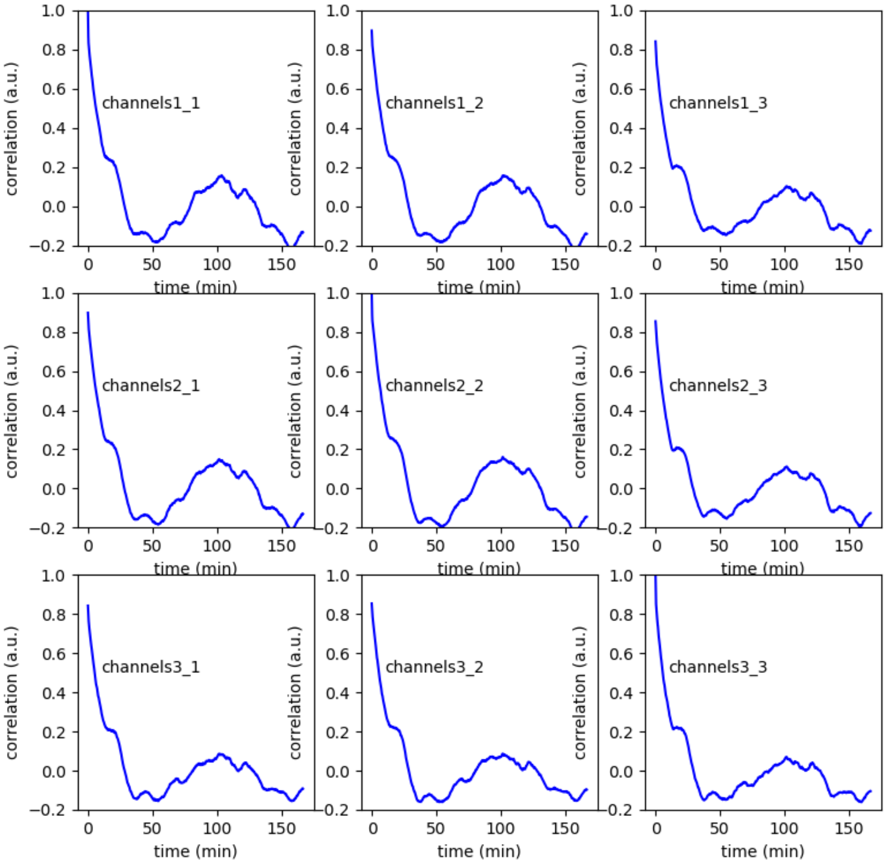


Figure 5.17: Pearson correlations with lag time for patient 55

As we can see from Figure 5.17 and Figure 5.18, there is a peak for the correlation value observed at close to 100 min. This matches the medically proven fact that every sleep stage repeats itself after round about 100 min interval.

Typically, a person would begin a sleep cycle every 90-110 minutes resulting in four to five cycles per sleep time. The order within one sleep stage cycle is: Wake-N1-N2-N3-N2-N1-REM and then a new cycle begins, starting from N1 again.

5.2.9 MDS Plots

Table 5.4 contains the results of the MDS plots for Patient 1, channel 1 for the specified layers. ’L’ refers to the layer in the neural network from which the MDS plot is taken.

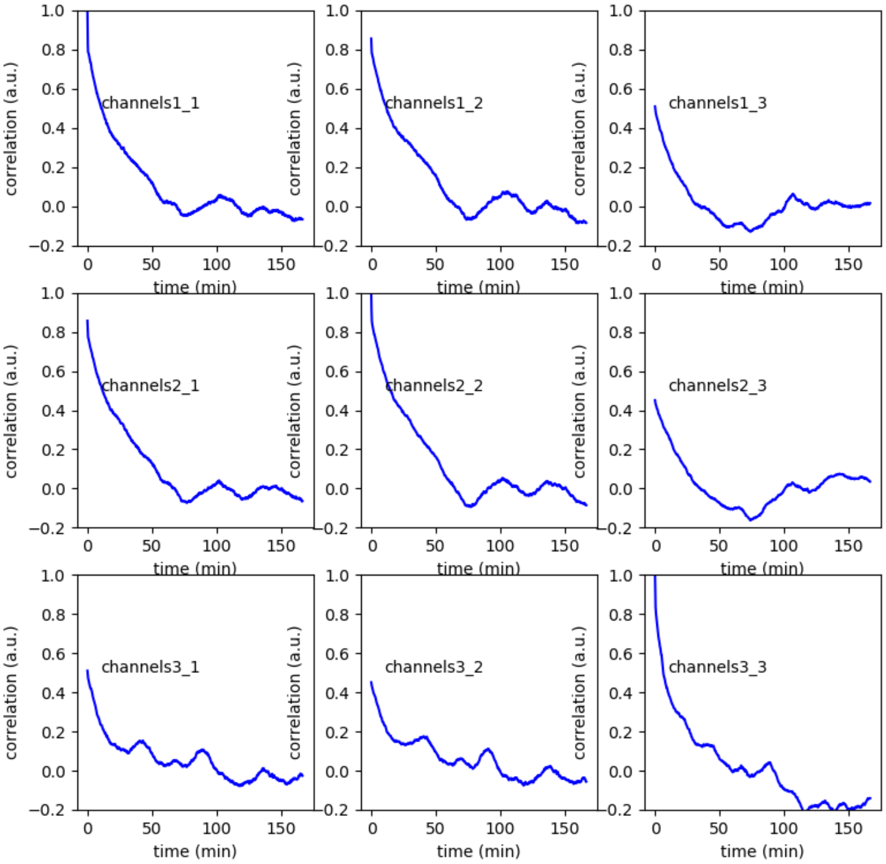


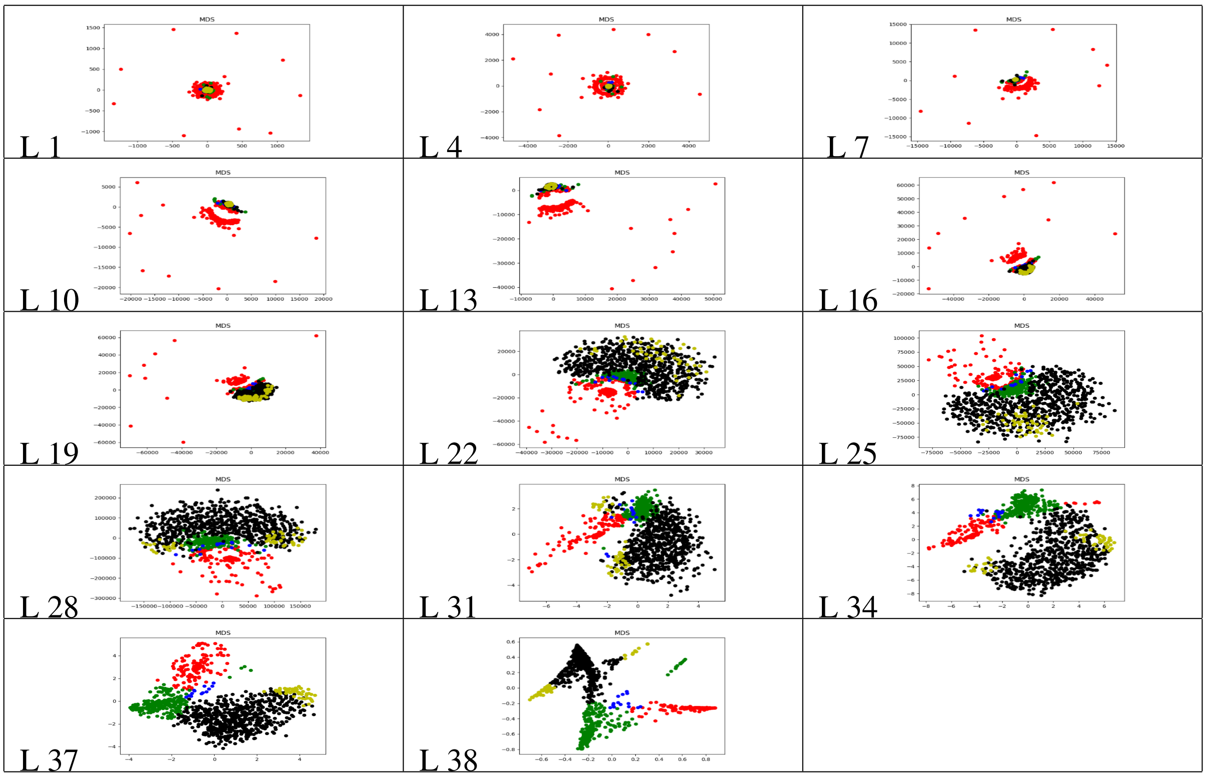
Figure 5.18: Pearson correlations with lag time for patient 63

As we can see from the series of MDS plots, there is a clear class separation observed in the final layers of our network.

Such plots have been generated for all patients, all channels and specified layers. For the sake of convenience, only one patient's first channel results have been shown above.

5.2.10 GDV plots with error bars

Figure 5.19 represents the average GDV values which have been plotted along with the error bars for each of the 14 layers of our neural network. We see a decreasing trend in the values proceeding towards the last layers.



L 1 L 4 L 7

L 10 L 13 L 16

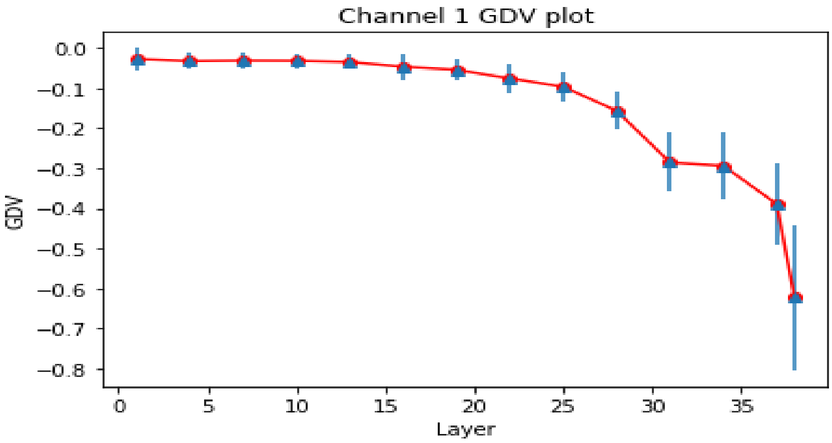
L 19 L 22 L 25

L 28 L 31 L 34

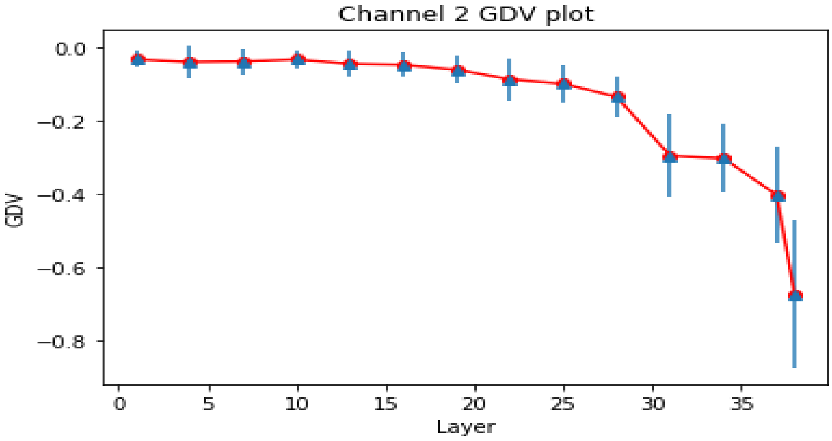
L 37 L 38

Table 5.4: MDS plots for Patient 1, Channel 1

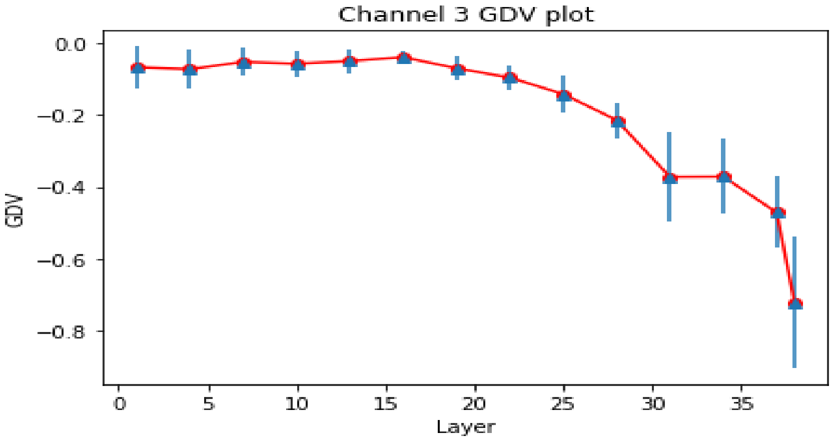
Values become more negative which indicates that intra-class distances decrease and inter-class distances increase towards the last layers, i.e., the class separation becomes better with each subsequent network layer.



(a) Averaged GDV plot channel 1



(b) Averaged GDV plot channel 2



(c) Averaged GDV plot channel 3

Figure 5.19: Averaged GDV plots per channel

Chapter 6

Conclusion

6.1 Conclusion from different networks

In this first section of conclusion, we jot down the performances of various networks we have implemented in terms of the accuracy metric in Table 6.1.

Network	**Validation Accuracy**
CNN+LSTM without dropout, Channel 1	67
CNN+LSTM with dropout, Channel 1	70
CNN+LSTM with dropout, Channel 2	68
CNN+LSTM with dropout, Channel 3	67
VGG16	66
Resnet50	29
VGG19	45
Densenet121	58
InceptionV3	27
InceptionResnetV2	56
Xception	58
Attention mechanism based network	54

Table 6.1: Performance of different neural networks

Firstly, the best performance is achieved in the case of CNN+LSTM network with dropout and batch normalization, trained on channel 1 data i.e. the central electrode EEG data. Frontal and occipital electrode performance falls next in terms of performance. We later inspect performances on pre-trained networks, trained on image dataset. VGG16 network without LSTM performs the best among other transfer learning models. Attention based network is also not able to perform very well on our data. We thus conclude that a "*not very deep CNN network (trained from scratch) trained end to end along with LSTM layers*" performs the best on our raw single channel EEG data in terms of scoring the sleep stages.

6.2 Conclusion from hypnodensity plots

As we already know, the positions of the three electrodes : C4, F4, O2 are on central, frontal and occipital portions respectively of the right hemisphere of the brain. These electrodes comprise our Channel 1, Channel 2 and Channel 3 data in the order : C4, F4, O2. In the hypnodensity plot for channel 3, there is a lot of red portion (Figures 5.15 and 5.16) i.e. REM prediction probabilities which are related to EOG artifacts.This means there is a strong influence of EOG signals in Channel 3: O2. This also makes sense biologically from the electrode placement position because the cortex occipital lobe is associated with visual processing in the brain. Henceforth, the network trained and validated on the third channel data does not perform very well in terms of classifying sleep stages. *Thus we conclude that it is not a very good idea to base our sleep research on the EEG data obtained from the electrode placed on occipital lobe of the cerebrum because of EOG interferences.*

Next let us look into the Channel 2 data obtained from the electrode placed on the frontal lobe of the right cerebral hemisphere. A study by Sayaka [Ari12] revealed that an increased cerebral blood flow in the right frontal lobe area during sleep precedes self-awakening in humans. The results obtained from Figure 5.12 and 5.13 are in agreement with this study. The yellow portions representing Wake sleep stage are dominant here. Wake is the period when the brain wave activity is at its highest and muscle tone is active. Thus there is a possibility of EMG artifacts in the measured EEG signal. *Therefore it might not be a very good idea to again base our sleep research on right frontal lobe EEG data because it might be mixed with EMG interferences.*

Also considering all the hypnodensity plots coming from all channels it is clear that the most dominant sleep stage in which a human remains in most part of his/her sleep is the N2 stage i.e. the blue portions are the most dominant in all the hypnodensity plots. This indeed complies with the established study about sleep stages from ages. *According to studies, N2 accounts for 45-55% of the sleep time, which is validated by our plots as well.*

In any normal human subject, the order within one sleep stage cycle is is: Wake-N1-N2-N3-N2-N1-REM and then a new cycle begins starting with N1 again. Also from the plots, it is clear that any patient's sleep begins with yellow (Wake) domination and later enters into deeper stages like N3. N3 sleep is a regenerative period where the body heals and repairs itself. The first episode of Stage N3 lasts from 45-90 minutes. Subsequent episodes of N3 sleep have shorter and shorter time periods as the night progresses. This information is also largely conveyed from our plots. For some patients, e.g. patient 68 and patient 61, there is a drastically decreased span of sleep stage N3. Probably the patient is old or has sleep apnea. Usually, REM sleep occurs 90 minutes after one falls asleep. The first period of REM typically lasts 10 minutes. Each of the later REM stages gets longer, and the final one may last up to an hour. From our plots as well, we know that REM has greater prediction probabilities towards the end of sleep rather than in beginning. One can have intense dreams during REM sleep, since the brain is more active during this stage. There are a lot of other variations in switching patterns of sleep stages and duration of sleep stages in different patients, but all these details give us some idea about the medical conditions of the patient.

By far, the performance of the central electrode channel has been good in terms of sleep stage scoring. Plots also indicate that the central channel incorporates a good blend of information from both frontal and occipital lobes without letting artifacts (EMG or EOG) interfere. Accuracy values comply with this argument.

6.3 Conclusions from temporal resolutions

Hypnodensity plots with 30 second temporal resolution and 5s temporal resolution for any given patient and any channel look similar. Going back to the basics, the model has been trained on 30s interval data samples and tested on the samples from 5s resolution. Considering different temporal resolutions help us check how the introduction of 5 second new samples on the right side of the window and elimination of 5 second old samples on the left side of the window affects the overall class prediction probabilities. The overlap case has exactly 6 times higher resolution than the non-overlap case, because every 30s interval will have six 5 second intervals.

6.4 Conclusions from Pearson correlations

From the Pearson correlation values table (Table 5.3), it is clear that there is no anticorrelation between channels. They are always positively co-related with correlation values ranging between 0.1 to 0.3. Going by the general trend, the correlation values between channel 1 and channel 3 are the greatest. It means that the central electrode EEG is around 20 % similar to the occipital electrode EEG in terms of its effect on sleep stages. However, none of the correlation values in the table reach 1 or even close to 1. So no conclusion can be derived regarding different parts of the brain being in exactly same sleep stages at the same time. From the graphs obtained for the Pearson correlations with time lag (Figures 5.17 and 5.18), we conclude that an average sleep cycle in a healthy subject lasts for around 100 minutes. This exactly goes hand in hand with the medically proven values.

On an average, a healthy human subject goes through 5 sleep cycles, each lasting around 90-100 min overnight (over a period of 8 hours).

6.5 Conclusions from MDS and GDV plots

As we can see from the series of MDS plots (Table 5.4), in the initial layers all the data points are merged. As the layers proceed, the class separations become more and more clear. This is also validated by the decreasing GDV values in Table 6.2.

Layer	**Channel 1 GDV**	**Channel 2 GDV**	**Channel 3 GDV**
1	-0.02710283	-0.03408605	-0.06689271
4	-0.03251901	-0.04147052	-0.07233295
7	-0.03128821	-0.03971379	-0.05209723
10	-0.03170958	-0.03447679	-0.05708989
13	-0.03487176	-0.04648827	-0.0496664
16	-0.04648112	-0.04908253	-0.0389913
19	-0.05422728	-0.06256721	-0.06965655
22	-0.0754481	-0.08850469	-0.09580956
25	-0.09651839	-0.10079171	-0.14166454
28	-0.15636904	-0.13626807	-0.21428136
31	-0.2854611	-0.29673629	-0.37266508
34	-0.29396277	-0.30402516	-0.37198284
37	-0.38930891	-0.40403928	-0.47150323
38	-0.62289644	-0.67361992	-0.72172337

Table 6.2: Average GDV values for Network with dropout

The average GDV values (averaged over all patients) decrease with increase in the layers. For channel 3, this decrease has been the best (-0.7). So, in terms of GDV, channel 3 performs the best.

Chapter 7

Future Work

7.1 Use the left cerebral hemisphere data

In our work, we have used the three channel data: C4, F4 and O2 recorded by the electrodes placed on right cerebral hemisphere. In a similar manner, the channel data coming from the left cerebral hemisphere can also be inspected to understand how the left and right cerebral hemispheres are similar or different in terms of the underlying sleep stages of the brain. In this case, the pearson correlations between the channels are symmetrically opposite to each other i.e. C3-C4, F3-F4, O1-O2 would be interesting to observe. It might indicate if different parts of the brain are in same/ different sleep stages at the same time.

7.2 Use the remaining 18 EEG placements of 10-20 system

In order to incorporate maximum information in our study, we could make use of all the 21 electrodes (3 are already used in our study) of a typical 10-20 system. Here, the EEG information from surface of ears will also be considered. So, indeed we can study what kind of electrical activity is observed in different parts of our brain while in different sleep stages. This will give us a deeper insight into sleep related normalities / abnormalities and broaden our scope of sleep research.

7.3 Data preprocessing could be explored more

In our study, we have used least amount of data preprocessing i.e. just the z-scoring. We have tried to make use of almost raw single channel EEG data. More sophisticated data preprocessing techniques including jump artifact removal and manually removing the obvious artifacts by visual inspection methods could help us get rid of outliers in our data and make it better for the network to learn and interpret. In other words, use only those 30 second segments which make more sense and ignore all the other background noise. This could improve the accuracies.

7.4 Give Wavenet a shot

Wavenet is a deep generative model of raw audio waveforms. Since our EEG data is also a wave / signal, a wavenet model can be tried on it.

7.5 Wavelet transform

Wavelet transform can be used for artifact detection and signal processing.

Appendix A

Network pseudocode

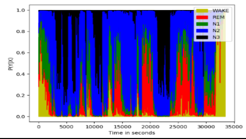
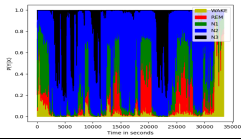
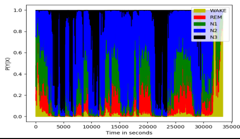
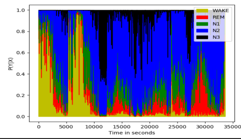
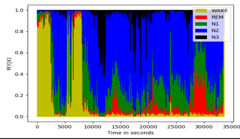
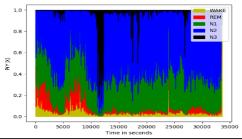
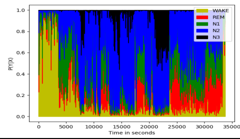
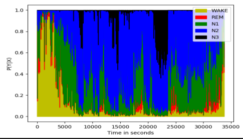
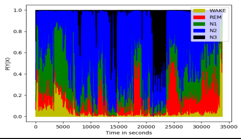
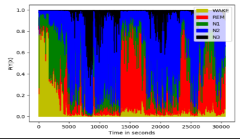
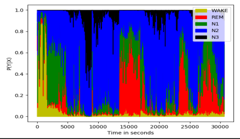
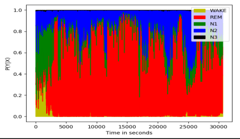
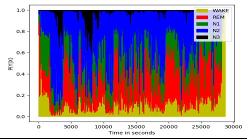
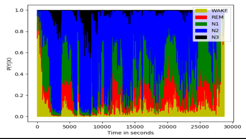
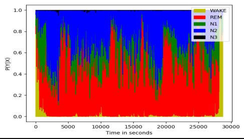
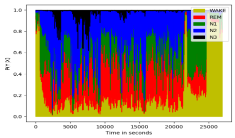
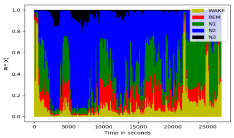
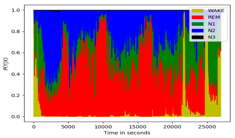
Below is the pseudo code for the CNN-LSTM network trained from scratch with dropout. The layers in bold are the one's taken for plotting MDS and GDV:

```
SEQUENTIAL MODEL ()
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 32, strides=1, input_shape=input_shape, activation='relu')) ........................................................................................Layer 1
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation='relu'))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation='relu'))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation='relu'))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation='relu'))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation='relu'))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation='relu'))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation=’relu’))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation=’relu’))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (4), filters = 64, strides=1, activation=’relu’))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (MaxPooling1D(pool_size=2))
ADD_TO_MODEL (Conv1D (kernel_size = (5), filters = 128, strides=5))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (BatchNormalization())
ADD_TO_MODEL (Bidirectional(LSTM(32, return_sequences=True , input_shape=input_shape)))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (BatchNormalization())
ADD_TO_MODEL (Bidirectional(LSTM(32, return_sequences=False , input_shape=input_shape)))
ADD_TO_MODEL (Dropout(0.2))
ADD_TO_MODEL (BatchNormalization())
ADD_TO_MODEL (Dense(n_classes, activation = ’softmax’,name=’softmax’))...................Layer 38
COMPILE_MODEL (loss=’categorical_crossentropy’, optimizer=’adam’,metrics = [’accuracy’])
```

Appendix B

Hypnodensity plots for the test set

Hypnodensity plots for 30s non-overlap for all the test set patients:

Patient	**Channel 1(C4)**	**Channel2 (F4)**	**Channel3 (O2)**
55			
56			
57			
58			
59			
60			

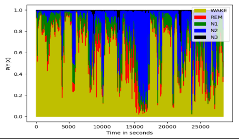
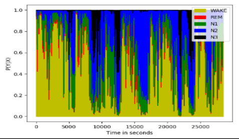
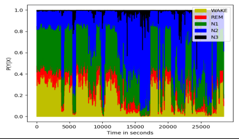
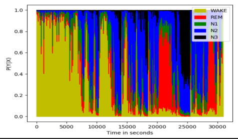
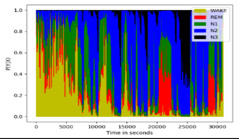
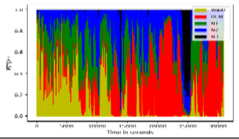
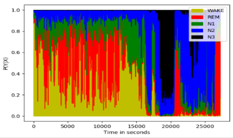
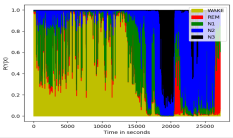
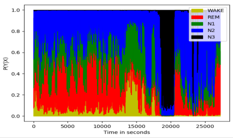
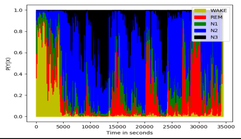
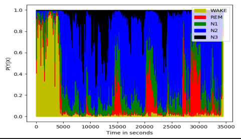
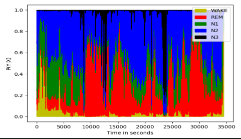
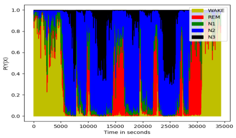
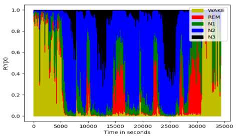
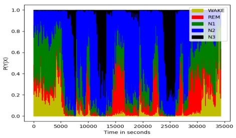
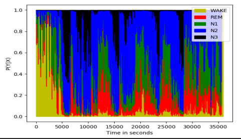
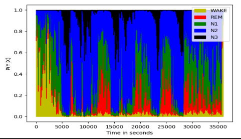
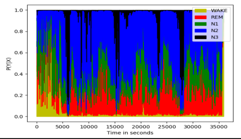
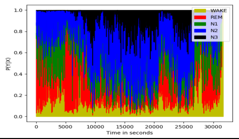
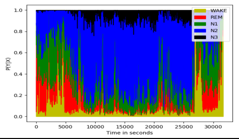
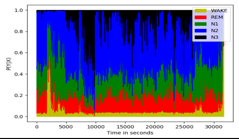
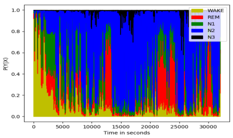
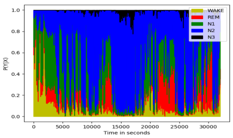
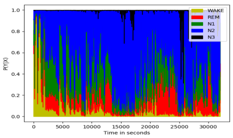
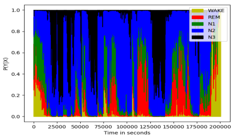
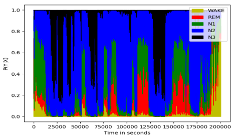
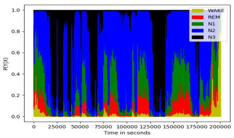
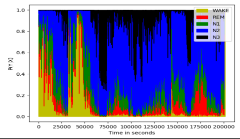
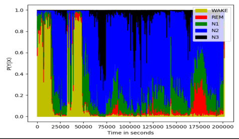
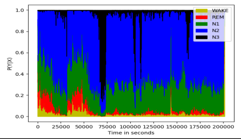
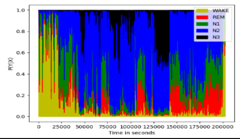
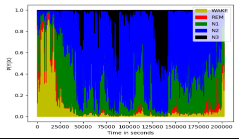
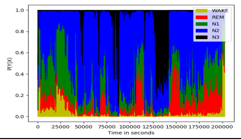
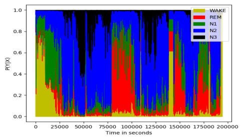
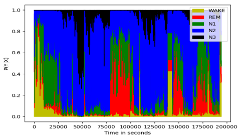
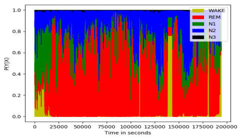
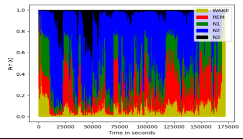
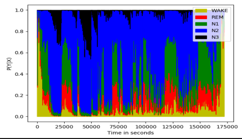
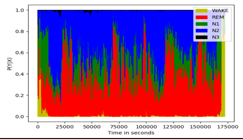
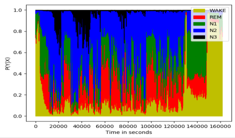
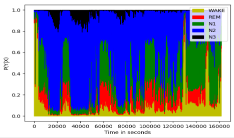
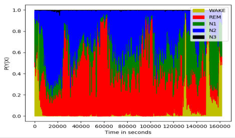
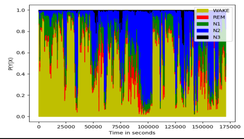
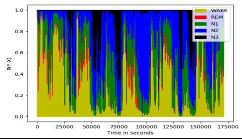
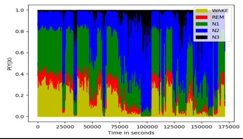
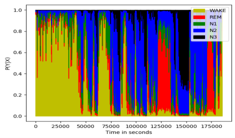
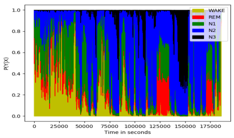
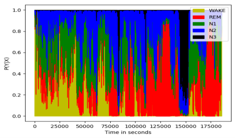
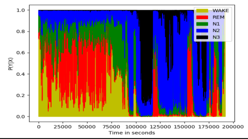
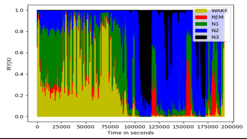
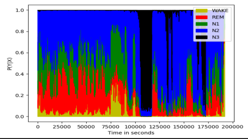
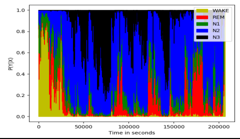
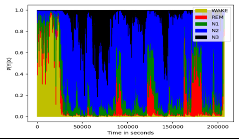
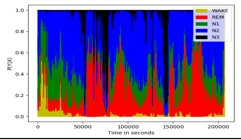
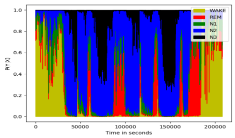
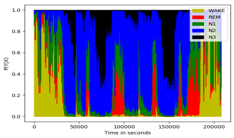
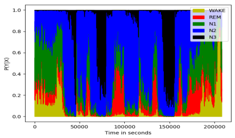
61			
62			
63			
64			
65			
66			
67			
68			

Table B.1: Hypnodensity plots without overlap

Hypnodensity plots for 5s overlap for all the test set patients:

Patient	Channel 1(C4)	Channel2 (F4)	Channel3 (O2)
55			

56			
57			
58			
59			
60			
61			
62			
63			
64			
65			

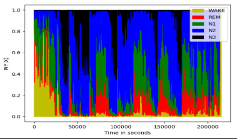
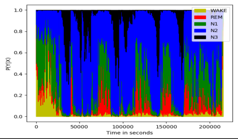
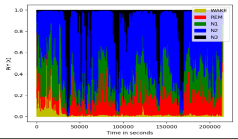
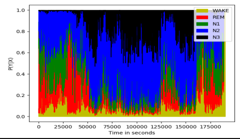
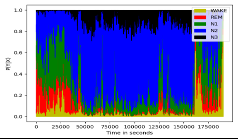
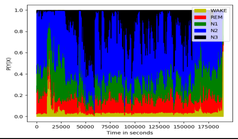
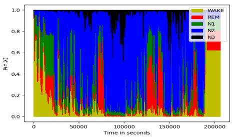
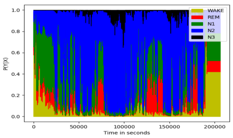
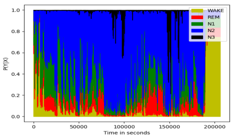
66			
67			
68			

Table B.2: Hypnodensity plots with overlap

Appendix C

VGG16 Network

C.1 Accuracy - Loss curve

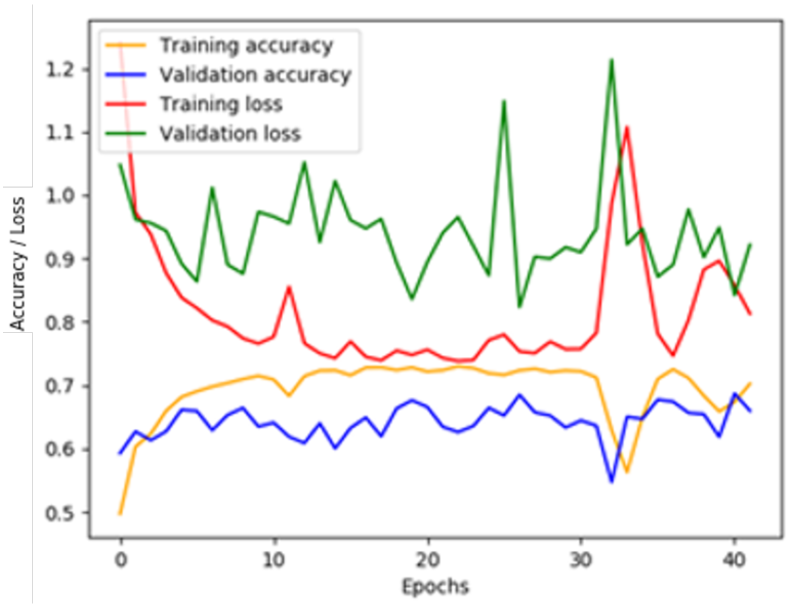
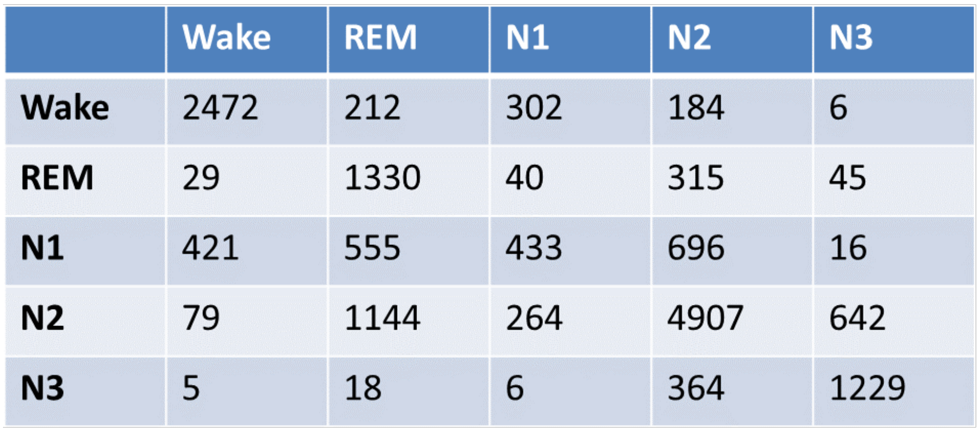


Figure C.1: Accuracy-Loss curves for VGG16

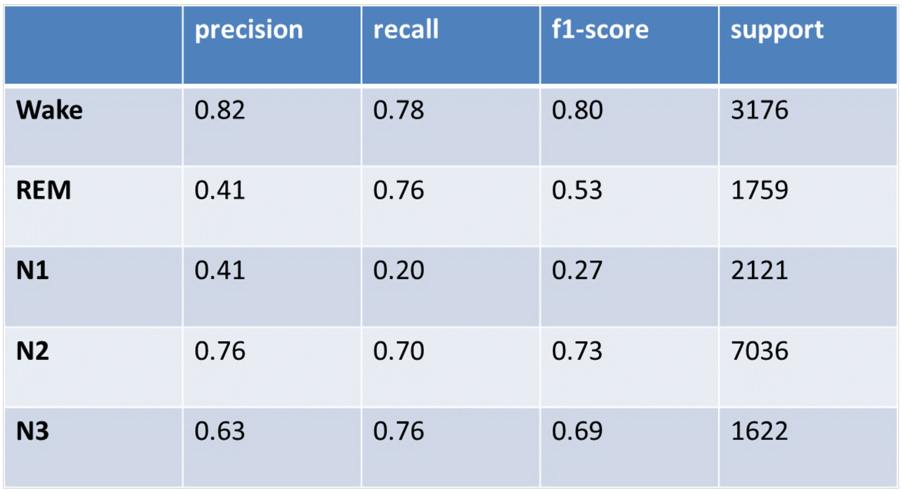
C.2 Confusion Matrix



	Wake	REM	N1	N2	N3
Wake	2472	212	302	184	6
REM	29	1330	40	315	45
N1	421	555	433	696	16
N2	79	1144	264	4907	642
N3	5	18	6	364	1229

Figure C.2: Confusion matrix for Vgg16

C.3 Classification Report



	precision	recall	f1-score	support
Wake	0.82	0.78	0.80	3176
REM	0.41	0.76	0.53	1759
N1	0.41	0.20	0.27	2121
N2	0.76	0.70	0.73	7036
N3	0.63	0.76	0.69	1622

Figure C.3: Classification Report for VGG16

C.4 Ground truth versus Predicted labels

Similarly, we plot test dataset hypnodensity plot:

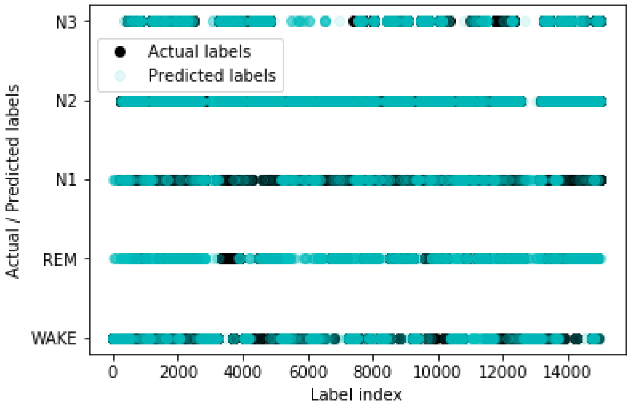


Figure C.4: Ground truth versus predicted labels for all test samples

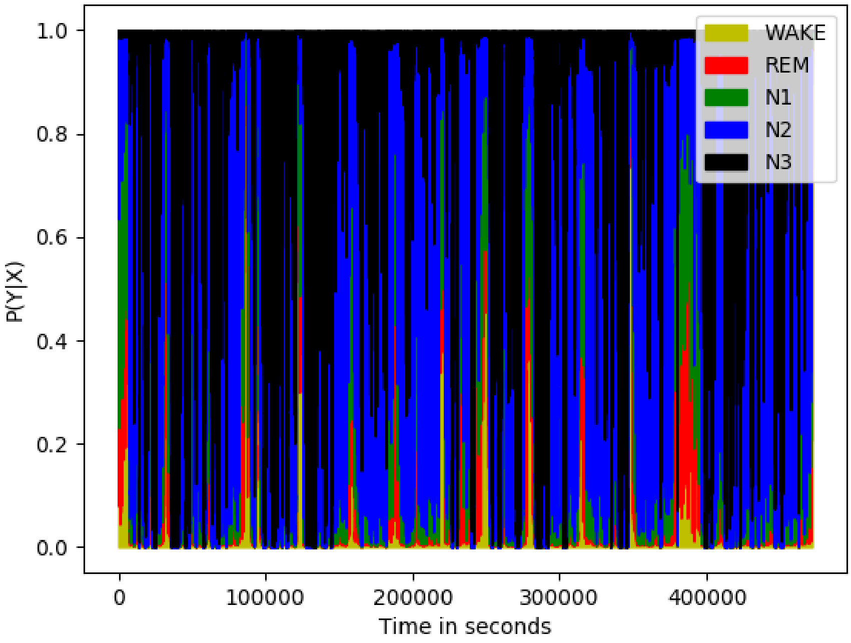


Figure C.5: Hypnodensity plot for patient 55-68 combined

C.5 Hypnodensity plots for non overlap and 5s overlap cases

’P’ stands for Patient.

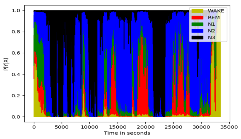
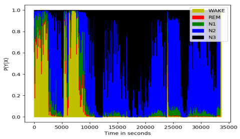
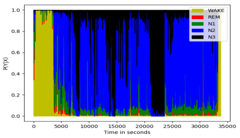
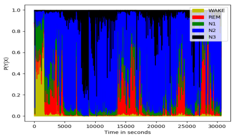
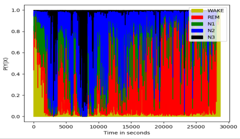
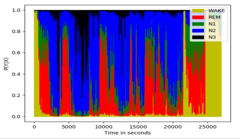
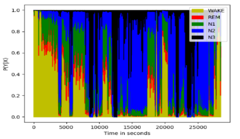
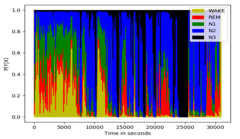
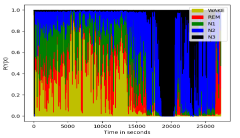
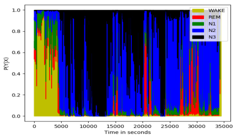
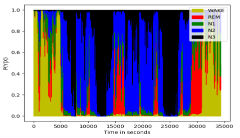
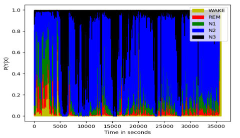
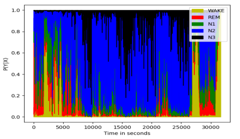
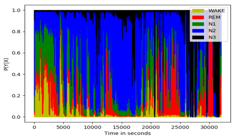
P 55 	P 56 
P 57 	P 58 
P 59 	P 60 
P 61 	P 62 
P 63 	P 64 
P 65 	P 66 
P 67 	P 68 

Table C.1: Hypnodensity plots without overlap

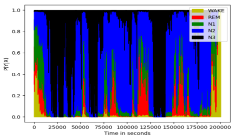
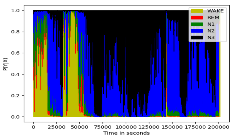
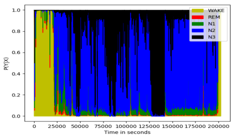
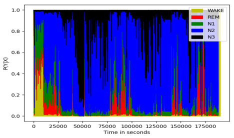
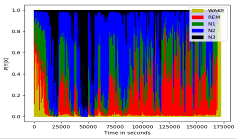
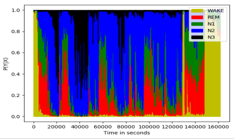
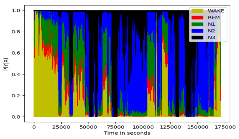
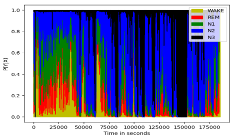
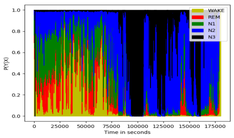
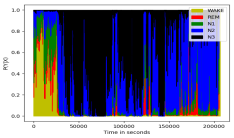
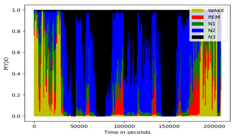
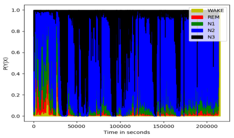
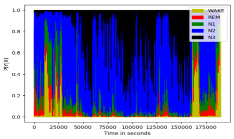
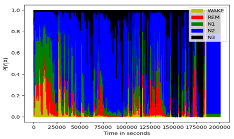
 P 55	 P 56
 P 57	 P 58
 P 59	 P 60
 P 61	 P 62
 P 63	 P 64
 P 65	 P 66
 P 67	 P 68

Table C.2: Hypnodensity plots with overlap

C.6 Pearson Correlation

Since we combine all the three channels in case of VGG16 network, it makes no sense to calculate Pearson correlation.

C.7 MDS plots

’L’ stands for Layer.

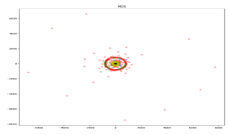
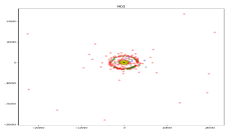
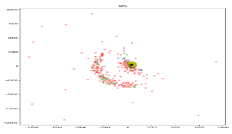
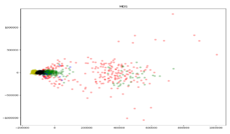
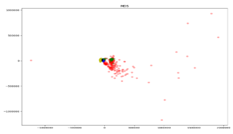
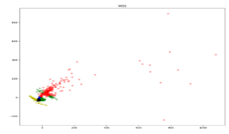
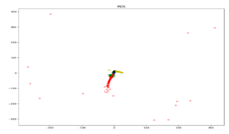
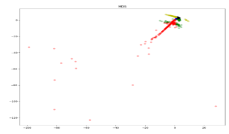
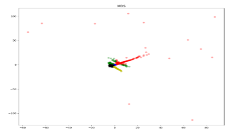
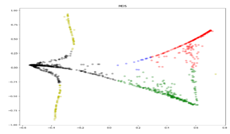
 L1	 L 4
 L 7	 L 11
 L 15	 L 19
 L 20	 L 21
 L 23	 L 25

Table C.3: MDS plots for Patient 1

C.8 GDV plot with error bars

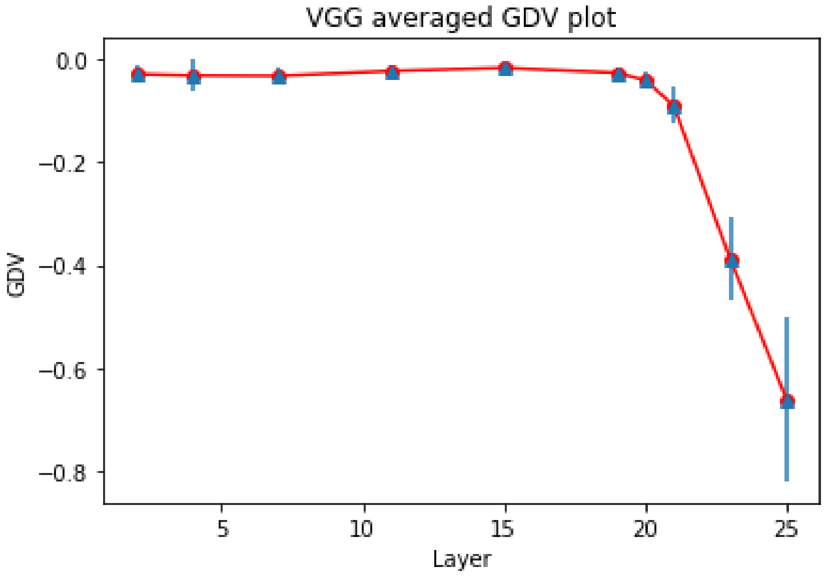


Figure C.6: Averaged GDV plot

List of Figures

1.1 Hypnogram for healthy human sleep overnight [Wis14] 2
1.2 Structure of a neuron [Com19] 2
1.3 Parts of the brain [Wik19a] 3
1.4 Electrode placement system for EEG [Cam12] 4

2.1 Vanilla Neural Network 6
2.2 Most common activation functions [xA19] 7
2.3 Working of a typical CNN [Des16] 9
2.4 Convolution Operation on Volume [Gas18] 9
2.5 Vgg16 layer architecture [uH18] 11
2.6 RNN architecture [Ola15] 12
2.7 One to many RNN [Wik19b] 12
2.8 LSTM architecture [Ola15] 13
2.9 LSTM architecture compressed version [Ola15] 14
2.10 Typical hypnogram recorded overnight [Gor11] 15
2.11 Typical hypnodensity plot obtained for nacroleptic and non-nacroleptic subject [Ste18] 16
2.12 Pearson Correlation coefficient significance [Tut19] 17
2.13 Typical MDS plot [Elh14] 18

3.1 AASM Manual [Ibe07] 22
3.2 Hypnogram plot for HMM approach [Dor07] 28

4.1 Workflow pipeline 31
4.2 Real data depiction 32
4.3 Data Preparation stage 33
4.4 Data visualization for patient 1 33

4.5 Data splitting stage 33
4.6 Final data representation 34
4.7 Data elimination stage 35
4.8 Why CNN-LSTM network ? 35
4.9 Neural network stage without dropout 36
4.10 Neural network stage with dropout 37
4.11 Numpy array concatenation without overlap for Patient 1 37
4.12 Numpy array concatenation with overlap for Patient 1 38
4.13 Three channel EEG data illusion 41
4.14 Data preparation for VGG16 network 42
4.15 VGG16 network 43

5.1 Accuracy versus epochs for CNN-LSTM network without dropout 47
5.2 Raw confusion matrix without normalization 48
5.3 (A) Row-wise normalized (B) Column-wise normalized confusion matrices . . . 48
5.4 Classification report for CNN-LSTM network without dropout 49
5.5 Accuracy versus epochs for CNN-LSTM network with dropout, channel 1 50
5.6 Raw confusion matrix without normalization, channel 1 51
5.7 (A) Row-wise normalized (B) Column-wise normalized (channel 1) confusion matrices 51
5.8 Classification report for CNN-LSTM network with dropout, channel 1 52
5.9 Actual and predicted labels for all test set samples, channel 1 52
5.10 Hypnodensity plot for patient 55-68 combined, channel 1 53
5.11 Accuracy versus epochs for CNN-LSTM network with dropout, channel 2 53
5.12 Raw confusion matrix without normalization, channel 2 54
5.13 Hypnodensity plot for patient 55-68 combined, channel 2 54
5.14 Hypnodensity plot for patient 55-68 combined, channel 3 55
5.15 Raw confusion matrix without normalization, channel 3 55
5.16 Hypnodensity plot for patient 55-68 combined, channel 3 56
5.17 Pearson correlations with lag time for patient 55 58
5.18 Pearson correlations with lag time for patient 63 59
5.19 Averaged GDV plots per channel 61

C.1 Accuracy-Loss curves for VGG16 75
C.2 Confusion matrix for Vgg16 76

C.3 Classification Report for VGG16 76
C.4 Ground truth versus predicted labels for all test samples 77
C.5 Hypnodensity plot for patient 55-68 combined 77
C.6 Averaged GDV plot 81

List of Tables

4.1 Numpy arrays of class probabilities for each patient 38
4.2 Channels for Pearson correlation with lag time 39
4.3 Numpy array arrangement for VGG16 network 42

5.1 Hypnodensity plots without overlap 56
5.2 Hypnodensity plots with 5 second overlap 57
5.3 Pearson correlations without lag time 57
5.4 MDS plots for Patient 1, Channel 1 60

6.1 Performance of different neural networks 63
6.2 Average GDV values for Network with dropout 66

B.1 Hypnodensity plots without overlap 72
B.2 Hypnodensity plots with overlap 74

C.1 Hypnodensity plots without overlap 78
C.2 Hypnodensity plots with overlap 79
C.3 MDS plots for Patient 1 80

Bibliography

[Abo16] Khald Ali I Aboalayon, Miad Faezipour, Wafaa S Almuhammadi, and Saeid Moslehpour. Sleep stage classification using eeg signal analysis: a comprehensive survey and new investigation. *Entropy*, 18(9):272, 2016.

[Ali18] Emina Alickovic and Abdulhamit Subasi. Ensemble svm method for automatic sleep stage classification. *IEEE Transactions on Instrumentation and Measurement*, 67(6):1258–1265, 2018.

[Ari12] Sayaka Aritake, Shigekazu Higuchi, Hiroyuki Suzuki, Kenichi Kuriyama, Minori Enomoto, Takahiro Soshi, Shingo Kitamura, Akiko Hida, and Kazuo Mishima. Increased cerebral blood flow in the right frontal lobe area during sleep precedes self-awakening in humans. *BMC neuroscience*, 13(1):153, 2012.

[Bet17] MD Bethesda. National library of medicine (us). *National Center for Biotechnology Information*, 2017.

[Boo17] Reza Boostani, Foroozan Karimzadeh, and Mohammad Nami. A comparative review on sleep stage classification methods in patients and healthy individuals. *Computer methods and programs in biomedicine*, 140:77–91, 2017.

[Cam84] Scott S Campbell and Irene Tobler. Animal sleep: a review of sleep duration across phylogeny. *Neuroscience & Biobehavioral Reviews*, 8(3):269–300, 1984.

[Cam12] Patrizio Campisi, Daria La Rocca, and Gaetano Scarano. Eeg for automatic person recognition. *Computer*, 45(7):87–89, 2012.

[Com19] Wikimedia Commons. File:neuron.svg — wikimedia commons, the free media repository, 2019. [Online; accessed 5-December-2019].

[Cri08] Shelly Crisler, Michael J Morrissey, A Michael Anch, and David W Barnett. Sleep-stage scoring in the rat using a support vector machine. *Journal of neuroscience methods*, 168(2):524–534, 2008.

[Des16] Adit Deshpande. A beginner's guide to understanding convolutional neural networks. *Retrieved March*, 31(2017), 2016.

[Dh09] Heidi Danker-hopfe, Peter Anderer, Josef Zeitlhofer, Marion Boeck, Hans Dorn, Georg Gruber, Esther Heller, Erna Loretz, Doris Moser, Silvia Parapatics, et al. Interrater reliability for sleep scoring according to the rechtschaffen & kales and the new aasm standard. *Journal of sleep research*, 18(1):74–84, 2009.

[Dor07] LG Doroshenkov and VA Konyshev. Usage of hidden markov models for automatic sleep stages classification. In *Russian-Bavarian Conference on Bio-Medical Engineering*, page 19. Citeseer, 2007.

[Elh14] Nuha Elhassan, Eyoab Iyasu Gebremeskel, Mohamed Ali Elnour, Dan Isabirye, John Okello, Ayman Hussien, Dominic Kwiatksowski, Jibril Hirbo, Sara Tishkoff, and Muntaser E Ibrahim. The episode of genetic drift defining the migration of humans out of africa is derived from a large east african population size. *PloS one*, 9(5):e97674, 2014.

[Fer08] Raffaele Ferri, Mauro Manconi, Giuseppe Plazzi, Oliviero Bruni, Stefano Vandi, Pasquale Montagna, LUIGI FERINI-STRAMBI, and Marco Zucconi. A quantitative statistical analysis of the submentalis muscle emg amplitude during sleep in normal controls and patients with rem sleep behavior disorder. *Journal of sleep research*, 17(1):89–100, 2008.

[Gas18] Dina Gasong. *Belajar dan pembelajaran*. Deepublish, 2018.

[Gor11] Shery Goril. *The patterns of sleep disorders and circadian rhythm disruptions in children and adolescents with Fetal Alcohol Spectrum Disorders*. PhD thesis, 2011.

[Han01] Masaaki Hanaoka, Masaki Kobayashi, and Haruaki Yamazaki. Automated sleep stage scoring by decision tree learning. In *2001 conference proceedings of the 23rd annual international conference of the IEEE engineering in medicine and biology society*, volume 2, pages 1751–1754. IEEE, 2001.

[Ibe07] Conrad Iber and Conrad Iber. *The AASM manual for the scoring of sleep and associated events: rules, terminology and technical specifications*, volume 1. American Academy of Sleep Medicine Westchester, IL, 2007.

[Lia15] Sheng-Fu Liang, Chih-En Kuo, Fu-Zen Shaw, Ying-Huang Chen, Chia-Hu Hsu, and Jyun-Yu Chen. Combination of expert knowledge and a genetic fuzzy inference system for automatic sleep staging. *IEEE Transactions on Biomedical Engineering*, 63(10):2108–2118, 2015.

[Lou04] Rhain P Louis, James Lee, and Richard Stephenson. Design and validation of a computer-based sleep-scoring algorithm. *Journal of neuroscience methods*, 133(1-2):71–80, 2004.

[Mar72] WB Martin, LC Johnson, SS Viglione, P Naitoh, RD Joseph, and JD Moses. Pattern recognition of eeg-eog as a technique for all-night sleep stage scoring. *Electroencephalography and clinical neurophysiology*, 32(4):417–427, 1972.

[Mos09] Doris Moser, Peter Anderer, Georg Gruber, Silvia Parapatics, Erna Loretz, Marion Boeck, Gerhard Kloesch, Esther Heller, Andrea Schmidt, Heidi Danker-Hopfe, et al. Sleep classification according to aasm and rechtschaffen & kales: effects on sleep scoring parameters. *Sleep*, 32(2):139–149, 2009.

[Mou19] Sajad Mousavi, Fatemeh Afghah, and U Rajendra Acharya. Sleepeegnet: Automated sleep stage scoring with sequence to sequence deep learning approach. *PloS one*, 14(5):e0216456, 2019.

[Ola15] Christopher Olah. Understanding lstm networks. 2015.

[Qas18] Hussam Qassim, Abhishek Verma, and David Feinzimer. Compressed residual-vgg16 cnn model for big data places image recognition. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 169–175. IEEE, 2018.

[Ryt11] Kirsi-Marja Rytkönen, Jukka Zitting, and Tarja Porkka-Heiskanen. Automated sleep scoring in rats and mice using the naive bayes classifier. *Journal of neuroscience methods*, 202(1):60–64, 2011.

[Sch18] Achim Schilling, Claus Metzner, Jonas Rietsch, Richard Gerum, Holger Schulze, and Patrick Krauss. How deep is deep enough?–quantifying class separability in the hidden layers of deep neural networks. *arXiv preprint arXiv:1811.01753*, 2018.

[Ste18] Jens B Stephansen, Alexander N Olesen, Mads Olsen, Aditya Ambati, Eileen B Leary, Hyatt E Moore, Oscar Carrillo, Ling Lin, Fang Han, Han Yan, et al. Neural network analysis of sleep stages enables efficient diagnosis of narcolepsy. *Nature communications*, 9(1):5229, 2018.

[Tut19] SPSS Tutorials. Pearson correlations–quick introduction. *Hentet fra https://www. spss-tutorials. com/pearson-correlation-coefficient*, 2019.

[uH18] Muneeb ul Hassan. Vgg16–convolutional network for classification and detection. *Neurohive. Dostopno na: https://neurohive. io/en/popular-networks/vgg16/[10. 4. 2019]*, 2018.

[Wan12] Jeen-Shing Wang, Guan-Rong Shih, and Wei-Chun Chiang. Sleep stage classification of sleep apnea patients using decision-tree-based support vector machines based on ecg parameters. In *Proceedings of 2012 IEEE-EMBS International Conference on Biomedical and Health Informatics*, pages 285–288. IEEE, 2012.

[Wik19a] Wikipedia contributors. Human brain — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Human$_{b}rainoldid = 928929959, 2019. [Online; accessed5 - December - 2019]$.

[Wik19b] Wikipedia contributors. Sarah (cheetah) — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Sarah$_{(}cheetah)oldid = 904109809, 2019. [Online; accessed5 - December - 2019]$.

[Wis14] Malgorzata Wislowska, Renata del Giudice, and Manuel Schabus. Searching for neuronal correlates of consciousness-what do we learn from coma and disorders of consciousness. *Functional Neurology, Rehabilitation, and Ergonomics*, 4(2/3):135, 2014.

[xA19] xristica Aporra. activation functions. *QuantDare, Madrid, Spain*, 2019.

[Yan16] Shi Yan. Understanding lstm and its diagrams. *Software engineer & wantrepreneur. Interested in computer graphics, bitcoin and deep learning*, 13(03), 2016.

[Yil19] Ozal Yildirim, Ulas Baran Baloglu, and U Rajendra Acharya. A deep learning model for automated sleep stages classification using psg signals. *International journal of environmental research and public health*, 16(4):599, 2019.

[Zha16] Junming Zhang, Yan Wu, Jing Bai, and Fuqiang Chen. Automatic sleep stage classification based on sparse deep belief net and combination of multiple classifiers. *Transactions of the Institute of Measurement and Control*, 38(4):435–451, 2016.