Automatic Detection of EEG Microstates using Unsupervised Learning

***Abstract*—EEG signal is generally used to detect problems in the electrical activity of the brain that can be used to analyze certain brain disorders. EEG topography represents the dynamical view of how spatial distribution of the electric potential on the scalp changes over time. Different distributions of neuron generations in the brain gives rise to the different configurations of potential maps on the scalp. EEG data is a multi-channel data which makes it's analysis complex. This data can be compressed by finding the microstates and by substituting the topographies in the original EEG data by the microstate labels. This provides the compressed and simplified view of the data. In this paper we are extracting microstates from the EEG signal by using different unsupervised clustering algorithms and analyze their performance to obtain the efficient method for microstate detection.**

I. Introduction

EEG signal is a series of topographies that denotes the electric potential over multi-channel electrode array at a particular time instant. We can get the optimal representation of EEG time series data by obtaining small number of topography maps that will give the optimum representation of whole data set. For this purpose we perform microstate detection in which single topography is made to represent bunch of topographies based on similarity criterion.

EEG microstates are generally identified using clustering algorithms and they may vary depending on the choice of algorithm and parameters used.

Microstates computed by different clustering algorithms cannot be labelled by symbols unequivocally and hence maps obtained from one clustering algorithm cannot be directly compared with the maps obtained from another algorithm. For comparing the performance of different clustering algorithms on EEG data we use Global Explained Variance (GEV) which basically denotes the variance captured by the topographies present in set of microstate maps. Hence it denotes how much information got successfully extracted from the original EEG data. GEV is calculated across the whole sequence of EEG data. It is calculated by using formula

$$GEV = \sum_{i=1}^{n} \frac{(corr(x_i, m_i) \times GFP_i)^2}{\sum_n GFP^2}$$

Here GFP denotes the global field power of the multi-channel EEG signal. It is a single time series signal that denotes the strength of total electric potential obtained from all the electrodes. It is calculated by using the formula

$$GFP = \sum_{i=1}^{n} \frac{\sqrt{(V_i(t) - V_{mean})^2}}{n}$$

Here n denotes the total number of electrodes that are used to take EEG data from the scalp. $V_i(t)$ denotes the electric potential present at electrode i at time instant t and V_{mean} denotes the average of electric potentials from all the electrodes at particular time instance.

A. Microstate Detection

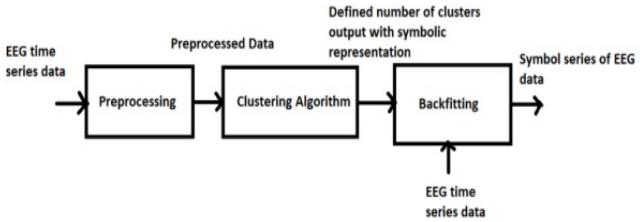

Fig. 1. Microstate Detection

In microstate detection, preprocessed EEG time series data is given to the clustering algorithm along with the desired number of clusters to obtain microstates. This microstates are then represented by certain labels which can be backfitted into the original EEG data to obtain compressed representation of EEG data which provide simplified view of data. Figure 1 shows the block diagram which represents the procedure followed during microstate detection.

Figure 2 gives the detailed overview of steps followed in the process of microstate detection. Here time series EEG data corresponding to thirty-two electrodes is given to the clustering algorithm along with input number of clusters N=4. Clustering algorithm outputs the four microstates that are then symbolically represented by letters A,B,C and D. These symbols are then backfitted into the original EEG data as shown in figure.

The main idea of this project is to explore the effectiveness of unsupervised learning for automatically detecting the microstates in EEG signal.

II. Literature Review

In EEG multi-channel time series data large number of topographic maps are present, but in the studies done by researchers it has been observed that usually $> 70\%$ of the topographic variance can be represented by using fewer number of maps and hence entire EEG signal can be represented by using small number of topographic maps through microstate detection.

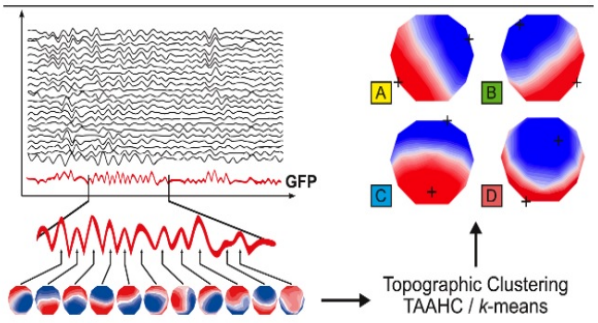

Figure 1.2: Clustering

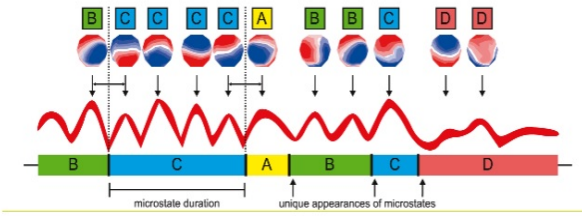

Fig. 2. Microstate Detection [3]

EEG microstate detection aims to reduce the multi-cha topographic EEG time series data to the small set of repre tative EEG topographic maps. Techniques used for micro detection has evolved a lot over time. In previous t adaptive segmentation [3] method was used for micro detection. In this method, current GFP peak is compared the previous one, if the difference between these two is gr than certain threshold value then the current GFP peak wi considered as a start of new microstate. Due to advancen in the fields of data science and machine learning large nu of clustering algorithms are present which can be used in the process of microstate detection. Clustering algorithms gives the advantage of differentiating the input data into different classes and hence allows to use one representative for each class.

Different clustering methods such as K-means, principal component analysis(PCA) [4] and independent component analysis(ICA) [4] are used for this purpose. Each method have advantages and disadvantages associated with it. The main disadvantage of PCA algorithm for clustering in microstate detection is it requires mandatory orthogonality of the clusters which can produce microstates that may lack biological relevance. Similarly in ICA, there is a condition of independent clusters which can interfere with the biological information. Microstates obtained from a clustering algorithm is highly dependent on the way used in an algorithm to assign certain datapoint to a particular cluster. Hence microstates obtained from different algorithms are different.

From the previous research in the field of microstate detection it has been observed that as we increase the number of clusters(N), GEV may get improved but increase in the value of GEV will not be significant. If we take number of clusters as 4, we can get upto 60% of the information from the original data. Hence while studying the effect of different clustering algorithms on the process of microstate detection we consider number of clusters as 4.

EEGLAB [11], FieldTrip [12] and MNE-Python are few of the technologies that are used for EEG analysis. They provide the functionalities such as data preprocessing, classification and statistical analysis. In our algorithmic implementations we have used MNE library [8] which gives different prebuilt functions for analyzing and visualizing the data.

III. Dataset Description

DEAP dataset [7] contains preprocessed files of EEG data of 32 subjects. This files are generated by the Actiview recording software where for each subject 48 recording channels (32 EEG channels) at 512Hz were used. This data is then down-sampled to 2128Hz, preprocessed and segmented which makes it more suitable for classification and regression technique.

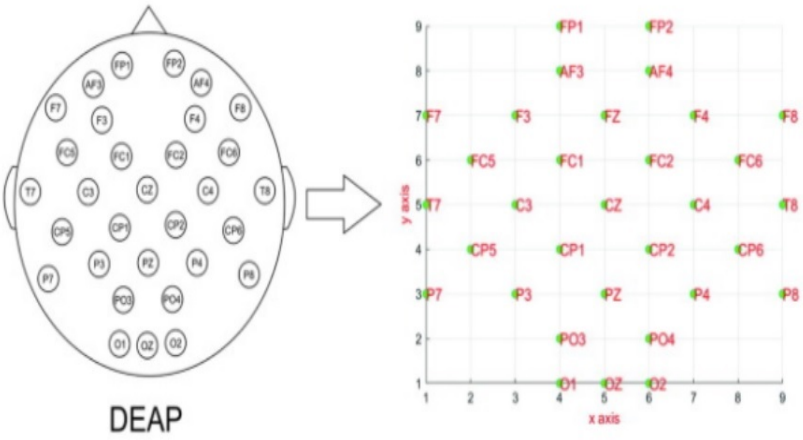

Fig. 3. Electrode positions on scalp while recording EEG data for DEAP dataset

Figure 3 shows the position of 32 electrodes on the scalp of the subject while recording the EEG signal. This positions are used while plotting the microstate maps. Each microstate map in the further section shows the electric potential on the scalp at all electrodes at a particular time instant.

In this project all experimentation is performed on sample 1 of DEAP dataset.

IV. Experiments and Results

In microstate analysis, topographies present in the EEG data are given to the clustering algorithm to obtain desired number of clusters. Maps corresponding to these clusters are represented symbolically and further used for backfitting into the original EEG data. In our project, we have used K-means, Self Organizing Feature Map (SOFM), Affinity propagation and Atomize and Agglomerate Hierarchical Clustering (AAHC) unsupervised clustering methods to analyze the performance of different clustering algorithms and to obtain the efficient method that can represent the maximum information from the input data in minimum number of maps. Experimentation done

in this section is not restricted to number of clusters N=4 and can be applied to any number of clusters.

Microstates computed by different clustering algorithms cannot be labelled by symbols unequivocally and hence maps obtained from one clustering algorithm cannot be directly compared with the maps obtained from another algorithm. For comparing the performance of different clustering algorithms on EEG data we use Global Explained Variance (GEV) which basically denotes the variance captured by the topographies present in set of microstate maps. Hence it denotes how much information got successfully extracted from the original EEG data. GEV is calculated across the whole sequence of EEG data. It is calculated by using formula

$$GEV = \sum_{i=1}^{n} \frac{(corr(x_i, m_i) \times GFP_i)^2}{\sum_n GFP^2}$$

Here GFP denotes the global field power of the multi-channel EEG signal. It is a single time series signal that denotes the strength of total electric potential obtained from all the electrodes. It is calculated by using the formula

$$GFP = \sum_{i=1}^{n} \frac{\sqrt{(V_i(t) - V_{mean})^2}}{n}$$

Here n denotes the total number of electrodes that are used to take EEG data from the scalp. $V_i(t)$ denotes the electric potential present at electrode i at time instant t and V_{mean} denotes the average of electric potentials from all the electrodes at particular time instance.

First all the topographies present in the original EEG data are given to the clustering algorithms. Next EEG data corresponding to GFP peaks is given as input to the clustering algorithms which improved the performance of clustering models by increasing the obtained GEV. All the results shown in this section are for sample 1 of DEAP dataset.

A. K-means

K-means is a well-known most basic method used for clustering. It is an iterative algorithm which partitions the input data into predefined k clusters. Here we have used EEG data corresponding to the GFP peaks as a input to this algorithm. In this algorithm, initially k centroids are initialized by some random input data points. Then we compute sum of the squared distance between a input datapoint and a centroid for all the k centroids. The input datapoint is then assigned to the closest centroid. Then for all the clusters, centroid is calculated by taking average of input data points that belong to this cluster. In our experimentation we varied k from three to ten to analyze the effect of value of k on the obtained Global Explained Variance (GEV). Table I denotes how GEV varies as number of clusters (N) are increased from three to ten. Here it can be observed that as number of microstates increases GEV also increases, but the increase in GEV is not much significant for higher values of N and hence does not give justice to the amount of data it is combating from reduction.

We have also calculated Shannon entropy and microstate distribution this algorithm. Shannon entropy specifies the amount of information that gets stored in a variable. It is calculated by using the formula

$$H(x) = -\sum_{x=1}^{N} p(x) log\, p(x)$$

Microstate distribution shows how calculated microstates are distributed across the whole data. It shows the probability of occurrence of specific microstate in the whole data.

Shannon entropy for K-means model when EEG data corresponding to GFP peaks is applied as input is 1.3683. Microstate distribution for this model is p(A)= 0.2092, p(B)= 0.3074, p(C)= 0.1978 and p(D)= 0.2856. This shows that microstates are almost equally distributed across the whole data.

Main drawback of K-means is that the final clusters obtained depends on initialization of centroid. Hence in our experimentation we ran K-means on ten different random initializations and consider the output clusters for which Global Explained Variance (GEV) is maximum.

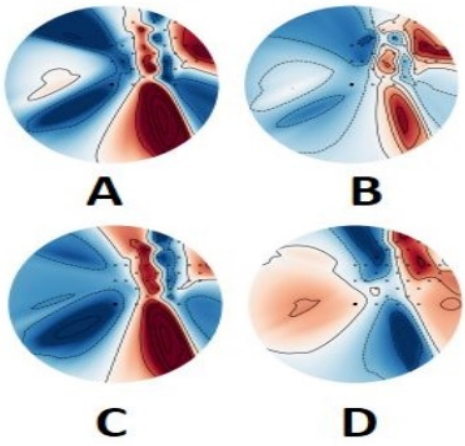

Fig. 4. Microstate maps using K-means clustering

Figure 4 shows the microstate maps obtained using K-means clustering for N=4.

Number of microstates	GEV obtained using K-means
3	0.6002
4	0.6254
5	0.6458
6	0.6621
7	0.6736
8	0.6823
9	0.6903
10	0.6972

TABLE I
GEV OBTAINED USING K-MEANS CLUSTERING FOR DIFFERENT VALUES OF N

B. Self Organizing Feature Map (SOFM)

SOFM is basically a neural network which is trained using unsupervised learning to produce the low-dimensional representation of the input data by using competitive learning. Like most of the neural networks, SOFM operates in two modes

training and mapping. In training mode, model is trained on given input data while in mapping mode new input vector get classified.

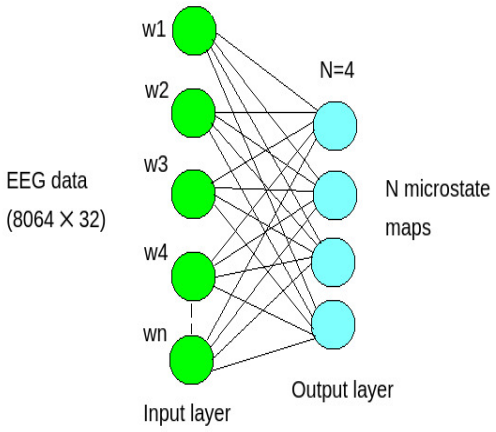

Fig. 5. SOFM neural network

Figure 5 shows the basic structure of SOFM neural network. Here number of nodes in input layer of neural network are equal to the number of samples in the input data while number of nodes in the output layer are equal to the inputted number of clusters N. Each node has some weight associated with it. Initially this weights are initialized with some random value and in the training phase of the algorithm these weights gets adjusted by the input vectors to be able to classify the data.

Algorithm 1 gives the basic overview of the steps followed in SOFM clustering.

Algorithm 1 Self Organizing Feature Map (SOFM) algorithm

Input: (a) EEG data (b) α: learning rate (c) $epochs$: Number of epochs (d) N: Number of output clusters

1: Calculate GFP peaks.
2: Calculate the input data matrix for SOFM by taking the data at the the GFP peaks from EEG data.
3: Construct the 2-dimensional weight matrix with size number of features by N. Initialize the weight matrix by random values.
4: Take any random input from the input data for SOFM. Calculate the square of euclidean distance for each node by using formula

$$D_j = \sum_i (w_{ij} - x_i)^2$$

Node for which value of D is lower will be considered as a winning node.
5: Update the weights of the winning node by using formula

$$w_{ij(new)} = w_{ij(old)} + \alpha(x_i - w_{ij(old)})$$

6: Repeat the steps 4 & 5 for all inputs for given number of epochs.

In our experimentation we evaluated SOFM model for both complete EEG data and EEG data corresponding to GFP peaks. Here we took learning rate as constant $\alpha = 0.24$ and neighbourhood function as gaussian. Model is trained for 200 number of epochs. First we applied the whole EEG data to the SOFM model. This yielded a GEV of 0.5056 for the input number of microstates N=4. Next, we applied EEG data corresponding to GFP peaks to the SOFM model, this improved the GEV to 0.5305 for the same value of N. Figure 6 shows the microstate maps obtained using SOFM clustering for N=4 for the input EEG data corresponding to GFP peaks. Table II denotes how GEV varies as number of clusters are increased from three to ten. Here we can observe that GEV increases as number of microstates (N) increases but for higher values of N corresponding amount of improvement in GEV is insignificant.

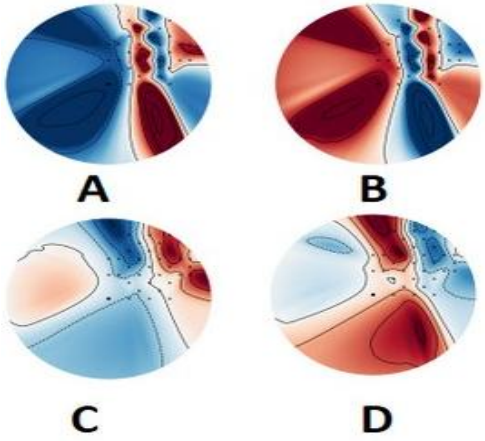

Fig. 6. Microstate maps using SOFM clustering for the input of EEG data corresponding to GFP peaks only

Number of microstates	GEV obtained using SOFM (EEG data corresponding to GFP peaks only)
3	0.4794
4	0.5305
5	0.5631
6	0.5710
7	0.0.5817
8	0.5871
9	0.5936
10	0.5986

TABLE II
GEV OBTAINED USING SOFM CLUSTERING FOR DIFFERENT VALUES OF N

For further analysis, we varied the learning rate dynamically instead of using the constant rate to improve the performance of the model. Here we changed learning rate of the training model from constant function to different decaying functions which resulted in the significant improvement of GEV. Learning rate is a hyper-parameter that is used to control the rate at which an algorithm updates the parameter estimates i.e. weight matrix.Here we used four different decaying functions that are linear decaying function, power series decaying function,

circular decaying function and decaying function with constant delta decay rate.

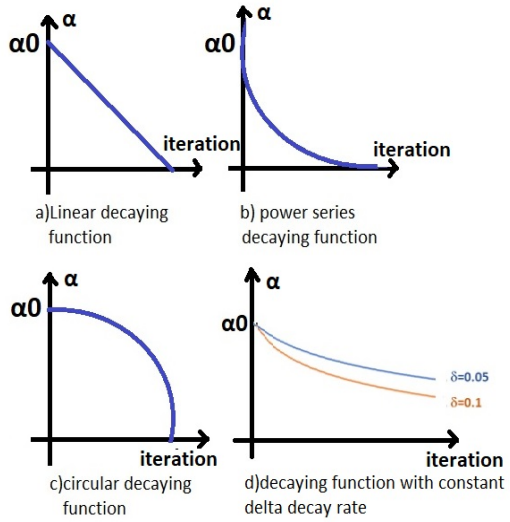

Fig. 7. Different decaying functions used for learning rate

Figure 7 gives the graphical representation of different decaying functions used in learning rate. Here values on x-axis denotes the iteration number while values along y-axis denotes learning rate.

1) Learning rate as linear decaying function: SOFM model is trained on EEG data corresponding to GFP peaks with learning rate as linear rate decaying function with initial learning rate $\alpha_0 = 0.24$. This yielded GEV of 0.6155 for number of microstates N=4, number of epochs=200. Figure 8 shows the microstate maps corresponding to this configuration. Below equation shows the formula used to calculate the learning rate, here t denotes the iteration number while T denotes the total number of iterations.

$$\text{Learning rate } \alpha = \alpha_0(1 - \frac{t}{T})$$

2) Learning rate as power series decaying function: Next, we used power series decaying function for learning rate while training SOFM model on EEG data corresponding to GFP peaks. This yielded a GEV of 0.6068 for number of clusters N=4, number of epochs=200 and $\alpha_0 = 0.24$. Figure 9 shows the microstate maps corresponding to this configuration. Below equation shows the formula used to calculate the learning rate, here t denotes the iteration number while T denotes the total number of iterations.

$$\text{Learning rate } \alpha = \alpha_0 e^{\frac{-t}{T}}$$

3) Learning rate as circular decaying function: Further, we used circular decaying function for learning rate while training SOFM model on EEG data corresponding to GFP peaks. This yielded a GEV of 0.6071 for number of clusters N=4, number of epochs=200 and $\alpha_0 = 0.24$. Figure 10 shows the microstate maps corresponding to this configuration.

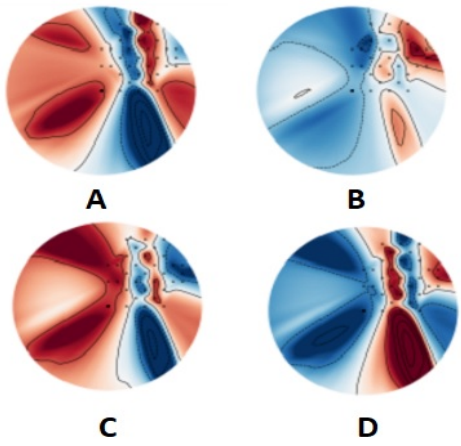

Fig. 8. Microstate maps using SOFM clustering for learning rate as linear decaying function

Below equation shows the formula used to calculate the learning rate, here t denotes the iteration number while T denotes the total number of iterations.

$$\text{Learning rate } \alpha = \sqrt{\alpha_0^2 - \frac{t}{T}^2}$$

Table III shows the GEV obtained in SOFM model for linear, power series and circular decaying functions as the value of number of clusters N is varied from three to ten. Here it can be observed that increase in the value of N does not assure increase in the value of GEV for higher values of N. All three functions shows significant improvement in the value of GEV as compared to constant learning rate function.

4) Learning rate as decaying function with constant delta decay rate: Here we used decaying function with constant delta decay rate in learning rate for training the sofm model on EEG data corresponding to GFP peaks. For $\delta = 0.12$, N=4, $\alpha_0 = 0.24$, number of epochs=200 this model yielded a

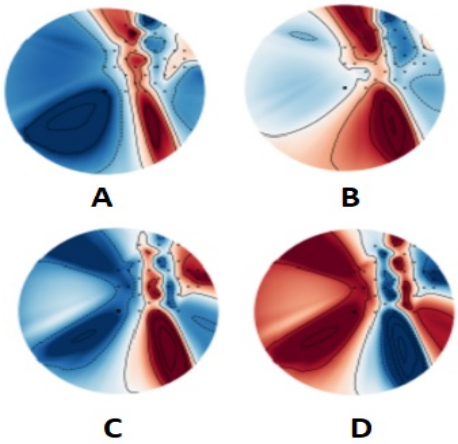

Fig. 9. Microstate maps using SOFM clustering for learning rate as power series decaying function

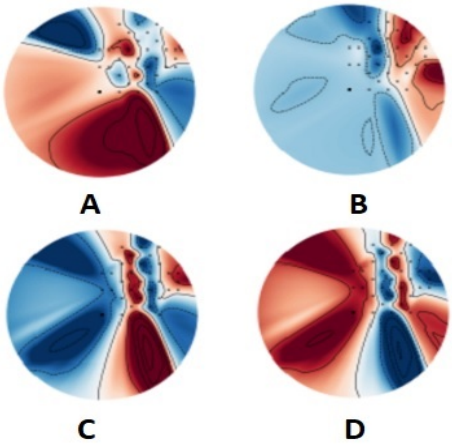

Fig. 10. Microstate maps using SOFM clustering for learning rate as circular decaying function

highest GEV 0.6170. Table IV shows the values of obtained GEV for different values of delta decay rate.

$$\text{Learning rate } \alpha = \frac{\alpha_0}{1+\delta \times epoch\ number}$$

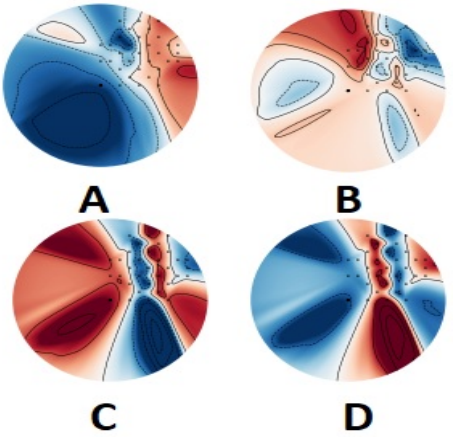

Fig. 11. Microstate maps using SOFM clustering for learning rate as constant delta decaying function

C. Atomize and Agglomerate Hierarchical Clustering (AAHC)

Atomize and agglomerate hierarchical clustering(AAHC) is a hierarchical clustering in which smaller clusters are merged into the larger clusters based on certain criteria. It is a deterministic clustering algorithm in which when certain input is applied, same steps will be followed which finally results in the same microstate maps. In AAHC, initially all input datapoints are considered as individual clusters and in each step of algorithm two most similar clusters are merged together. This step is repeated until it reaches the desired number of clusters. It uses a bottom-up approach where initially large number of clusters are present and during each iteration two clusters get merged.

GEV obtained using SOFM model			
Number of clusters	**Linear decaying function**	**Power series decaying function**	**Circular decaying function**
3	0.5945	0.5867	0.5911
4	0.6155	0.6068	0.6071
5	0.6155	0.6222	0.6034
6	0.6173	0.6193	0.6170
7	0.6121	0.6198	0.6115
8	0.6100	0.6143	0.6178
9	0.6102	0.6120	0.6119
10	0.6067	0.6243	0.6111

TABLE III
GEV OBTAINED USING SOFM FOR LEARNING RATE AS DIFFERENT DECAYING FUNCTIONS

delta	GEV obtained using SOFM
0.08	0.6045
0.11	0.6076
0.12	0.6170
0.13	0.6167
0.15	0.6137

TABLE IV
GEV OBTAINED USING SOFM FOR LEARNING RATE AS DECAYING FUNCTION

Algorithm 2 gives the basic overview of the steps followed in AAHC clustering.

Algorithm 2 Atomize and Agglomerate Hierarchical Clustering (AAHC) algorithm

Input: (a) EEG data (b) N: Number of output clusters

1: Calculate GFP peaks.
2: Calculate the input data matrix for AAHC by taking the data at the the GFP peaks from EEG data.
3: Initialize the clusters matrix by input data for AAHC.
4: Check if obtained number of clusters are equal to N, if yes then break.
5: Calculate correlation between datapoints and maps.
6: Find a datapoint for which correlation is maximum but GEV is minimum.
7: Find cluster with which this datapoint have maximum correlation and add it to that cluster.
8: Update the map corresponding to this cluster by using eigen vector method.
9: Repeat the steps three to eight until the desired number of clusters are obtained.

For number of clusters N=4 AAHC gives the GEV of 0.6165 and hence maps obtained from this model are able to represent 61% of the information from the original EEG data. Table V represents how GEV varies as the number of microstates are changed from three to ten. Figure 13 shows the microstate maps obtained using AAHC clustering for N=4. Here it can be observed that as the number of microstates(N) increases obtained GEV also increases. The main disadvantage of AAHC algorithm is it's relatively high computation time.

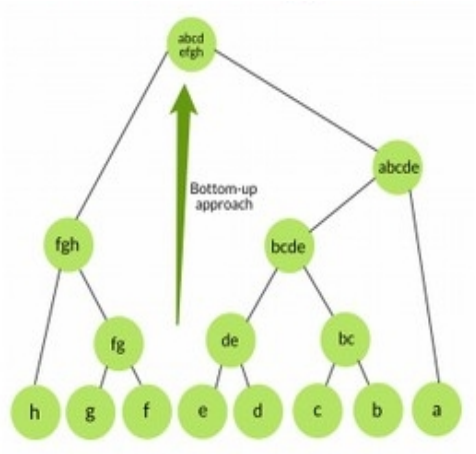

Fig. 12. AAHC clustering [13]

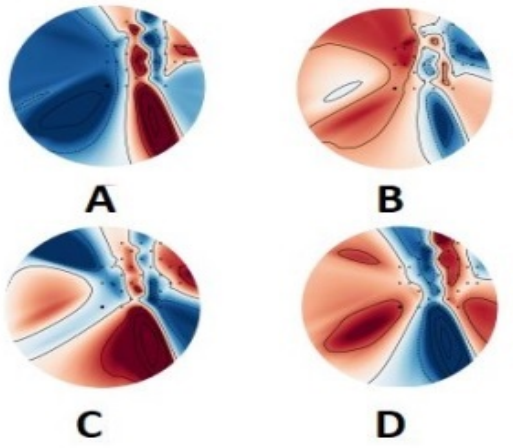

Fig. 13. Microstate maps using AAHC clustering

D. Affinity Propagation

Affinity propagation [10] is a machine learning algorithm that is used for clustering. The main advantage of affinity propagation clustering is that it does not require estimation of number of clusters before running the algorithm. It finds exemplars which acts as representative of the clusters from the input data set. In clustering, it takes similarities between data points as input and identifies exemplars based on certain criteria.

In process of clustering, four matrices similarity matrix, responsibility matrix, availability matrix and criterion matrix are calculated. These all are the two dimensional matrices of size $n \times n$ where n denotes the number of samples in the input data set.

1) Calculation of Similarity matrix: In similarity matrix x_{ij} denotes how similar input i is, with input j. Here non-diagonal values are calculated by taking negation of the sum of the squares of the differences between features of the two inputs. This can be mathematically represented as

Number of microstates	GEV obtained using AAHC
3	0.5919
4	0.6165
5	0.6396
6	0.6596
7	0.6687
8	0.6736
9	0.6824
10	0.6891

TABLE V
GEV OBTAINED USING AAHC CLUSTERING FOR DIFFERENT VALUES OF N

$$S(i,j) = -\|x_i - x_j\|^2$$

Diagonal values are filled with the lowest value among all the cells.

2) Calculation of Responsibility matrix: Values in responsibility matrix denote how responsible one object is for another. r_{ij} is calculated by similarity between input i and input j, s_ij minus maximum of similarities of input i. This can be mathematically represented as

$$r(i,k) = s(i,k) - \max_{k' \neq k} \{a(i,k') + s(i,k')\}$$

Here $r(i,k)$ denotes how well suited element k is to be an exemplar of element i, taking into account the nearest contender k' to be an exemplar of i.

3) Calculation of Availability matrix: Values in availability matrix denotes how available one object is to be an exemplar of another. They are calculated by using formula

$$a(i,k) = \min\left(0, r(k,k) + \textstyle\sum_{i' \notin \{i,k\}} max(0, r(i',k))\right)$$

for $i \neq k$

It denotes how appropriate is it for i to choose k as its exemplar, taking into account the support from other elements that k should an exemplar.

4) Calculation of Criterion matrix: Criterion matrix is calculated by adding *Responsibility matrix* and *Availability matrix*.

$$c(i,k) = r(i,k) + a(i,k)$$

It denotes that an element i will be assigned to an exemplar k which is not only highly responsible but also highly available to i.

The element with the highest criterion value in each row is considered to be an exemplar. Elements corresponding to the rows which share the same exemplar are clustered together.

In our experimentation we trained affinity propagation model on two variations of data. First we considered EEG data corresponding to only gfp peaks. This gave 86 clusters yielding GEV of 0.6994. Next we trained model on all the input samples in EEG data. This gave 305 clusters yielding GEV of 0.7718.

V. Conclusion

In this paper we analyze the performance of K-means, Self organizing feature map (SOFM), Atomize and agglomerate clustering (AAHC) and Affinity propagation clustering algorithms in the process of microstate detection. For analyzing the amount of information that got extracted from the original EEG data through clustering GEV parameter is used. From experimentation and analysis presented in paper we can observe that for number of clusters N=4 almost 62% of original information can be restored from symbolically compressed data that used K-means, SOFM or AAHC clustering. We also tried to improve the performance of SOFM model by using different functions for dynamically varying the learning rate. We also tried to analyze the performance of Affinity propagation clustering which gives the advantage of not requiring the estimation of number of clusters before running the algorithm.

References

[1] Pascual-Marqui RD, Michel CM, Lehmann D., "Segmentation of brain electrical activity into microstates: model estimation and validation", Biomedical Engineering, IEEE Transactions on. 1995; 42:658.

[2] Michel, C. M., Koenig, T., Brandeis, D., Wackermann, J., Gianotti, L. R. (Edition 2009),Electrical neuroimaging, Cambridge University Press.

[3] Khanna, A., Pascual-Leone, A., Michel, C. M., Farzan, F., "Microstates in resting-state EEG: current status and future directions", Neuroscience Biobehavioral Reviews, 49, 105-113.

[4] VonWegner, Frederic, Paul Knaut, and Helmut Laufs, "EEG microstate sequences from different clustering algorithms are information-theoretically invariant", Frontiers in computational neuroscience 12 (2018): 70.

[5] Arnaud Delorme, Scott Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis", 2004 Mar 15;134(1):9-21.

[6] Robert Oostenveld, Pascal Fries, Eric Maris, Jan-Mathijs Schoelen, "FieldTrip: Open Source Software for Advanced Analysis of MEG, EEG, and Invasive Electrophysiological Data", Computational Intelligence and Neuroscience, vol. 2011, Article ID 156869, 9 pages, 2011. https://doi.org/10.1155/2011/156869.

[7] C. Muehl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, I. Patras, "DEAP: A Database for Emotion Analysis using Physiological Signals", S. Koelstra, EEE Transactions on Affective Computing, vol. 3, no. 1, pp. 18-31, 2012

[8] https://mne.tools/stable/overview/

[9] https://sapienlabs.org/finding-eeg-microstate-classes-with-clustering

[10] https://towardsdatascience.com/math-and-intuition-behind-affinity-propagation-4ec5feae5b23

[11] https://eeglab.org/

[12] https://www.fieldtriptoolbox.org/

[13] https://www.geeksforgeeks.org/ml-hierarchical-clustering-agglomerative-and-divisive-clustering/

[14] J. Kaur and A. Kaur, "A review on analysis of EEG signals," 2015 International Conference on Advances in Computer Engineering and Applications, 2015, pp. 957-960, doi: 10.1109/ICACEA.2015.7164844.

[15] A. Nawrocka and A. Kot, "Methods for EEG signal analysis," 2011 12th International Carpathian Control Conference (ICCC), 2011, pp. 266-269, doi: 10.1109/CarpathianCC.2011.5945861.

[16] J. Garza, Y. Li, Y. Chang and H. Lin, "A Real Time EEG Analysis System," 2016 IEEE First International Conference on Data Science in Cyberspace (DSC), 2016, pp. 550-555, doi: 10.1109/DSC.2016.29.

[17] S. Dhivya and A. Nithya, "A Review on Machine Learning Algorithm for EEG Signal Analysis," 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2018, pp. 54-57, doi: 10.1109/ICECA.2018.8474801.