A Generalized Architecture for EEG Data Analysis

Ayush Jain (SN: 260673423), Neeth Kunnath (SN: 260665112), Harsh Satija (SN: 260672703)

***Abstract*—Electroencephalogram (EEG) is a test that detects electrical activity in the brain. Their applications, if decoded are numerous. Currently EEG data is analyzed to aid in the diagnosis of various neurological conditions like epilepsy, sleep disorder, coma, encephalopathies etc. Statistical EEG analysis is also finding applicability in cognitive science, cognitive psychology, and psychophysiological research. In the past, scientists have attempted to use EEG data in various prediction task such as seizure prediction in epileptic patients. There has also been numerous works done in 'decoding' EEG signals. This can open new avenues in the area of human-computer interaction (HCI). The main challenge when deploying machine learning techniques on EEG data is that almost always, it requires a highly specific and tuned architecture that is very specific to the task at hand. As a result, there are many specialized architectures for very specific tasks, but there is no common architecture to handle all EEG data irrespective of the task. In this project, we set out to design a generalized, task-insensitive architecture for EEG data. We present our experiments and results in this report.**

I. Introduction

EEG is electrophysiological monitoring of brain where electrical activity at different locations along scalp is recorded with electrodes. EEG captures voltage fluctuations resulting from ionic current within the neurons. EEG analysis finds applicability in epilepsy analysis, sleep disorder diagnosis, coma, and encephalopathies. Derivatives of the EEG include evoked potentials and event-related potentials are usually used to study EEG responses that are time-locked to more complex processing of stimuli; this technique is used in cognitive science, cognitive psychology, and psychophysiological research.

Despite limited spatial resolution, EEG data is a valuable tool for research an diagnosis, especially when millisecond-range temporal resolution is required. The statistical analysis of EEG data is as yet not fully standardised, and the strategies employed vary considerably depending on the hypothesis and purpose underlying the experiment. The most frequent approach is to analyse variance of signal amplitudes for different electrodes, sensors or dipoles are considered as dependent variables, and extracted for each subject for further analysis. In case of lack of a specific hypothesis, exploratory strategies are used where separate statistical tests are computed electrode by electrode, or dipole by dipole. Most of the research targeted towards automated statistical EEG analysis depend on domain-specific and task-specific knowledge base.

Our objective for this project is to use the recent advances in deep learning techniques to propose a general framework for statistical EEG data set, which doesn't require any hand crafted nor domain specific features for EEG classification. We plan to apply our framework on various EEG data sets to evaluate its performance against existing models.

II. Related Work and Data sets

EEG data sets exist in various configuration. The data sets can vary on the sampling rate of device used, number of channels available, and so on. These variations are not necessarily limited between data sets, the data within a data set could be collected using different equipment, as a result, there could be variation within the data set itself. To ensure that our model addresses, this diversity, we decided to use multiple data sets while building our model and to test its performance on all of them. In this section we describe the data sets we used and current state-of-the-art work on the model.

A. Epilepsy Seizure Detection

The dataset is taken from Clinique of Epileptology of Bonn University [2]. In this data set single channel EEGs are noted from people having different brain electrical potential components at a sampling rate of 173.61 Hz for 23.6 seconds. There are three kinds of subjects: healthy people, epileptic people during seizure free interval(interictal) and epileptic people during seizure. The data from these subjects is further split into five classes, where each class has 100 single channel EEG recordings, where each observation has 4096 samples. The classes are:

1) Z: Data obtained from healthy people with eyes closed.
2) O: Data recorded from healthy people with eyes open.
3) N: Data taken from the hippocampal pattern of the brain from interictal people.
4) F: Data from Epileptogenic sections of the brain that represent the focal intellectual activity from interictal people.
5) S: Recordings obtained from an epileptic subject during seizure interval.

A lot of past work has been done on this data set where researchers have analyzed non-linear features, entropy-based features, wavelet-based features and time frequency-based features of EEG signals. Authors of [3] evaluated various de-noising techniques like Short Term Fourier Transform (STFT), Wavelet Transform (WT), and Recursive Least Square (RLS). In [4] authors obtained Power Spectral Density (PSD) bands and applied supervised learning algorithms over that. In the work [5] authors used variance based techniques based on entropies and time-frequency distributions for seizure detection. The authors of [6] demonstrated the EEG signal processing using STFT, Independent Component Analysis (ICA) and neural networks. They studied various prediction models used for this task. A comprehensive list of work done on this data set is available at the data set download page [2].

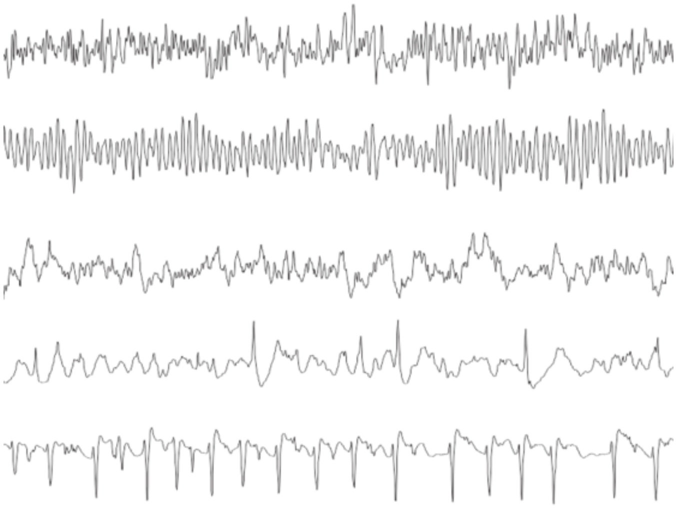

Fig. 1. EEG waves for epilepsy data set. From top to bottom classes Z, O, N, F, S respectively.

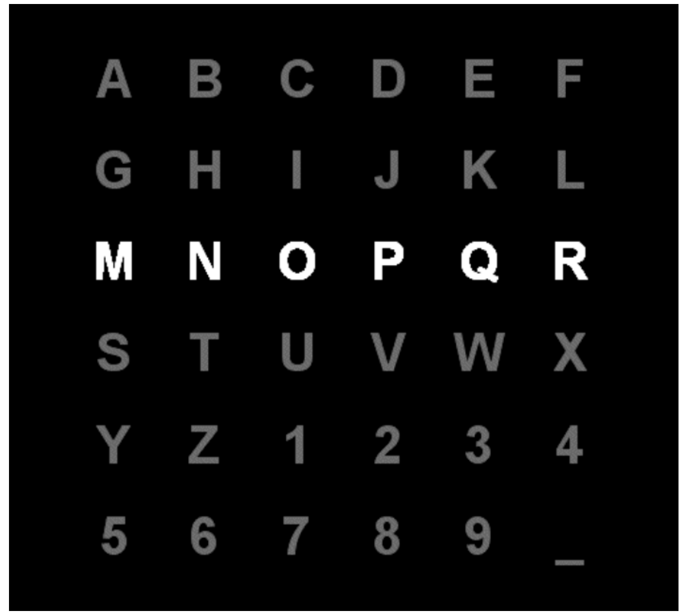

Fig. 2. Example of a 6 x 6 user display for a P300 Speller.

B. P300 Speller

P300 speller, based on Oddball paradigm stating- rare expected stimuli produce a positive deflection in the EEG after about 300 ms, has been introduced by Farwell and Donchin [7]. A subject is presented a 6 x 6 character matrix as illustrated in Figure 2 . For a character detection, each of the 12 rows and columns of the matrix is intensified according to a random sequence and P300 evoked potentials are recorded in response to intensification of the row and column containing the character a subject is focusing on. To make this procedure more reliable, sequence of intensification is repeated 15 times for each character to spell.

This data set is picked from BCI Competition III and is available on the competition web page [8]. It comprises 64-channel EEG signals digitized at 240 Hz. A more detailed description of the data set can be found at [9]. The classification problem we address is the following : given the 64-channel signals collected after the intensification of a row or column, named a post-stimulus signal, we want to predict if such signal contains or not a P300 ERP. After this identification, characters can easily be detected from row and column information.

Since this data set is taken from a BCI competition [8], significant work has been done on this data set as well. A lot of specialized algorithms have reached around 90% accuracy [10].

C. BCI-125

Brain-Computer Interface-125[13] (BCI-125) contains EEG data for motor-imagery tasks. The data was collected with two devices-g.tec (g.USBamp) and Neuroscan (SynAmps RT). g.tec sampled data at a rate of 256Hz and Neuroscan at a rate of 250Hz. Further details is provided in Table II-C. The number of electrodes used also varied within the data set. Hence the data was recorded in configurations of 5, 6, and 14 channels.

TABLE I
RECORDING DEVICES SPECIFICATION FOR BCI-125

Device	Sampling Frequency	Measuring Unit	Bandpass Filter
g.tec	256Hz	V	2-30 Hz
Neuroscan	250Hz	μV	0.1-100 Hz

The cue-based BCI paradigm consisted of two/three motor-imagery tasks-imagining movement of left hand (LH), right hand (RH) and both feet (F). Subjects were presented a cue (an arrow pointing either to left, right or bottom) corresponding to the class LH, RH, and F respectively. This prompted them to perform the desired imagery task.
For our prediction task, we used only the three-class prediction task due to time constraints. Further, the EEG recordings were of different time duration ranging from 3 to 5 seconds. In

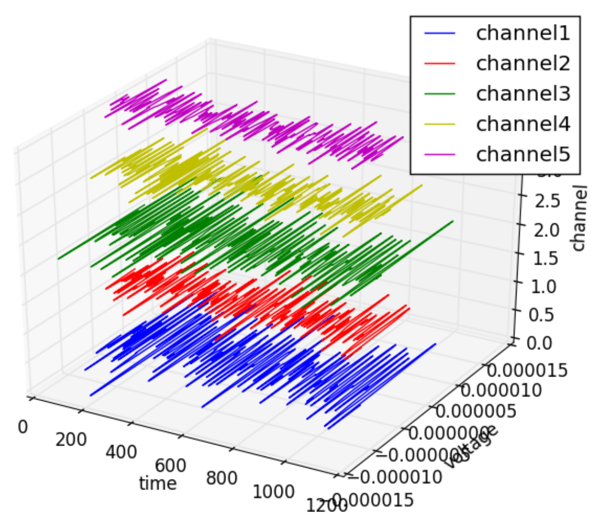

Fig. 3. Sample EEG signal for BCI Data set.

order to have a uniform input dimension, we pad the end of the signals with zeros till they have length 1280 (5×256). This also solves the issue of have different number of data points due to 2 different sampling rates.

III. Data Pre-Processing

We try to restrict the pre-processing to be as non-specific as possible.

A. P300 Speller data set

P300 evoked potentials appear about 300ms after stimulus, hence signals 0 to 667 ms posterior to beginning of intensification is large enough to capture signal time sequence for efficient classification. For each channel, data samples from this window of 160 signals = 240 x 0.667 is extracted.

Signals in this data set have already been bandpass-filtered from 0.1 - 60 Hz so there is no requirement of any addition band filtering.

Data set is made of 85 characters spelling which correspond to 15300 = 12 x 15 x 85 post-stimulus labeled signals (each of them collected over 64 channels). Since only 2 out of 12 intensifications correspond to desired character's row or column, there is a large class imbalance(82.33% vs. 16.67%). This imbalance is compensated for with six fold repetition of minority class. Class imbalance correction lead to 28050 labeled signals.

P.W. Mirowski, Yann LeCun en al. [11] found cross-channel correlation leads to better classification accuracy. Since each channel has a unique base signal, signals are normalized to zero mean and unit variance.

After this pre-processing stage, a post-stimulus signal has been transformed into a vector of dimension 28050 x 160 x 64, where 28050 are different instances of intensifications, 160 corresponds to time sequence for each instance, and for each time unit we have data from 64 different channels.

B. Epilepsy Seizure Detection and BCI-125 data sets

We use two pre-processing techniques-Short Time Fourier Transform (STFT) and Statistical feature extraction.

STFT: Signals in time domain are rather hard to denoise. Fourier Transform (FT) transforms the signals from time domain to frequency domain. Since noise has high frequency, in the frequency domain, denoising is as simple as applying a low-pass filter.
The FT of a signal can be calculated rather easily using Eq.(1).

$$\hat{F}(k) = \sum_{t=0}^{\frac{N}{2}} F(x) e^{-i\frac{2\pi}{N}kt} \tag{1}$$

where, $F(x)$ is the input signal in time domain at time, t, and N is the length of the signal.
Similarly, the inverse FT can be calculated as show in Eq.(2)

$$F(x) = \sum_{k=0}^{\frac{N}{2}} \hat{F}(k) e^{-i\frac{2\pi}{N}kt} \tag{2}$$

Note that here, N is the maximum frequency in $\hat{F}(k)$.
We use a variation on FT, STFT. These can be thought of as a FT applied with a moving window. In our model, we assume the window size to be 4 sampled points. The window is moved by 2 sampled points every time. In every window, we apply a Fast Fourier Transform (FFT). The window is then shifted by two sampled points and this process is repeated till the entire signal has been traversed. The resultant signal was then denoised by clipping off the frequencies about a threshold. The denoised signal is then converted back to time domain using an inverse transform (ISTFT). It works similar to the way STFT does by applying transforms on moving windows. The difference is that an ISTFT apply the inverse FFT rather than an FFT. We implemented our on STFT and ISTFT to build our denoising module. We used Numpy's[1] FFT and inverse FFT modules within our STFT and ISTFT functions.

Statistical Features Extraction: We extracted a few additional statistical features from the denoised signal to enhance the learnability. The features we extracted were maximum voltage, minimum voltage, and mean and standard deviation of signal voltages. These were again calculated on a moving window.

IV. Methodology

A. Convolutional Neural Network

1) Introduction: Convolutional Neural Networks (CNNs) [15], [16] a type of architecture of artificial neural networks which are designed to make use of properties of overlapping regions in visual field. In CNN we use these properties of images to generate feature maps from local connections over the input layer using convolutions.

A typical CNN has the following architecture :

- INPUT will hold the raw pixel values of the image, in this case an image of 48 x 48
- CONVOLUTE layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and the region they are connected to in the input volume. Convulation operator is defined for 2D as:

$$o[m,n] = f[m,n] * g[m,n] \tag{3}$$
$$= \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} f[u,v] g[m-u, n-v] \tag{4}$$

- ACTIVATION layer will apply the non-linear activation function, such as tanh or sigmoid.

$$h_{ij}^k = \tanh((W^k * x)_{ij} + b_k)$$

- POOL layer will perform a downsampling operation along the spatial dimensions (width, height), resulting in reduction in volume
- FULLY-CONNECTED layer will compute the class scores via softmax. This is similar to a regular layer in a fully-connected neural network where each neuron in this layer will be connected to all the numbers in the previous volume.The softmax function is as follows:

$$s(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

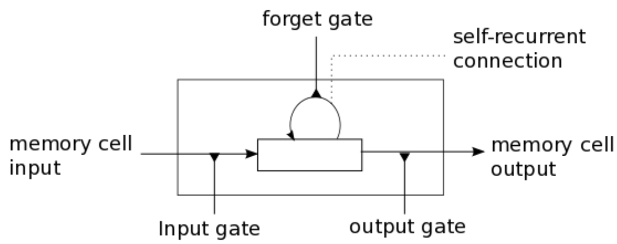

Fig. 4. Illustration of LSTM memory cell [19]

We use specific convolutional network architecture similar to [17]. Since the signals in our case are 1D, we experimented with kernels of sizes 1x5, 1x3, 1x16 as our aim for using CNNs is to capture the spatial activation in the signals. We train the CNN in the same manner as feed-forward neural network, using back-propagation and gradient descent.

B. Long-Short Term Memory Network

A recurrent neural network (RNN) enables us to exploit the temporal structure of the signals which might be ignored by a normal feed forward neural network. However a typical RNN suffers from vanishing gradient problem where the gradient signal gets so small that learning either becomes very slow or stops working altogether. It also effects the learning long-term dependencies capabilities of the network. Therefore, we use a long short-term memory (LSTM) network [18] that efficiently handles these issues. The fundamental unit of LSTM is a memory cell [Figure 4] which is composed of fourn main elements: an input gate, a neuron with a self-recurrent connection (a connection to itself), a forget gate and an output gate. The gates serve to modulate the interactions between the memory cell itself and its environment. [19]

The updates to the cell are as follows: First, we compute the values for i_t, the input gate, and $\widetilde{C_t}$ the candidate value for the states of the memory cells at time t :

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{5}$$
$$\widetilde{C_t} = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{6}$$

Second, we compute the value for f_t, the activation of the memory cells' forget gates at time t :

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{7}$$

Given the value of the input gate activation i_t, the forget gate activation f_t and the candidate state value $\widetilde{C_t}$, we can compute C_t the memory cells' new state at time t and with the new state of the memory cells, we can compute the value of their output gates and, subsequently, their outputs. [19]

$$C_t = i_t * \widetilde{C_t} + f_t * C_{t-1} \tag{8}$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o C_t + b_o) \tag{9}$$
$$h_t = o_t * tanh(C_t) \tag{10}$$

C. Autoencoder

Autoencoders are a popular for dimensionalilty reduction. They take a d-dimensional input and map it to a hidden d'-dimensional representation. This is mathematically represented in Eq.(11).

$$\boldsymbol{y} = \boldsymbol{s}(\boldsymbol{W}\boldsymbol{x}) \tag{11}$$

Here, x is the d-dimensional input and y is the d'-dimensional hidden representation. W is the weight matrix and s is a non-linear function, like $tanh$. An autoencoder should be able to reconstruct the input from the hidden representation as show in Eq.(12).

$$\boldsymbol{x} = \boldsymbol{s}(\boldsymbol{W'}\boldsymbol{y}) \tag{12}$$

Here W' is the weight matrix for the reverse mapping (and not the transpose of W). During autoencoder training, the parameters of this model (W and W') are optimized to minimize reconstruction error.
In our model, we use a multi-tier autoencoder. The encoder's input layer has 1280 nodes. The hidden and encoded layers have 320 and 80 nodes respectively. The decoder takes the 80-dimensional output of the encoder and reconstructs the 1280-dimensional input of the autoencoder through a 320-node hidden layer. We used $tanh$ as the activation for all layers. We used keras[20] to build our autoencoder.

D. Support Vector Machines

Support Vector Machines (SVMs) are a set of very powerful supervised learning methods used for classification, regression and anomaly detection. We use SVM in this project for classification. SVM tries to find a hyper-plane that separates the data. It identifies 'support vectors' that maximize the margin at decision boundary. The ability to use kernels in SVMs is a neat trick. We used a Gaussian (radial base function, or rbf) Kernel as it allows the data to be projected into an infinite dimensional space. The kernel is mathematically expressed in Eq.(13). We used sklearn[22] to build our SVM.

$$K(x, x') = \exp\Big(- \frac{||x - x'||^2}{2\sigma^2} \Big) \tag{13}$$

where, $||x - x'||$ is the squared Euclidean distance between x and x' and σ is the free parameter[21].

E. Final Models

In this project we attempted two three models in effort to find a generalized model:

1) CNN Model: On P300 Speller data set we tried convolutional neural networks. Network starts with a convolutional layer with 64 filters of length 9. Then a pooling layer with pool size 2 followed by an other convolutional layer with 32 filters, then another pooling layer with pool size 2. Both convolutional layers have stride of 1, and no zero-padding. Convolutional layers have depth of 160 to analyze signal from each channel in entirety which is 160 time unit long in P300 Speller data sets's case (0 to 667 ms).

Network ends with two fully connected layers with width 256 and 1. These layers use nonlinear activation function-

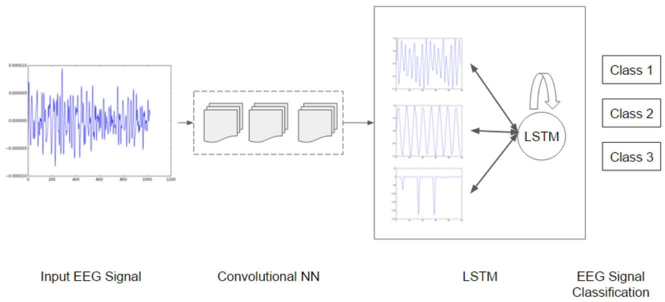

Fig. 5. Architecture of CNN-LSTM hybrid network

leaky rectifier. Introduced in [12], leaky rectifier compared to the standard activation functions has a nonzero gradient for negative input which helps in convergence.

Another version of convolutional neural network was tried on this P300 speller data set. This version convolved in two dimensions. Data was modified into a vector of 28050x160x8x8. Keeping everything same, convolutional and pooling layers were made 2-dimensional. We tried with 64 filters of 4x4 shape. Number of convolutional and pooling layers was reduced to 1.

Convolutional neural networks were built using Lasagne framework [14].

2) Autoencoder-SVM Model: We propose a hybrid model with an autoencoder layer that learns a different representation of the EEG data. We then remove the decoder and then fit this encoded data with an SVM. SVM has an rbf kernel which allows an infinite-dimensional representation of the data. We expect the autoencoder to learn distinguishing characteristics of the EEG. With these new features, we hope the SVM can classify the signals more accurately. We tested our model with both raw data and denoised data.

3) CNN-LSTM Model: Taking ideas from the current advances in deep video and speech processing systems, we propose another hybrid deep network to tackle the task of EEG classification which involves using combinations of convolutional neural networks (convnets) to obtain vector representation of signals and recurrent neural networks to decode those representations into class label. The essential ideology behind the model is to consider an EEG sample to built from series of frames where each frame represents the EEG signal at a time unit. In order to extract the spatial features from each frame per unit we plan to use an Convolutional neural network in the first phase (analogous to encoder) and then in-order to take advantage of temporal nature of the signals, we'll use LSTMs for the second phase (analogous to decoder) 5. Thus via this hybrid architecture we aim to capture both spatial and temporal dependencies in the EEG signals.

V. Experiments and Results

A. P300 Speller data set

Experiments with various configurations were performed on this data set. Data was heavily imbalanced inherently as explained in the preprocessing step. Training on original data set, without duplication of minority class, majorly failed. Predicting only majority class, network should get an accuracy of 82.33 which is very close to observed empirical results of 84.5% accuracy.

Results got better after class imbalance was removed. Learning with input depth as 1 and convolving over both channel and time-sequence as 160x64 shaped input instance showed results better than random prediction but converged to 64 67% accuracy.

Best results were observed when input depth was changed to 160 and entire signals from each channel is processed. Learning rate was altered as shown in table II.

TABLE II
Learning rate variations

Epochs	Learning Rate
0 - 150	0.1
150 - 350	0.01
350 - 450	0.005

Experimental training with a two dimensional convolutional network was also tried. Ideology was to explore efficient co-relation between greater number of channels. Data was converted into a 28050x160x8x8 vector and convolving and max pooling operations were made 2 dimensional. Added advantage of quicker filter dimension reduction helped us in removing second conv layer and pooling layer. Empirical results were contrary to initial belief and network got to 71 74% after 450 epochs. Final results are detailed in table III

TABLE III
Final classification accuracy scores on P300 Speller data set

Methodology	Accuracy
Imbalanced classes (Random: 83.33%)	84.50%
Convolving time and channels (Random: 50%)	66.70%
1-Dimensional CNN with entire time (Random: 50%)	86.20%
2-Dimensional CNN with entire time (Random: 50%)	71.80%

B. Epilepsy Seizure Detection and BCI-125 data sets

We tried tuning our Autoenoder-SVM model based on denoised data vs raw data, single channel vs mutli channel, and auto-encoder vs no autoencoder. The results are shown in table IV. We also tried various thresholds for denoising-1.5, 2, 2.5, 3, 3.5.

TABLE IV
AutoEncoder-SVM Test

AutoEncoder	MultiChannel	Denoising	Train Acc	Test Acc
No	No	No	0.66	0.39
No	No	Yes	0.84	0.41
No	Yes	No	0.67	0.37
No	Yes	Yes	0.88	0.43
Yes	No	No	0.53	0.39
Yes	No	Yes	0.76	0.40
Yes	Yes	No	0.57	0.39
Yes	Yes	Yes	0.72	0.42

For the CNN+LSTM model, we experimented with convnets layers: single and double, along with filters sizes: 1x3, 1x5 and

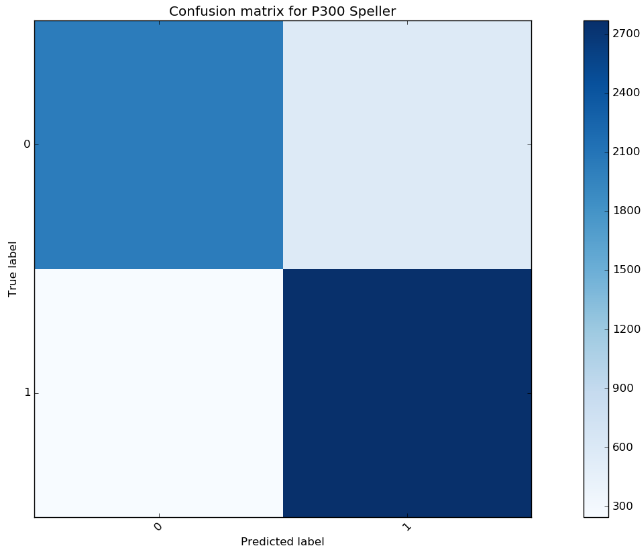

Fig. 6. Confusion Matrix for P300 Speller data set

1x16. For the decoder we only kept one LSTM layer followed by a fully connected layer and softmax layer. We trained the model by stochastic gradient descent with cross entropy being the loss function. We used Keras [20] for implementing the LSTM layer.

See table V for final results. In the case of epilepsy dataset our models did not perform any better than the random chance.

TABLE V
FINAL CLASSIFICATION ACCURACY SCORES ON THE DATA SETS

Data \Model	LSTM+CNN	AutoEncoder-SVM
Epilepsy	20.14	NA
BCI	31.34	48.99

VI. DISCUSSION

Results on P300 Speller data set with convolutional networks give encouraging results. As can be verified from confusion matrix in Figure 6, our model is able to successfully analyze EEG signals determining correct class of intensifications with nice precession.

Empirical results from two dimensional convolutional network is contrary to initial hypothesis. Probable reason behind this can be reduced net number of filters. Experiments without pooling layer may give better results.

As can be seen in the results, our models did not perform well on epilepsy data. We checked for oscillations in weight so learning rate was not the issue. We believe the following causes might have resulted in poor performance:

- Many feature processing methods used for EEG signals like wavelet-based features and time frequency-based (DWT, PSD, DFT) are linear operators. Our neural networks have a sigmoid or hyperbolic tangent function in the computation path, which might make it harder to simulate a linear operator closely enough. Also, a full DFT is an N by N matrix multiplication. Therefore a neural net has to be big enough to represent that many multiplications (at minimum O(NlogN)). [23]
- A large factor for success of neural network lies in the fact that there has been advancement in large classification data set on which they can be trained. Our data sets are pretty small in this regard (500 observations in epilepsy and 2800 for BCI task) which might be one of the probable cause.
- The inherent noise in the EEG data sets is maybe hindering from learning anything meaningful. Low signal-to-noise ratio might be the reason for this as many noise sources encountered with the EEG signal. Noise sources can be non-neural (eye movements, muscular activity, power-line noise) or neural (EEG features other than those used for control). However, we decided not focus on this assuming that processing noisier data would have better generalization properties, as is the case with neural networks.
- The total change in the weights was on the order of 10^{-8}, which indicates that little is actually being learned. Possible reasons for the poor performance is that conv layers were inadequate for representing the signal, or that network was constantly falling into local optimum.

VII. FUTURE WORK

The new approach proposed in this report for EEG data analysis didn't perform well for probable causes detailed in the Discussion section. Classification of generic EEG data without any task-specific and hand-crafted features is a difficult task. We hope that our finding will encourage more research in direction of a generic deep learning model of EEG analysis.

As a next step, we think autoencoders could be explored in-depth for better results. Autoencoders, while doing a decent job at dimensionality reduction, ignore the spatially-local correlations of EEG data. Classification of EEG data depends on identification of trend in time series and to discover localized features that repeat themselves. We feel convolutional autoencoders, autoencoders with local weights sharing and therefore preserving spatial locality, can be used here and should give better results in comparison to vanilla stacked autoencoders. We expect such a model to scale better with longer time sequences and more features.

Furthermore, we hope some research can be stemmed in the direction of unsupervised learning. Unsupervised training of an autoencoder removes all outliers in the training data set and hence they find an interesting application in anomaly detection. Getting labelled data sets for EEG is hard but there is tons of unlabelled EEG data ready for use in a semi-supervised or unsupervised learning. Anomaly detection using autoencoders seems to be a nice way to approach unsupervised epilepsy episodes or seizure detection. On the other hand, as EEG data varies a lot across individuals hence an autoencoder trained on a person's EEG data may classify everything as an anomaly for a different individual. It will be interesting to see if this pans out well for epilepsy episode and seizure detection on EEG data.

STATEMENT I

We hereby state that all the work presented in this report is that of the authors.

STATEMENT II

Ayush worked on P300 Speller data set and implemented Convolutional Network models. Neeth managed BCI data set and implemented Autoencoder+SVM model. Harsh worked on Epilepsy data set and implemented CNN+LSTM model.
Every author equally contributed to writing the report.

REFERENCES

[1] http://docs.scipy.org/doc/numpy-1.10.0/reference/routines.fft.html
[2] http://epileptologie-bonn.de/cms/front_content.php?idcat=193&lang=3&changelang=3
[3] MahbubulAlam, Md. Imdadul Islam, and M. R. Amin, "Performance Comparison of STFT, WT, LMS and RLS Adaptive Algorithms in Denoising of Speech Signal," IACSIT International Journal of Engineering and Technology, vol.3, no.3, June 2011, pp. 235- 238.
[4] ElifDerya Ubeyli, "Analysis of EEG Signals by Implementing Eigenvector Methods/Recurrent Neural Networks," Digital Signal Processing, vol. 19, no. 1, 2009, pp.134-143.
[5] Hamid R. Mohseni, A. Maghsoudi and Mohammad B. Shamsollahi, "Seizure Detection in EEG signals: A Comparison of Different Approaches," In ConfProc IEEE Eng Med BiolSoc, 2006, pp. 6724-6727.
[6] K Sivasankari , and K Thanushkodi , "An Improved EEG Signal Classification Using Neural Network with the Consequence of ICA and STFT", Journal of Electrical Engineering and Technology, vol. 3, no. 3, May 2014.
[7] L. Farwell and E. Donchin, "Talking off the top of your head : toward a mental prosthesis utilizing event-related brain potentials," Electroencephalography and Clinical Neurophysiology, vol. 70, no. 6, pp. 510–523, 1988.
[8] B. Blankertz, "Bci competition III webpage." [Online]. Available: http://www.bbci.de/competition/iii
[9] http://www.bbci.de/competition/iii/desc_II.pdf
[10] Alain Rakotomamonjy and Vincent Guigue, "Ensemble of SVMs for BCI P300 Speller" [Online]. http://asi.insa-rouen.fr/enseignants/~arakoto/publi/rakoto_bci.pdf
[11] P.W. Mirowski, Yann LeCun, Deepak Madhavan, and Ruben Kuzniecky, "Comparing SVM and Convolutional Networks for Epileptic Seizure Prediction from Intracranial EEG", In IEEE's Machine Learning for Signal Processing, 2008, pp. 244-249.
[12] http://web.stanford.edu/~awni/papers/relu_hybrid_icml2013_final.pdf Andrew L. Mass, Awni Y. Hannun and Andrew Ng, "Rectifier Nonlinearities Improve Neural Network Acoustic Models", ICML, 2013.
[13] http://www.bsp.brain.riken.jp/~qibin/homepage/Datasets_files/EEG_BCI_MI_AllSub.zip
[14] http://lasagne.readthedocs.org/en/latest/
[15] http://deeplearning.net/tutorial/lenet.html
[16] http://cs231n.github.io/convolutional-networks/
[17] P. W. Mirowski, D. Madhavan, Y. LeCun, "Time-Delay Neural Networks and Independent Component Analysis for EEG-Based Prediction of Epileptic Seizures Propagation", Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, AAAI Press 2007.
[18] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
[19] http://deeplearning.net/tutorial/lstm.html
[20] http://keras.io/
[21] https://en.wikipedia.org/wiki/Radial_basis_function_kernel
[22] http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC
[23] http://stackoverflow.com/questions/10336682/neural-networks-cant-figure-out-fourier-transforms