Decoding of Electroencephalography (EEG) Signals using Neural Networks

Rushi Bhatt
University of California, Los Angeles
rushibhatt@g.ucla.edu

Shivani Ganti
University of California, Los Angeles
sganti@g.ucla.edu

Ronak Kaoshik
University of California, Los Angeles
ronak42@g.ucla.edu

Subhiksha Mani
University of California, Los Angeles
subhiksha@g.ucla.edu

Abstract

This report aims to provide insights on Brain Computer Interaction (BCI) Competition's electroencephalography (EEG) data [1] by employing various deep learning techniques and models. Using this temporal EEG data, the classification task is to predict one of four actions performed by a subject.

The models explored include Convolutional Neural Networks (CNN),Recurrent Neural Networks (RNN), Long Short-Term Memory Networks (LSTM), Variational Autoencoder (VAE), and their hybrid approaches combining select models. These models were further iterated upon with preprocessing steps such as adding random noise, stratified sampling, novel low pass filtering for smoothing EEG and data augmentation.

This paper will target a performance comparison for each of these models, while also providing additional experiments to better understand performance across all subjects and number of time samples.

1. Introduction

This report constructs a CNN, RNN, LSTM, CNN+LSTM, CNN+RNN, and VAE + Logistic Regression, with an emphasis on CNN and CNN+LSTM as they proved to be the two best models after performing extensive experimentation.

1.1. CNN

The proposed CNN model consists of three convolutional blocks with each block consisting of a 2-dimensional convolutional layer, and a 2-D max pooling layer. Following each convolution layer, a dropout of 0.5 as well as batch normalization layer is added to avoid overfitting. The Exponential Linear Unit(ELU) was used as the activation for the 2-D convolutional layers. The three blocks are followed by an output fully-connected layer using softmax activation. This was the best performing model. Reference the architecture details in **Figure 3** and **Table 1**.

1.2. RNN

The RNN model uses 3 GRU layers with a dropout of 0.5 and relu activation.It is followed by a batch normalization and dropout layer to reduce overfitting. The final output layer uses the softmax activation.

1.3. LSTM

The Basic LSTM model makes use of two LSTM layers using the elu activation followed by batch normalization layers. The final connected layer makes use of the softmax function.

1.4. CNN + LSTM

The CNN+LSTM model was developed to have the CNN layers used to extract features from the input data and the LSTM layers are included to help with sequence prediction. The hybrid model consists of 3 convolutional blocks with each block consisting of a 2-dimensional convolutional layer, and a 2-D max pooling layer. Two LSTM layers with a dropout of 0.6 are also added. The output layer still makes use of the softmax activation.

1.5. CNN + RNN

The CNN+RNN model makes use of two CNN blocks, with each block including a 2-D convolution layer, a 2-D max pooling layer, batch normalization layer and dropout layer. It is followed by two RNN layers and an output layer.

1.6. VAE + Logistic Regression

Variational Autoencoder networks contain encoder and decoder components. For our experiment, the encoder layer comprises of two convolution blocks and two fully-connected layers which obtain an encoded vector from the

raw input. This vector is then decoded using the decoder network which consists of a fully-connected layer followed by three inverse convolutional blocks. The VAE loss function is formed using the reconstruction loss as well as the KL-divergence loss. After sufficient training, the encoder weights are used to obtain the latent space embeddings. Finally, a logistic regression model is used for classification of these embeddings.

2. Results

The best performing model was the 3-layer CNN using preprocessed EEG signals, augmented with a novel low pass filter with a best test accuracy of 76.46%. Reference the confusion matrix for this model in **Figure 1**. To understand how this model was iterated upon from the baseline score of 55%, reference **Figure 4**. Because the problem at hand is a multiclass classification, only test accuracy is not the best measure of the performance of the classifier, and therefore confusion matrix as well, as class-wise precision, recall and F1-score are computed in **Figure 1** and **Table 5** respectively.

2.1. Model Comparisons

Table 1 displays the best test accuracy for all the six models along with a brief architectural description.

2.2. Accuracy vs. Preprocessing Techniques

Table 2 displays the test accuracy for models with varying preprocessing and sampling techniques. These are computed for the top two models: CNN and CNN+LSTM. Training over epochs plots for the CNN training and validation accuracy can be found in **Figure 6** and **Figure 7**.

2.3. Accuracy vs. Subject Comparison

The test accuracy for each subject using CNN and CNN+LSTM models can be found on **Table 3**.

2.4. Accuracy vs. Time Period Comparison

Table 4 displays the test accuracy over various time samples for CNN and CNN+LSTM models.

3. Discussion

3.1. Tuning Hyperparameters

The parameters that were tuned during the experiments were batch size, learning rate, number of epochs, number of layers, filter size, kernel size, type of activation layers and type of optimizer.

After numerous experiments, incremental filter sizes for each of the layers proved to be more efficient than repeating the same filter size as data varies from layer to layer. Increasing epochs also allows for a longer time for the model to learn and accurately train the data. Accuracy of some models began stagnating or decreasing after experimenting with too large of epochs, so a nominal value of 50 was chosen as it was also faster to run. Increasing the batch size to 200 allowed for more samples to be worked through and a higher accuracy rate. Despite starting out with a four-layer model, a three-layer CNN model was more promising. Since the model makes use of batch normalization, it allowed for higher learning rates to be implemented and for the accuracy to converge faster.

3.2. Preprocessing

After conducting multiple experiments, it was observed that the models are able to perform exceptionally well on the validation set, however, the test set performance lagged significantly. We figured out that the test set signals on an average have more high frequency noise as opposed to the train set. Hence, we made use of tuned low pass filters to further smoothing the dataset, refer to **Figure 5**. Then, these denoised signals were utilized as augmented data points, along with the original EEG signals for training the classifier. In addition, a slew of preprocessing techniques like trimming, max pooling, averaging, and adding noise gave a significant improvement in the model performance. As can be seen from the results stated in **Table 2**, CNNs have an improvement in test accuracy from 55% to 76.46% and similarly for CNN+LSTMs.

A plot of the 22 channel-EEG signal from the raw dataset showcase that the majority of the information is stored in the initial half of the time-series sample and the later half consists of noise. So the very first pre-processing step is trimming half of the series. It is followed by max pooling operation in a temporal sense which leads to further halving of the series. Similarly, another series is obtained using averaging in a temporal sense and adding white noise. These two series are concatenated to form an even larger training set. A final step involves sub-sampling to further augment the dataset. The operations of max-pooling, averaging and sub-sampling helped to preserve significant information and this is ascertained from the results. The deep-networks are able to learn well from such a well pre-processed and augmented dataset as opposed to the raw dataset. The use of filtered and unfiltered data in the train set strongly enforces the models to understand the inherent shape and this explains the boost in performance.

3.3. Train/ Validation split

The first splitting technique randomly shuffles the data and obtains a randomly indexed validation set of 18% the entire train set. However, this method does not take into account the data imbalanced distribution (across class and subjects) in train and validation set.

Hence, we finally implemented stratification across all

subjects and all classes. It is implemented by creating equal splits across all 36 labels (4 classes and 9 subjects) and then including them in train-validation splits using the aforementioned ratio of 18 %. This led to an improvement in performance across all classes as seen in **Table 2**.

3.4. Comparison Across Subjects

This experiment compares accuracy using two different preprocessing techniques across all nine subjects. Results can be found in **Table 3**.

Regardless of the choice of model or preprocessing approach, some subjects consistently seem to perform either extremely well or poorly. For instance, subjects 0, 4, and 8 routinely score amongst the highest of all subjects, whereas subjects 1 and 5 regularly score amongst the lowest. The results perhaps suggest that the variability amongst subjects remains the same across CNN-based models with similar sampling techniques.

3.5. Comparison Across Time Periods

In this experiment, we focus on exploring how many time samples are required to get a required to get a reasonable classification accuracy. All the tests are performed on preprocessed EEG data, which consists of 250 time samples for each EEG channel for each subject. Also, the experiments were conducted for both random sampling as well as stratified sampling.

From the results in **Table 4 and Figure 8** , we observe that for the proposed 3-layer CNN model, obtains a fairly high accuracy only with 125 time samples, which is about 50 % of the entire time sequence. For the proposed CNN + LSTM model, the accuracy saturates after 150 time samples, which is 60 % of the entire time sequence.

Moreover, from the experiments with and without stratification, we observe that the stratification test accuracy almost always outperforms the random selection accuracy, regardless of the model. From this, we can further conclude that equally stratifying data (over 36 classes), and then preparing train/ validation splits boosts the overall classification performance.

3.6. Leave One User Out Analysis

In this experiment, we aimed at exploring the variance among each subject. Moreover, this experiment is also a test of the generalization power of our proposed 3-layer CNN model.

First, we leave out one of the user from the training dataset and use them as test set, and train the model on the remaining eight subjects. We further use the same model and evaluate it against the actual test set and the results are compiled in **Figure 9**.

From **Figure 9**, we can see that subject 6 has the lowest leave one out accuracy (model trained on subject 0,1,2,3,4,5,7,8) about 90 %, however all the other subjects have a very high leave one out accuracy in the range of 97 %. Thus, subject 6's EEG data is not a generalized representation of the entire dataset and has a overall higher variance.

Moreover we also observe that for all the users, the accuracy for the leave one user out scenario is about 25 % higher than the accuracy on the test dataset. This gives us an indication that probably the test set has a lot of noisy samples, or is drawn from a different distribution than the training set. This further enforces the exploration of various noise reduction/filtering techniques which have been described in preprocessing section.

3.7. Comparison Across All Models

We experimented with six models namely; CNN, CNN+LSTM, CNN+RNN, LSTM, RNN and VAE.

Out of these, the 3-layer CNN model performed the best with an accuracy of 76.46 % while LSTM performed the worst with an accuracy of 25.06 %.The basic models of CNN and RNN performed poorly, but through observation, the CNN model provided a larger accuracy between the two. Therefore, the hybrid models of CNN+LSTM and CNN+RNN were explored since LSTMs and RNNs help with sequence detection. The ensemble approaches featuring CNNs seem to be much more accurate than their basic model counterparts (LSTM, RNN) with accuracies of 70.9 % and 63.4 %. We also explored Variational Autenocders in conjunction with Logistic regression. After thorough tuning of the VAE network, we were able to achieve an accuracy of 55.72% for classfying the embeddings obtained from the VAE encoder. We made use of a basic logistic regression model for classifying the embeddings. VAEs took significant amount of time for training and still performed poorly.

Since CNNs are faster and easier to train, most of the focus was towards improving the basic CNN and CNN hybrid models. Basic RNNs have the problem of exploding or vanishing gradients and thus aren't as stable and performed poorly despite adding batch normalization and dropout. Meanwhile, LSTMs are able to maintain information in memory for long periods of time. The better performance in LSTMs over RNNs in the hybrid models can be attributed to this fact.

The best performing model is the 3-layer CNN, along with preprocessing, train/ validation stratification and data augmentation using novel low pass filter. The incremental increase in the performance can be seen in **Figure 4**. Also, from the box plot of test accuracy as shown in **Figure 2**, we observe that the proposed model and augmentation technique significantly boots performance (peak accuracy of 76.46 %), as well as consistently reaches a test accuracy of greater than 74 % over multiple iterations.

References

[1] Organizers of BCI Competition IV (2008) *https://www.bbci.de/competition/iv/*, BCI Competition IV.

[2] Robin Tibor Schirrmeister et al (2018) *https://arxiv.org/pdf/1703.05051.pdf*, Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human EEG.

[3] Li X, Zhao Z, Song D, et al. Latent Factor Decoding of Multi-Channel EEG for Emotion Recognition Through Autoencoder-Like Neural Networks. Front Neurosci. 2020;14:87. Published 2020 Mar 2. doi:10.3389/fnins.2020.00087

4. Tables/ Figures

Models	Architecture	Best Test Accuracy
CNN	3 x Conv 2d(25, 50 & 100) FC Layer(4)	**76.46 %**
RNN	3 x GRU(50, 25 & 25) 2 x FC Layer(20 & 4)	39.48 %
LSTM	2 x LSTM(32 & 32) FC Layer(4)	25.06 %
CNN + RNN	2 x Conv 2d(16 & 32) 2 x GRU(128 & 64) FC Layer(4)	64.9 %
CNN + LSTM	3 x Conv 2d(100, 100 & 100) FC Layer(100) 2 x LSTM(100 & 70) FC Layer(4)	72.12 %
VAE + Logistic Reg.	Encoder: 2 x Conv 2d(32 & 64) 2 x FC Layer(16 & 2) Decoder: Dense(2496) 3 x Conv 2dT*(64, 32 & 1)	55.72 %

Table 1. Model comparison

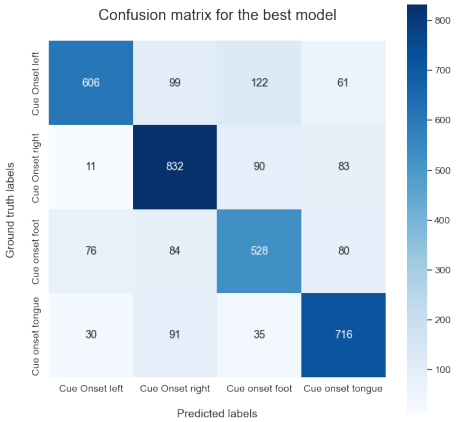

Figure 1. Confusion Matrix for 3-layer CNN with Low Pass Filter Augmentation

Pre-processing	Train/ Val split	CNN	CNN + LSTM
Without Preproc.	Random Sampling	55 %	49 %
With Preproc.	Random Sampling	72 %	70.8 %
	Stratification	**76.46 %**	72.12 %

Table 2. Accuracy with v/s without Preprocessing Techniques

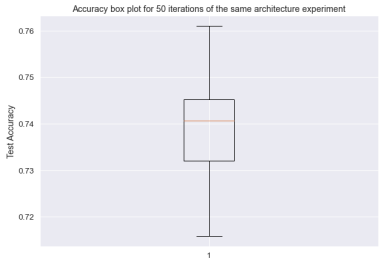

Figure 2. Boxplot of test accuracy for 3-layer CNN with Low Pass Filter Augmentation

Subject	CNN		CNN + LSTM	
	Random	**Stratify**	**Random**	**Stratify**
0	70 %	63.5 %	63.5 %	55 %
1	49.5 %	52 %	49.5 %	46.5 %
2	68 %	68 %	62 %	70 %
3	73.5 %	65 %	64.4 %	59 %
4	78 %	80 %	75 %	72 %
5	48 %	56 %	39 %	57 %
6	72.5 %	68.5 %	50 %	56 %
7	61 %	63 %	54.5 %	53.5 %
8	73 %	84 %	73 %	79 %

Table 3. Test Accuracy vs. Subject Comparison

Time Samples	CNN		CNN+LSTM	
	Random	**Stratify**	**Random**	**Stratify**
25	38.6 %	40.6 %	38.4 %	41.4 %
50	53.7 %	54.6 %	50.1 %	51.1 %
75	58.4 %	58.6 %	54.7 %	54.1 %
100	65.6 %	62.9 %	57.9 %	59.7 %
125	70.0 %	69.9 %	60.1 %	63.0 %
150	70.2 %	72.6 %	65.7 %	65.8 %
175	68.4 %	67.9 %	65.1 %	66.6 %
200	69.8 %	70.7 %	66.2 %	64.7 %
225	68.3 %	69.2 %	64.6 %	64.7 %
250	70.6 %	70.7 %	65.9 %	65.8 %

Table 4. Test Accuracy vs. Time Period Comparison

	Precision	Recall	F1-Score	Support
Class 0	0.84	0.68	0.75	888
Class 1	0.75	0.82	0.78	1016
Class 2	0.68	0.69	0.68	768
Class 3	0.76	0.82	0.79	872
Accuracy			**0.76**	3544
Macro Avg.	0.76	0.75	0.75	3544
Weighted Avg.	0.76	0.76	0.76	3544

Table 5. Classification Performance metrics for 3-layer CNN with Low Pass Filter Augmentation

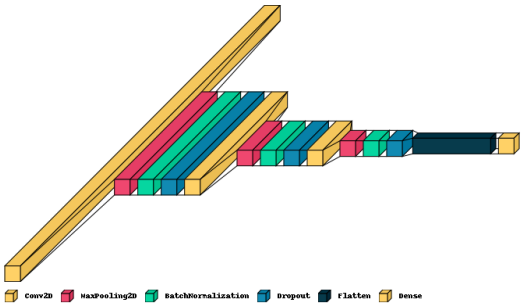

Figure 3. Architecture Overview of 3 layer CNN

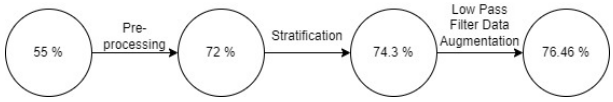

Figure 4. Incremental improvement of Test Accuracy

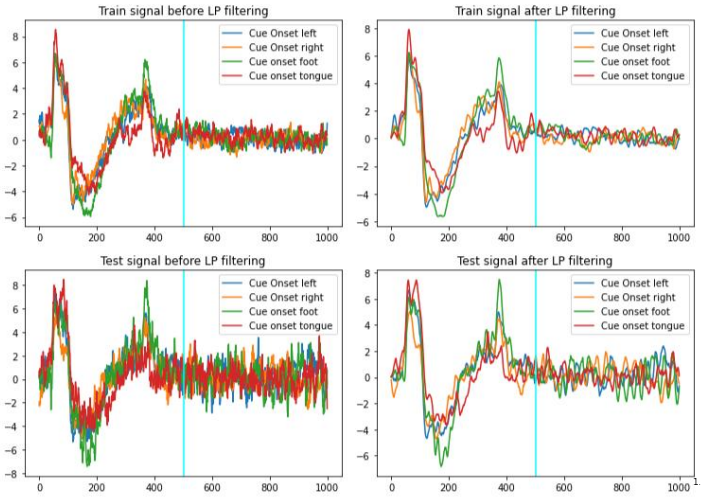

Figure 5. Train/ Test EEG before and after LP filter

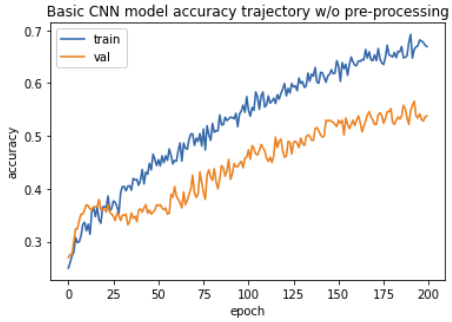

Figure 6. CNN Accuracy w/o preprocessing

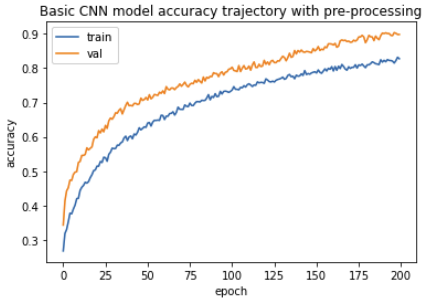

Figure 7. CNN Accuracy w/ preprocessing

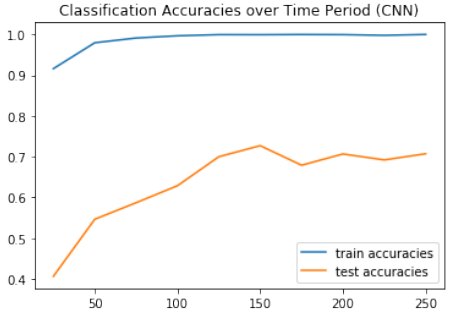

Figure 8. CNN Accuracy over Time Period

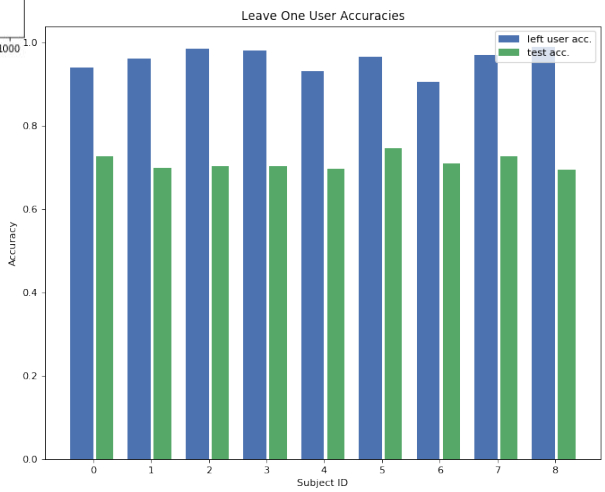

Figure 9. CNN Leave One User Out Accuracy v/s Subjects