CLASSIFICATION FOR EYE STATE USING UNSUPERVISED LEARNING AND EEG

by

DURGANSHU MISHRA

(Roll No. - 2016086**)**

RISHABH BAJPAI

(Roll No. - 2016207**)**

Supervisor

DR. DEEPMALA

Assistant Professor, Department of Natural Sciences,

PDPM IIITDM, Jabalpur

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, DESIGN AND MANUFACTURING JABALPUR

MTH612 PROJECT REPORT

CONTENTS

1. INTRODUCTION

Electroencephalography (EEG) is a non-invasive technology used for recording the activities of brain from the scalp. For recoding the activities of brain, some electrodes are placed at different locations on the scalp. Generally, an EEG device has more than one electrode, which makes the data multi-dimensional. This multi-dimensional data has both temporal and special properties. But due to difficulty in the representation of data in matrix form and problems in processing multi-dimensional data, merely temporal information is used by researchers.

Some fundamental aspects of EEG are presented below:

1.1 How is EEG data construed?

Neurons are the cells responsible for transduction, transmission, storage and processing of the relevant data for the survival of a living being (however, not all living being have neurons ex. plants, Sponge). There are many types of neurons and they are present in almost all parts of the human body. On the basis of their direction of transfer of information, neurons can be classified into three classes: Sensory (or afferent), Motor (or efferent), and Intermediate neurons.

1. Sensory neurons: respond to sensory stimuli and send information from sensory organs (e.g., in skin, eyes, nose, tongue, ears) TOWARD the central nervous system (The CNS is the part of the nervous system consisting primarily of the brain and spinal cord). They perform a very special operation of transduction of external stimuli into electrical signals.
2. Motor neurons: send information AWAY from the CNS to muscles or glands. In other words, they receive signals from the brain and spinal cord to control everything from muscle contractions to glandular output.
3. Interneurons: send information between sensory neurons and motor neurons. Most interneurons are located in the central nervous system.

These cells communicated with each other to perform various operations. And this collection of communicating neurons is known as a neural network (it was the inspiration of present Artificial neural networks: ANN). Each cell in the network performs its specific function by processing the upcoming data and transferring the resultant signal to other neurons. By collective processing of neurons in a neural network, highly complex computations are possible. [1]

The cumulative sum of activities of billions of neurons gives rise to behavior of an animal. And the best thing about these signals is that we can record them from scalp with non-invasive techniques. Two most popular such techniques among researchers are Electroencephalography (EEG) and functional Magnetic Resonance Imaging (fMRI).

In EEG, the potential difference between electrodes is recorded and a matrix is constructed. Row of the matrix represents channels (potential difference between electrodes), and column represents time points. So, dimension of data matrix for an EEG device recording data of 5 seconds having sampling frequency of 256 Hz with 128 electrodes will be 128x1280 (5*256=1280).

1.2 How is EEG data analysed?

Analysis of EEG data is quite challenging due to the low signal to noise ratio. Signal processing, artefact detection and attenuation, feature extraction, feature selection and computation of mental metrics such as workload, engagement, drowsiness, or alertness all require a specific degree of aptitude and experience to recognise and separate significant data from the gathered information appropriately.

Generally, an EEG device has more than one electrode, which makes the data multi-dimensional. This multi-dimensional data has both temporal and spatial properties. But due to difficulty in the representation of data in matrix form and problems in processing multi-dimensional data, merely temporal information is used by researchers.

1.3 Applications of EEG

Brain signals have various applications in varied fields. Some of the areas are listed below:

1. **Medical**
   For diagnosis and treatment of various mental disorders and diseases, recording and external stimulation of brain signals are being used. In future, we can expect better decoding of the activities of the brain and control on the stimulation of brain regions and it will open new doors for the treatment of many diseases.
   An interested reader can go through the given links:
   a) Diagnosis of depression using EEG
   b) Diagnosis of Human Psychological Disorders using Supervised Learning and Nature-Inspired Computing Techniques
   c) Parkinson's disease
   d) Resting-State fMRI

2. **Robotics**
   To improve the Human-Robot Interaction (HRI), brain recording can play an important role. The biggest challenge faced by a robot/AI while interacting with a human is to predict/understand human behaviour. However, by using facial expressions, skin conductance, EEG, etc. as a tool for predicting human behaviour, new natural way of interaction techniques can be developed, which will eventually improve HRI.

3. **Smart Systems**
   Brain-Computer Interface (BCI) is used in various fields and providing promising results. These BCI can be integrated with other systems to improve their performances. And due to improvements in the decoding techniques, in future, we can expect robust brain-controlled smart systems such as assistive health monitoring app, and personal assistance.

4. **Exoskeleton and prosthesis**
   Hybrid BCI is an interface which uses a multi-modal approach to solve the given problem of BCI. For example, prosthetic limbs can be efficiently controlled with the help of a combination of EEG and EMG (Electromyography) signals. An exciting future application of EEG is in the development of robust exoskeletons.

5. **Intelligent systems and brain-inspired Artificial Intelligence (AI)**
   Nature always serves as an inspiration for developing intelligent systems. For example, a simple mathematical model of a neuron is similar to Perceptron: Perceptron was invented in 1958 at the Cornell Aeronautical Laboratory by Frank Rosenblatt. In later years people developed a multi-layered model of perceptrons, and then advanced machine learning algorithms like neural networks and finally deep neural networks. [2]

The data matrix explained in section 1.1 is used for analysing the EEG data. The first the EEG data is pre-processed by removing the low and high-frequency noises, removing outliers, removing bad channels and components of noise by using algorithms like independent component analysis (ICA). Then the data is analysed according to the requirements of the task. For analysing event-related potentials, the data is converted into time epochs, and then the averaged data matrix is constructed. On the other hand, for performing steady-state visual evoked potential (SSVEP), the data is processed in the frequency domain.

The problem of eye state classification is usually solved using supervised methods. The neural network is trained on a set of data and tested on the other set of data for performance parameters. However, we used a different approach to solve this problem. We used unsupervised learning and developed a robust system for eye state classification.

Here, we present a method that converts the EEG data to a latent space, where each EEG sample is represented with a latent vector. We used an auto-encoder (a neural network), a deep-learning based approach for conversion of representation of data and k-mean algorithms for clustering. Our approach uses various concepts from Topology, neurophysiology, and deep learning. In section 1 we explained the fundamental of EEG, in section 2 we mentioned a few previous works related to the topic, in section 3 we present overview of the proposed method, in section 4 we explained the methodology used by us, and in section 5 and 6 we presented, discussed and concluded the results.

2. LITERATURE REVIEW

2.1 On eye state classification and neural networks

Some research work has already been done for eye state prediction, identification or detection in facial images and visual recordings too. Eye state classification is very crucial to medical care but also significant for daily life chores. It is a kind of common time series problem for detecting human cognitive state. There are various application areas where EEG eye state classification task is the central element, such as epileptic seizure detection, stress feature identification, driving drowsiness detection, infant sleep-waking state identification.

Samadi *et al.* [3] proposed an EEG based eye tracking solution by using the information from two different sources, i.e., Head mounted Video - Oculography and 16-channeled EEG signal. Their proposed model achieved the accuracy of 97.57% by extracting the features from source by using ICA rather than band-pass pre-processed EEG signals. A novel machine learning approach, incremental attribute learning (IAL) is proposed by Wang *et al.* [4] for EEG eye state time series classification. The IAL algorithm progressively imports and trains features one at a time to predict the class labelling.

The focus of Narejo *et al.* [5] is to implement deep learning architectures as classification models; specifically, Deep Belief Network (DBN) and Stacked Auto Encoder (SAE) and provide comparative analysis on the behavior and performance of SAE and DBF. Subsequently, comparing the obtained results with conventional Machine learning models of earlier studies. Deep learning is achieving state-of-the-art results across a range of challenging problem domains. This research work is probably the first application of Eye state prediction of EEG signals using deep architectures.

The Wen *et al.* [6] have proposed an auto-coding framework - based deep network by combining it with convolution and deconvolution in order to perform unsupervised feature learning from EEG signals. This model can effectively learn low- dimensional features from high-dimensional EEG data in order to help classifier achieve a higher detection accuracy and faster speed.

2.2 On clustering and metrization of topological spaces

Clustering is one of the most fundamental problems in data analysis and it has been studied extensively in the literature. Though many clustering algorithms have been proposed, clustering theories that justify the use of these clustering algorithms are still unsatisfactory. Further, it is a basic requirement to have the topological space be metric (the metrization problem) in order to perform clustering.

Chang *et al.* [7] have addressed two essential questions. Firstly, *what is a cluster in a set of data points?* In order to address such a question, the authors have considered a set of data points associated with a distance measure (metric). They have proposed a new cohesion measure in terms of the distance measure. Subsequently, this cohesion measure has been used to define a cluster as a set of points that are cohesive to themselves.

Secondly, *how do we find clusters and good partitions of clusters under such a definition?*

The authors have proposed a hierarchical agglomerative algorithm and a partitional algorithm, called the K-sets algorithm. Unlike standard hierarchical agglomerative algorithms, the hierarchical agglomerative algorithm proposed here has a specific stopping criterion and it stops with a partition of clusters. The K-sets algorithm is a new algorithm and it only takes one-step

minimisation in contrary to the Lloyd iteration that needs two-step minimisation. Further, they have proposed a duality result between a distance measure and a cohesion measure. The dual K-sets algorithm converges in the same way as a sequential version of the classical kernel K-means algorithm.

The Metrization Problem: The problem is to state in terms of the concepts point, and point of accumulation the conditions that a topological space be metric.

Chittenden [8] gave three solutions of the general metrization problem; the first contains a modified form of the enclosable property of Hedrick, the second is due to Alexandroff and Urysohn, and the third is based upon the notion of coherence introduced by Pitcher and Chittenden. These three sets of conditions are alike in requiring the existence of a type of development Δ of fundamental importance which it is proposed to call *regular*.

In Clustering algorithms, fuzzy c-means (FCM) clustering and k-means clustering are the most used algorithm. Different Neural activities are analysed using EEG signal and are classified by different classifiers. The input data set is clustered using K-means clustering and fuzzy c-means (FCM) clustering and is input to the neural network.

Pradhan *et. al.* [9] have proposed spatial clustering algorithm for EEG signals to improve the efficiency. The results obtained have been compared and analysed against K-means algorithm and FCM algorithms.

The *k*-means++ approach has been applied since its initial proposal. Shindler [10] has reviewed many types of clustering algorithms, the method is said to successfully overcome some of the problems associated with other ways of defining initial cluster-centres for *k*-means clustering.

3. PROPOSED WORK

Our proposed model consists of two parts:

3.1 Conversion of high-dimensional EEG data to low dimensional latent space

For efficient processing of information-rich data, dimension reduction is a crucial step. But designing an efficient dimension reduction method is itself a complicated task. The reduced dimensional representation should have characteristic properties of the data. Further, it should be able to reproduce the high-dimensional representation of the data without many perturbations.

Recently, autoencoders showed promising results for data compression. We have used this property of autoencoder for reducing dimensions of the data such that it can be considered as Euclidean space and can be used for clustering.

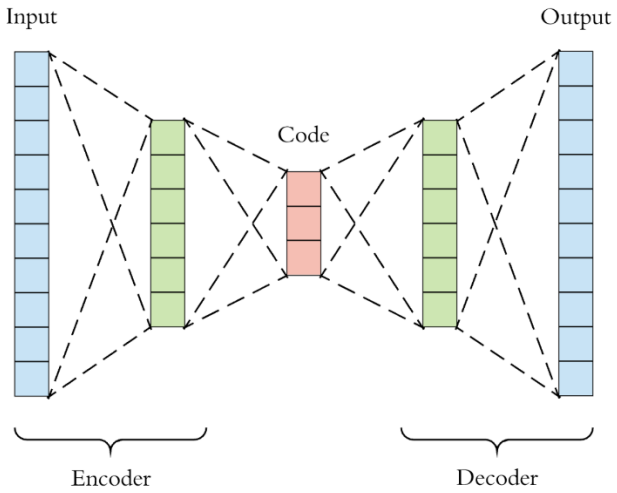


Fig. 1 Compressive autoencoder is trained to convert high dimensional data (Input) to low dimensional latent space (Code) and reproduce the data (Output) from latent space.

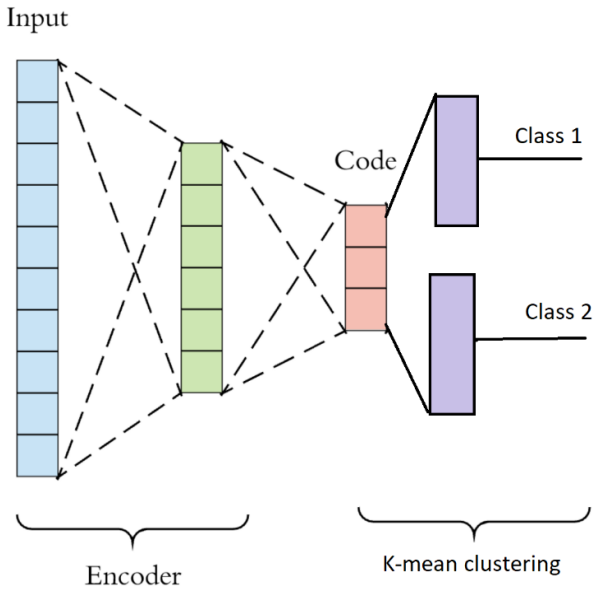


Fig. 2 Proposed method

Let us first try to understand mathematics behind the concept of convolutional autoencoder.

3.1.1 Convolution

Convolution is an operation applied on a signal to get transformed and relevant signal. For convolution a filter or LTI (Linear and Time-Invariant) is required which is multiplied and translated on the signal to get new signal values.

3.1.1.1 1D convolution

If we have a signal $s(t)$ passing through a system with impulse response $h(t)$, the output is the convolution of $s(t)$ with $h(t)$. The convolution is simply the integral of the product of the two functions (in this example the functions are $s(t)$ and $h(t)$), where one is reversed.

$$(s * h)(t) = \int_0^\infty s(\tau)h(t-\tau)d\tau$$

3.1.1.2 2D convolution

The convolution can generalize to more than one dimension. Two-dimensional (2D) convolutions are also extremely useful, for example in image processing. An image is a 2D signal and can be the input to a 2D filter as well. If an image is a signal $s(x, y)$, where (x, y) is the position of a pixel, and $h(x, y)$ is the kernel of a filter, the convolution is:

$$(s * h)(x, y) = \int_x \int_y s(\alpha, \beta)h(x-\alpha, y-\beta)\partial\beta\partial\alpha$$

3.1.1.3 Convolution using neural networks

Convolution can be applied on the signal using neural networks. For understanding consider a 2D input data with dimensions 32*32, and filters having dimensions 3*3. Now the first layer of neural network will have 1024 neurons for representing input data and the next layer will have 1022 neurons. And the connections between these layers will have sparseness. For each filter there are 9+1 parameters for training as the filter weights need to be trained (see equation mentioned below).

$$w^T x + b$$

Here w is weight matrix, x is signal and b is bias term.

The output for the rth neuron of the second layer is represented by the following equation:

$$h^r = \sum_{ijk} x^r{}_{ijk} W_{ijk} + b$$

3.1.2 Neural Network

In neural networks, neurons are connected with each other and produce their output by processing the data coming to them. A simple neuron model is shown below.

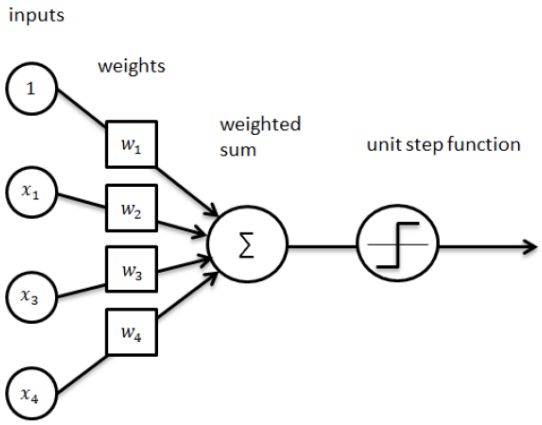


Fig. 3 Mathematical model of perceptron

During training, involves finding the minimun value of a loss function and it is a complicated problem.

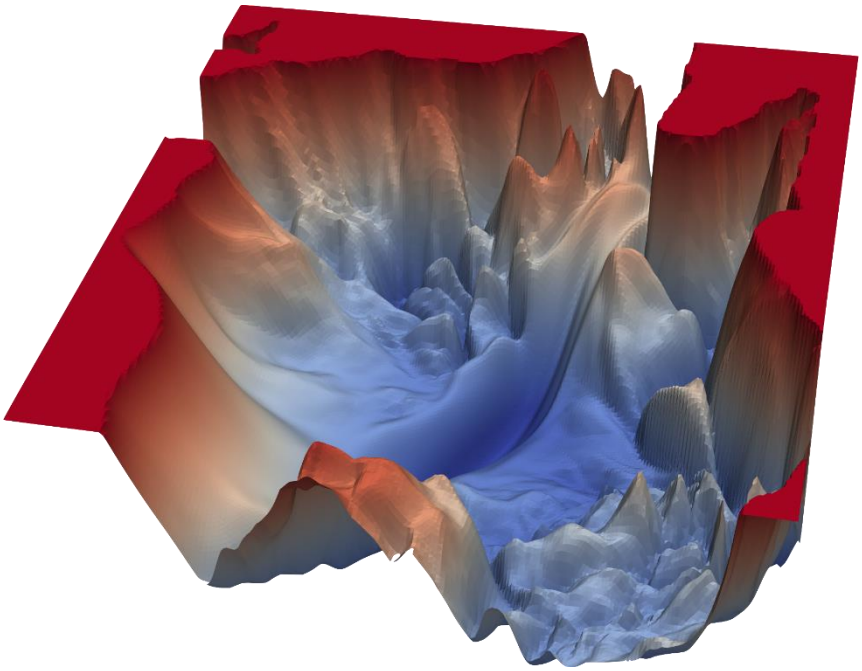


Fig.4 A sample loss function in 3D space is shown. From the figure, we can estimate the complexity of the problem. Any classification problem can be represented with such loss function, and at the point of minima, in the curve the has maximum accuracy. During the training of a neural network, optimizer like Adam, SGD can be used to find minima of the loss function. These optimizers backpropagate the error to the network and update the weights of the neurons. After many iterations, finally, the network converges.

During the training of the neural netwok a Hessian Matrix is computed for each step. A Hessian Matrix accumulates all these gradients which are needed for updatetion of the weights of the network.

We used Adam (Adaptive Moment Optimization) algorithm that combines the heuristics of both Momentum and RMSProp. Equetions used are mentioned below.

$$\nu_t = \beta_1 * \nu_{t-1} - (1 - \beta_1) * g_t$$
$$s_t = \beta_2 * s_{t-1} - (1 - \beta_2) * g_t^2$$
$$\Delta\omega_t = -\eta \frac{\nu_t}{\sqrt{s_t + \epsilon}} * g_t$$
$$\omega_{t+1} = \omega_t + \Delta\omega_t$$

$\eta : Initial\ Learning\ rate$
$g_t : Gradient\ at\ time\ t\ along\ \omega^j$
$\nu_t : Exponential\ Average\ of\ gradients\ along\ \omega_j$
$s_t : Exponential\ Average\ of\ squares\ of\ gradients\ along\ \omega_j$
$\beta_1, \beta_2 : Hyperparameters$

3.2: Clustering of the converted data

K-means clustering is a partitioning method. The proposed model partitions data into k mutually exclusive clusters and returns the index of the cluster to which it assigns each observation. It treats each observation in the data as an object that has a location in space. The function finds a partition in which objects within each cluster are as close to each other as possible, and as far from objects in other clusters as possible. k-means clustering requires the user to specify the number of clusters k before clustering.

Each cluster in a k-means partition consists of member objects and a centroid (or center). In each cluster, it minimizes the sum of the distances between the centroid and all member objects of the cluster. It computes centroid clusters differently for the supported distance metrics.

Here, the k-means++ algorithms have been used to initialize cluster centroids, and the squared Euclidean distance metric to determine distances.

4. METHODOLOGY & RESULTS

4.1 Conversion of high-dimensional EEG data to low dimensional latent

The dataset we used in the study was downloaded from NelsonGon's kaggle repository [11]. The data includes 117 sec EEG recoding from 14 electrodes. The sampling frequency of the data is 256Hz.

First, we converted the data into 201 epochs with 4 sec (4*256=1024 time points) total window size and 0.5 sec window sift. Thus, we have 201 samples with 14*1024 dimensions each. We trained a convolutional auto encoder as shown below:

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 14, 1024, 1)	0
conv2d_1 (Conv2D)	(None, 14, 1024, 128)	1280
max_pooling2d_1 (MaxPooling2	(None, 14, 512, 128)	0
conv2d_2 (Conv2D)	(None, 14, 512, 64)	73792
max_pooling2d_2 (MaxPooling2	(None, 14, 256, 64)	0
conv2d_3 (Conv2D)	(None, 14, 256, 32)	18464
max_pooling2d_3 (MaxPooling2	(None, 14, 128, 32)	0
conv2d_4 (Conv2D)	(None, 14, 128, 8)	2312
max_pooling2d_4 (MaxPooling2	(None, 14, 64, 8)	0
conv2d_5 (Conv2D)	(None, 14, 64, 2)	98
max_pooling2d_5 (MaxPooling2	(None, 14, 32, 2)	0
conv2d_6 (Conv2D)	(None, 14, 32, 2)	26
up_sampling2d_1 (UpSampling2	(None, 14, 64, 2)	0
conv2d_7 (Conv2D)	(None, 14, 64, 8)	152
up_sampling2d_2 (UpSampling2	(None, 14, 128, 8)	0
conv2d_8 (Conv2D)	(None, 14, 128, 32)	2336
up_sampling2d_3 (UpSampling2	(None, 14, 256, 32)	0
conv2d_9 (Conv2D)	(None, 14, 256, 64)	18496
up_sampling2d_4 (UpSampling2	(None, 14, 512, 64)	0
conv2d_10 (Conv2D)	(None, 14, 512, 128)	73856
up_sampling2d_5 (UpSampling2	(None, 14, 1024, 128)	0
conv2d_11 (Conv2D)	(None, 14, 1024, 1)	1153

==========

So, 14*1024=14336 are converted into 14*2=28 dimensions in latent space and then again converted to 14336 dimensions as output data. We trained the network on 160 samples and validated on 41 samples with 1 as batch size and 100 epochs. The training curve is shown in Fig. 4.

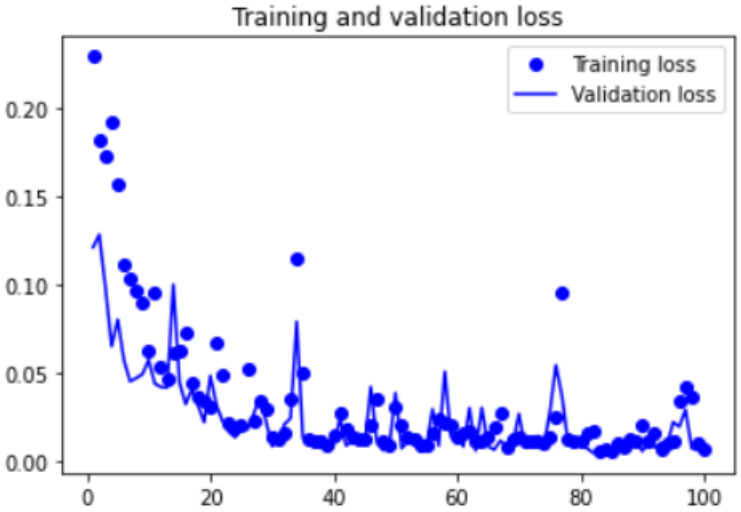
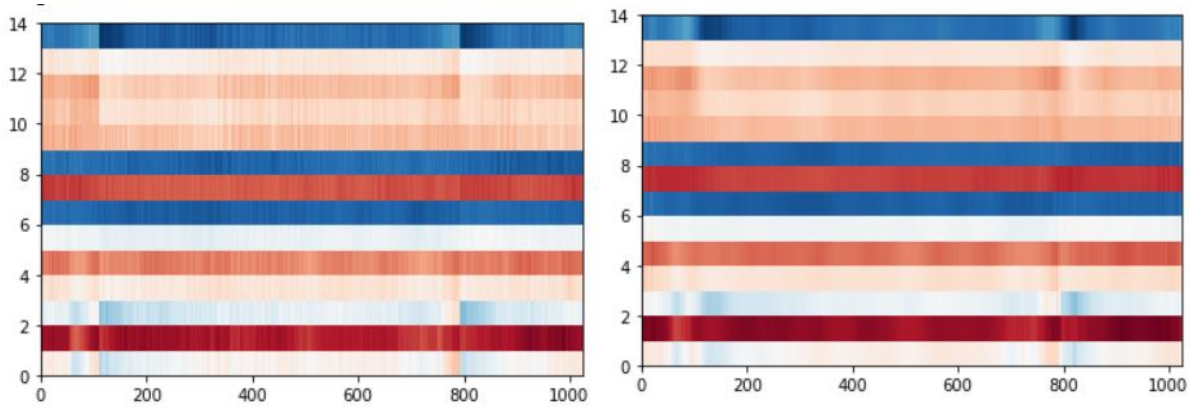


Fig. 4 Training and validation loss during the training of autoencoder



a) **b)**

Fig. 5 a) An input sample and b) an output sample generated from autoencoder

From Fig. 5 we can see that the 28 dimensions of the latent vector are able to capture most of the information of EED sample of 14336. Therefore, we used these 28 dimensions for clustering.

Before doing clustering, we tried to visualise the data (28 dimensions) and tried to understand how many clusters are required for classification. So, we performed Principal Component Analysis (PCA) on the data. Fig. 6 shows how Cumulative explained variance is varying with the number of components.

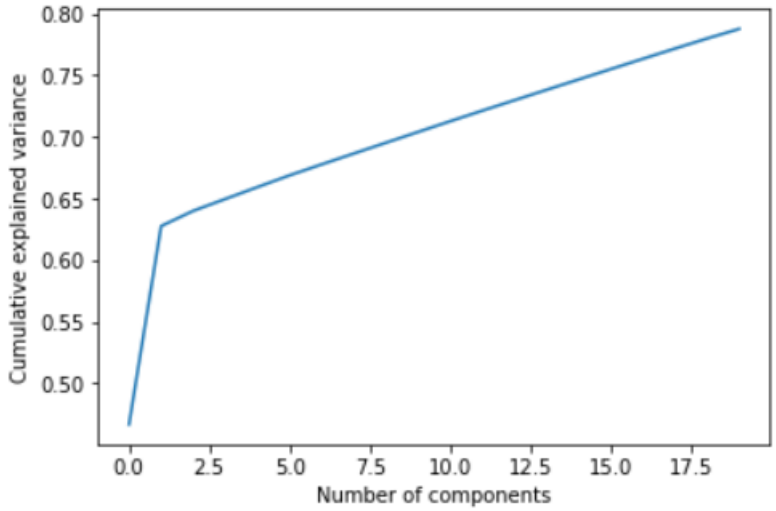


Fig. 6 Cumulative explained variance vs the number of components curve

From Fig. 6 it can be seen that 2 number of components can represent more than 62% information of the data. Thus, we performed PCA with two components (see Fig. 7).

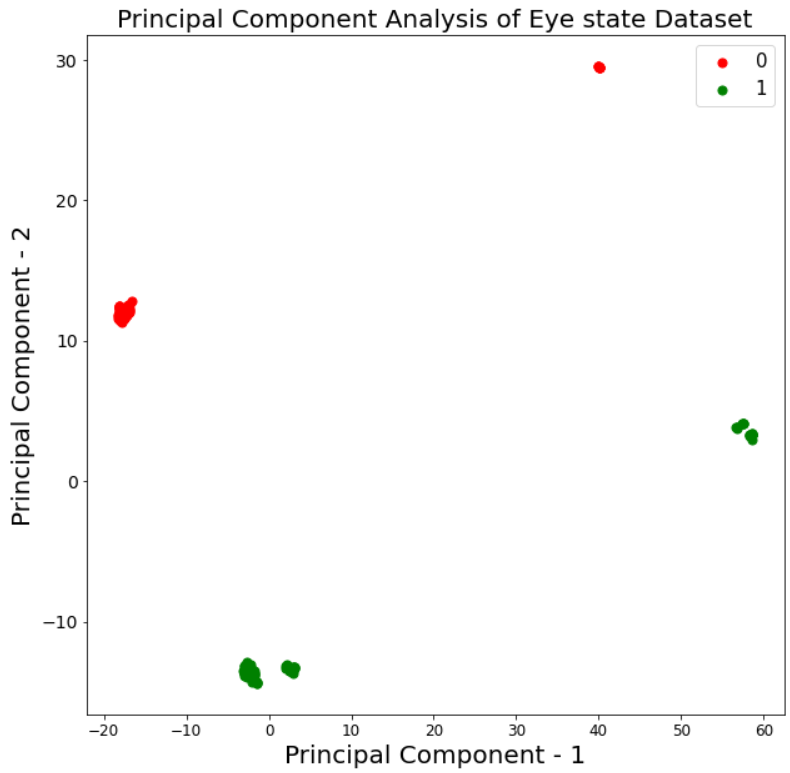


Fig. 7 Class 0 is for eye closed state, and class 1 is for eye opened state.

From Fig. 7 we can easily visualise that there are four clusters out of which 2 are representing class 0 and other 2 representing class1.

4.2 Clustering of the converted data

After the conversion of the high-dimensional EEG data to a low dimensional latent space, the clustering is performed. K-means++ method has been used to perform the initialisation of the cluster centroids. Afterwards, the k-means algorithm has been iteratively used to divide the data in the form of clusters.

4.2.1 K-means++ initialization

The exact algorithm is as follows:

1. Choose one centre uniformly at random among the data points.
2. For each data point *x*, compute D(*x*), the distance between *x* and the nearest centre that has already been chosen.
3. Choose one new data point at random as a new centre, using a weighted probability distribution where a point *x* is chosen with probability proportional to $D(x)^2$.
4. Repeat Steps 2 and 3 until *k* centres have been chosen.
5. Now that the initial centres have been chosen proceed using standard *k*-means clustering.

4.2.2 K-means algorithm

The K-means algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares (see below). This algorithm requires the number of clusters to be specified. It scales well to large number of samples and has been used across a large range of application areas in many different fields.

The k-means algorithm divides a set of N samples X, into K disjoint clusters C, each described by the mean μ_j of the samples in the cluster. The means are commonly called the cluster "centroids"; note that they are not, in general, points from X, although they live in the same space.

The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum-of-squares criterion:

$$\sum_{i=0}^{n} \min_{\mu_j \epsilon\, C} \left(\| x_i - \mu_j \|^2 \right)$$

Inertia can be recognized as a measure of how internally coherent clusters are. It suffers from various drawbacks:

- Inertia makes the assumption that clusters are convex and isotropic, which is not always the case. It responds poorly to elongated clusters, or manifolds with irregular shapes.
- Inertia is not a normalized metric: we just know that lower values are better and zero is optimal. But in very high-dimensional spaces, Euclidean distances tend to become inflated (this is an instance of the so-called "curse of dimensionality"). Running a dimensionality reduction algorithm such as Principal component analysis (PCA) prior to k-means clustering can alleviate this problem and speed up the computations.

K-means is often referred to as Lloyd's algorithm. In basic terms, the algorithm has three steps. The first step chooses the initial centroids, with the most basic method being to choose k samples from the dataset X. After initialization, K-means consists of looping between the two other steps. The first step assigns each sample to its nearest centroid. The second step creates new centroids by taking the mean value of all of the samples assigned to each previous centroid. The difference between the old and the new centroids are computed and the

algorithm repeats these last two steps until this value is less than a threshold. In other words, it repeats until the centroids do not move significantly.

After applying K-means clustering, we tested the data points, and we got **100% accuracy** from our method. This study, not merely proposed a robust and fast approach for classification of EEG data using unsupervised learning but also implemented a way of analysing the high dimensional EEG data in low dimensional space.

5. CONCLUSIONS

5.1 Concluding Remarks

In the current work, the classification for eye state using unsupervised learning methods and EEG is performed. The important conclusions of this study are listed as follows:

1. For efficient processing of information-rich data, dimension reduction is a crucial step. Here, each sample data containing **14336** dimensions is firstly converted into **28** dimensions in latent space and then again converted to 14336 dimensions as output data. This process has been achieved through a trained autoencoder.
2. The latent space was found to be successful in capturing most of the characteristics of the original data.
3. The reduced data is used for data classification.
4. The clustering was initialized using k-means++ algorithm. Subsequently, k-means algorithm has been used (k=4). PCA has been used to validate the dimensional reduction and clustering.
5. The unsupervised learning has been used and validated. The achieved accuracy is **100%**.

5.2 Scope for Future Work

This approach can be used for classifying EEG subjects for a different task like depression, emotions, SSVEP, BCI. Moreover, as we know that the size of EEG data is generally immense, by using the proposed autoencoder, the size of data can be reduced to **96.875%**. In future, we will test our method on other tasks, and we will improve the methods by trying different algorithms like GANs, FCM.

REFERENCES

1. ***Electroencephalography: What are neurons?*: Rishabh Bajpai on Medium.com**
2. ***An Overview of Brain Signals and its applications*: Rishabh Bajpai on Medium.com**
3. *Samadi, MRH and Cooke, N., 2014, September. EEG signal processing for eye tracking. In 2014 22nd European Signal Processing Conference (EUSIPCO) (pp. 2030-2034). IEEE.*
4. *Wang, T., Guan, S.U., Man, K.L. and Ting, T.O., 2014. EEG eye state identification using incremental attribute learning with time-series classification. Mathematical Problems in Engineering, 2014.*
5. *Narejo, S., Pasero, E. and Kulsoom, F., 2016. EEG based eye state classification using deep belief network and stacked autoencoder. International Journal of Electrical and Computer Engineering (IJECE), 6(6), pp.3131-3141.*
6. *Wen, T. and Zhang, Z., 2018. Deep convolution neural network and autoencoders-based unsupervised feature learning of EEG signals. IEEE Access, 6, pp.25399-25410.*
7. *Chang, C.S., Liao, W., Chen, Y.S. and Liou, L.H., 2016. A mathematical theory for clustering in metric spaces. IEEE Transactions on Network Science and Engineering, 3(1), pp.2-16.*
8. *Chittenden, E.W., 1927. On the metrization problem and related problems in the theory of abstract sets. Bulletin of the American Mathematical Society, 33(1), pp.13-34.*
9. *Pradhan, C.K., Rahaman, S., Sheikh, M.A.A., Kole, A. and Maity, T., 2019. EEG Signal Analysis Using Different Clustering Techniques. In Emerging Technologies in Data Mining and Information Security (pp. 99-105). Springer, Singapore.*
10. *Shindler, M., 2008. Approximation algorithms for the metric k-median problem. Written Qualifying Exam Paper, University of California, Los Angeles. Cited on, p.44.*
11. ***EEG_Clean* – NelsonGon on Kaggle**