EC ENGR C147 Project Writeup

Jack He Yuheng Ding Allen Wang James Jin
{jackhe313, yhding, allenwang2333, kaichenj7012}@ucla.edu

Abstract

In this project, we explored a variety of model architectures for EEG signal analysis, including CNN, RNN, attention-based models, Transformers, and hybrid models.We carried out four sets of experiments to evaluate the impact of various hyperparameters on a train-on-all and test-on-all setting as well as on a train-on-single and test-on-single setting, the influence of selecting different number of time bins for the EEG signals, and the effect of data augmentation on model performance. Our findings show that a CNN specifically designed for EEG outperforms other models across all experiments while hybrid architectures exhibit diminished performance. We also observed that the models achieve optimal results when just more than half of the available time bins were selected. In contrast to our initial expectation, data augmentation did not seem to enhance the models' performance in processing EEG signals.

1. Introduction

To resolve the complex patterns of neural activities using the EEG dataset, we have implemented multiple models with different architectures, including CNN, RNN, and other post-CNN models, to better learn the spatial-temporal dynamics of the brain signals. However, the motivations behind each architecture vary. The CNN model is chosen to better extract the spatial features of the data while the RNN model and the LSTM units further capture the temporal relationship among the data. Moreover, attention-based models emphasize the most significant parts of the model while Transformer-based models excel at recognizing patterns over long distances in the data. In an attempt to combine some of the stated benefits together, we have also implemented some hybrid models like CNN + RNN, CNN + Transformer, and others. The specific results about each model's performance is given in the Results section.

2. Data Preprocessing

When loading the data from pre-saved files, we first load the trainval and testing datasets along with each of their labels. Next, we execute a random split, utilizing a custom ratio of 0.8, to segregate the loaded data into distinct training and validation sets. If a specific subject or person is selected, then the data is filtered to include only those related to that particular subject. Otherwise, if the subject argument is set to -1, then all subjects' data is selected. Next, we adjusted the labels by subtracting them by 769 to ensure that they start at 0. To conform to the expected input shape for CNN-related models, we add an extra dimension of channels to the data. If specified, data augmentation is applied; the details of these augmentation techniques are provided in the Appendix. Moreover, their effectiveness will be tested in one of our experiments.

3. Results

For this project, we have tested the following eight models: a simplified ResNet architecture (MiniResNet), a CNN model designed specifically for EEG signal processing (EEGNet [1]), a general RNN model, a general CNN + RNN model, an EEGNet + RNN model, an EEGNet + Attention model, an EEGNet + Multi-head Attention model, and a general CNN + Transformer model. For more specific structure of each model, please refer to the Appendix.

We have implemented four experiments in total: 1) train on all subjects and test on all subjects; 2) train on a single subject and test on a single subject; 3) accuracy as a function of time; 4) effect of data augmentation. For optimizer details, please also refer to the Appendix. Training hyperparameters such as batch size remain consistent for each involved model across all experiments.

3.1. Experiment 1: Train on All, Test on All

For the first experiment, we have searched for eight sets of training hyperparameters (each set includes number of epochs, batch size, step size and gamma for learning rate scheduling) that provide the most decent accuracy for each model respectively when trained on all 9 subjects of the dataset and tested on all 9 subjects. The accuracy overview for the eight models are given in Table 1 under the Appendix. As one can see, EEGNet has achieved the highest test accuracy of 0.7133 across all eight models, followed by the hybrid models involving EEGNet. In other words, the hybridization does not boost the performance beyond

the single EEGNet structure. What comes next is the general CNN + RNN structure, obtaining a test accuracy of 0.4266, which is in the middle range of all the obtained test accuracy and shows a noticable increase from both the standalone CNN (MiniResNet) and RNN model. Also, it is worth mentioning that the hybrid of CNN and Transformer model yields a poor accuracy.

3.2. Experiment 2: Train on Each, Test on Each

For the next experiment, we tested four models in particular since they obtain the highest accuracy in the previous experiment: EEGNet, EEGLSTMNet, EEGAttentionNet, and EEGMultiAttentionNet. The first part of this experiment involves training the models on each of the nine subjects and testing them on the corresponding subject. For instance, each model is trained on subject 0 and then tested on subject 0 for evaluation, etc. The results are shown in Figure 1 under the Appendix. Similarly, one could easily observe that EEGNet has the best performance across almost all trials. However, the remaining models' performance ranking varies according to the subject. The difference in accuracy between EEGNet and the rest models is the most apparent on subject 1 and subject 4. Moreover, another interesting trend is that all four models perform relatively poorly on subject 1 and subject 5.

The second half of this experiment involves two testings 1) train on all subjects but test on subject 0, and 2) train on subject 0 but test on all subjects. The results are also shown in Figure 1. When trained on all, and tested on subject 0, almost all four models perform better than their own counterpart of training on subject 0 and testing on 0. When trained only on subject 0 and tested on all subjects, all four models perform poorly, but their accuracy hierarchy still persists.

3.3. Experiment 3: Accuracy VS Time Bins

In this experiment, we evaluated three models' accuracy with respect to how many time bins are sampled for both training and testing. Instead of evaluating EEGAttentionNet and EEGMultiAttentionNet separately, we have chosen to only test on EEGMultiAttentionNet, in addition to EEGNet and EEGLSTMNet. The default time bins selected for all models across all experiments are 600. To observe how the number of time bins selected affects the model's performance, we ran 5 sets of tests with 200, 400, 600, 800, and 1000 time bins selected respectively with a train-on-all and test-on-all setting. The results are shown in Figure 2 under the Appendix. Interestingly, the trend shown in the figure is quite obvious. EEGNet almost always outperforms EEGLSTMNet, which in turn, always outperforms EEGMultiAttentionNet. In addition, there is a parabolic-like relationship between the models' performance and the time bins selected. All three models' accuracy increases first as the number of time bins increases from 200 to 600. At 600 time bins, both EEGNet and EEGLSTMNet reach their own maximum accuracy while EEGMultiAttentionNet's accuracy continues to rise until it peaks at 800 time bins. Beyond these peaks, the accuracy of all models declines as the number of time bins further increases to 1000.

3.4. Experiment 4: Effect of Data Augmentation

For the last experiment, we incorporated two types of data augmentation into our training process: Gaussian noise and channel drop. The details of each technique can be found in the Appendix. Again, three models are tested here: EEGNet, EEGLSTMNet, and EEGMultiAttentionNet. The accuracy comparisons are displayed in Figure 3 under the Appendix. This plot also demonstrates some interesting trends. Similarly, EEGNet consistently performs better than EEGLSTMNet, which in turn, always has higher accuracy than EEGMultiAttentionNet. Nonetheless, applying these data augmentation techniques does not boost the models' accuracy. When applied independently, both Gaussian noise and channel drop reduces the models' performance on a train-on-all and test-on-all setting with the latter method lowering the accuracy more heavily. Moreover, while using both methods together still results in lower accuracy for all models compared to training without data augmentation, their performance shows an improvement compared to using just one of these methods.

4. Discussion

4.1. Accuracy Across All Subjects in Experiment 1

Through our first experiment (results shown in Appendix: Table 1), we found out that EEGNet and its derivatives outperform all other CNN, RNN, hybrid, and attention-based models. Adding extra layers, such as LSTM and attention after EEGNet does not show signs of accuracy improvement. We hypothesize the superior performance of EEGNet due to the following reasons.

4.1.1 Domain Specific Architecture

EEGNet is a special convolution architecture that is designed for EEG data classification. It is a compact model that effectively combines temporal and spatial information through convolution layers. Adding LSTM or Attention modules after the convolution layers does not help may partly due to the reason of redundant spatial data representation in the architecture.

4.1.2 Overfitting Resistance

EEGNet utilizes a simple architecture, which means the number of parameters is relatively small. Compared to other architecture that has significantly more parameters and considering the limited number of data in the dataset,

EEGNet is resistant to overfitting during training. It is better at capturing the true relationship of data distribution instead of memorizing them.

4.2. Accuracy Among Subjects in Experiment 2

Experiment 2 (results shown in Appendix: Figure 1) provides a clear indication of each tested model's ability to decode EEG signals, with significant variability in accuracy among subjects. We thereby propose the following reasons to explain the overarching trend.

4.2.1 Poor Performance on Specific Subjects

All models perform poorly on subjects 1 and 5, indicating that there may be subject-specific factors that are not being captured by any of the models. These factors could include noise in the data, or inherent variability in the EEG signals.

4.2.2 Cross-Subject Generalization

In the second half of the experiment, training on all subjects but testing on subject 0 yields better performance than training and testing on subject 0 alone. This suggests that incorporating a diverse set of training data can improve the robustness of the model for individual predictions. In contrast, training only on subject 0 and testing on all subjects results in poor performance across all models, emphasizing the difficulty of generalizing a model trained on a single subject's data to a broader population.

Despite the decrease in performance when generalizing from training on a single subject to testing on all, the relative accuracy ranking of the models remains consistent. This persistent hierarchy suggests that the core characteristics that lead to a model's success or failure are fundamental to the model and not solely dependent on the training set's diversity.

4.3. Accuracy Across Time Bins in Experiment 3

Experiment 3 (results shown in Appendix: Figure 2) aims to compare the accuracy of three different neural network models in processing EEG data as the number of time bins used for training and testing varies. We propose the following hypotheses to interpret the observed trends.

4.3.1 Poor Performance on Short and Long Time Bins

The performance declined when time bins were either too large or too small. A 200 time bin resulted in insufficient data for the models to accurately distinguish between categories, likely due to lost of critical information. Conversely, at 1000 time bin, performance worsened, suggesting that excessive time bin expansion may impair predictive ability, potentially due to overfitting or increased noise.

4.3.2 Optimal Time bins

The superior performance at 600 time bins for both EEGNet and EEGLSTMNet, and at 800 for the EEGMultiAttentionNet, likely stems from an optimal balance that involves sufficient temporal data to capture the critical patterns within the EEG signals, making the computation more efficient to avoid overfitting.

4.4. Data Augmentation Accuracy in Experiment 4

The results of Experiment 4 (shown in Appendix: Figure 3) suggest that while data augmentation is a powerful tool in many domains to improve model generalizability, its effectiveness varies. They may hinder model performance, possibly due to the delicate nature of EEG signals and the critical importance of maintaining the integrity of the original signal patterns.

4.4.1 Individual Impacts

The introduction of Gaussian noise as a form of data augmentation appears to decrease the performance of all models. This reduction in accuracy might be attributed to the fact that EEG data already contains a certain level of natural noise. The addition of artificial noise could be obscuring genuine EEG signal patterns that the models rely on for making accurate predictions.

The channel drop technique demonstrates a more pronounced negative impact on model accuracy compared to Gaussian noise. Channel drop simulates the loss of information by randomly omitting some channels of EEG data during training. This method may degrade performance significantly because it could be removing critical features that are necessary for the model to make accurate inferences.

4.4.2 Combination of Augmentation Techniques

Interestingly, using both Gaussian noise and channel drop together results in less of a performance decrease compared to using each method individually. This could indicate that the combination of both techniques forces the models to learn more robust features that are less reliant on any single channel or less sensitive to noise, leading to a slight improvement in performance when both types of noise are present.

References

[1] Vernon J Lawhern, Amelia J Solon, Nicholas R Waytowich, Stephen M Gordon, Chou P Hung, and Brent J Lance. Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces. *Journal of Neural Engineering*, 15(5):056013, July 2018. 1

Appendix:

Optimizer

For all models, Adam optimizer is used with a step learning rate scheduler.

Data Augmentations

Two data augmentation techniques are used: Gaussian and channel drop. Gaussian adds a random Gaussian noise tensor scaled by 0.01 to the original data to increase robustness of the model towards variation of the input data. Channel drop randomly zeros out an input channel to prevent the model to be over-reliance on any single channel.

MiniResNet

MiniResNet starts with a conv layer (32-filters of 1x10), followed by batchnorm, ReLU, and a max pooling (1x10 kernel). The architecture progresses through four downsampling residual blocks doubling the number of channels at each block from 32 to 512. Each residual block includes an initial convolution (1x3 kernel, 1x2 stride, 0x1 padding), batchnorm, and ReLU, followed by a second convolution (1x3 kernel, 1x1 stride, 0x1 padding) and batchnorm. The input's residual, processed with the third convolution (1x1 kernel, 1x2 stride, 0 padding) and batchnorm, is added to the second layer's output, leading to a final ReLU. The architecture concludes with a 1x1 adaptive average pooling, and a linear layer reducing dimensions to 4.

EEGNet

This model, inspired by EEGNet, is tailored for EEG signal processing with a CNN architecture. It initiates with a conv layer (32 filters of 1x51, 0x25 padding), followed by batchnorm, ELU, and average pooling (1x5 kernel, 1x5 stride). The sequence continues with a depthwise convolutional layer (32 8x1 filters with no padding), batchnorm, ELU, and the same average pooling. Next, the separable convolutional Layer (64 filters of 1x15 filters and 0x7 padding) expands to 64 channels, followed by batchnorm, ELU, and the same average pooling. Then, the signal is filtered with a 0.6 dropout layer, and compressed via three linear layers with ELU activation, reducing dimensions from 3840 to 1024, then to 512. The process ends with a fourth linear layer that further reduces the features to a size of 4.

RNNModel

This model adopts a classic RNN framework with five RNN layers (input size 22, hidden size 256), utilizing tanh activation. To mitigate overfitting, a 0.6 dropout layer is applied after each RNN layer, excluding the final one. Following the RNN layers, a linear layer reduces RNN output to 128 dimensions, followed by a ReLU activation, then compresses to a 4-dimensional output with a final linear layer.

HybridCNNLSTMModel

This model fuses a CNN architecture with LSTM units, starting with four convolutional blocks. Each block includes a conv layer (5x5 filter, 2 padding). The first block has a max pooling layer (3x1 kernel and stride, 1x0 padding), while the rest has average pooling (2x1 kernel and stride, no padding). All blocks have a batchnorm layer and a 0.6 dropout layer. The convolutional layers increase output channels from 25 to 50, to 100, then 200. After these blocks, the input is flattened and processed by a linear layer reducing dimensions to 40, coupled with an ELU activation. The architecture ends with LSTM, featuring 40 hidden units across 5 layers and a 0.6 dropout rate, then compacts the dimensions to 4 with a linear layer.

EEGLSTMNet

This model merges EEGNet with LSTM units to process temporal dependencies. Initially, input passes through a series EEGNet convolution blocks, then followed by a dropout layer, ELU, and a linear layer. Diverging from solely linear layers used in EEGNet, this model utilized LSTM, incorporating 512 units spread across 5 layers to effectively capture temporal dynamics. A 0.6 dropout is applied to mitigate overfitting, and finally compresses the output to a final 4-dimensional output with a linear layer.

EEGAttentionNet & EEGMultiAttentionNet

EEGAttentionNet integrates an attention mechanism to prioritize critical EEG signal features. Following the EEGNet blocks sequence, the output is flattened to 960 dimensions and divided into keys, queries, and values via three linear layers, each reducing dimensions to 512. Self-attention is performed and the output (4608 dimensions) is then condensed through three linear layers, sequentially narrowing dimensions from 4608 to 512, maintaining at 512, and finally to 4. ELU and a 0.6 dropout are applied after the first two linear layers for effective signal analysis.

Building on the EEGAttentionNet model, EEGMultiAttentionNet replaces the orignal self-attention with a multi-head attention mechanism of 8 heads and an embedding size of 512 for parallel processing of EEG signals.

HybridCNNTransformerModel

This model maintains the initial structure of the HybridCNNLSTMModel's four convolutional blocks and subsequent linear layer with ELU activation, but replacing LSTM units with a Transformer architecture. The Transformer comprises 4 attention heads with an embedding size of 40 for both input and output vectors, and 6 encoder and decoder layers (80-dimensional feedforward netowrk, 0.6 dropout rate, GELU activation). The output is then condensed to 4 dimensions through a final linear layer.

Model	Epochs	Batch Size	Step Size	Gamma	Val Accuracy	Test Accuracy
MiniResNet	30	128	30	0.3	0.2583	0.2506
RNN	40	128	10	0.3	0.2607	0.2257
General CNN + RNN	60	64	30	0.3	0.4645	0.4266
General CNN + Transformer	60	128	20	0.5	0.2607	0.2370
EEGNet	60	16	10	0.3	**0.6967**	**0.7133**
EEGNet + RNN	60	128	20	0.3	0.6730	0.6433
EEGNet + Attention	90	64	30	0.5	0.6232	0.6117
EEGNet + Multi-head Attention	90	128	30	0.5	0.6540	0.6072

Table 1. Accuracy Overview on All Subjects

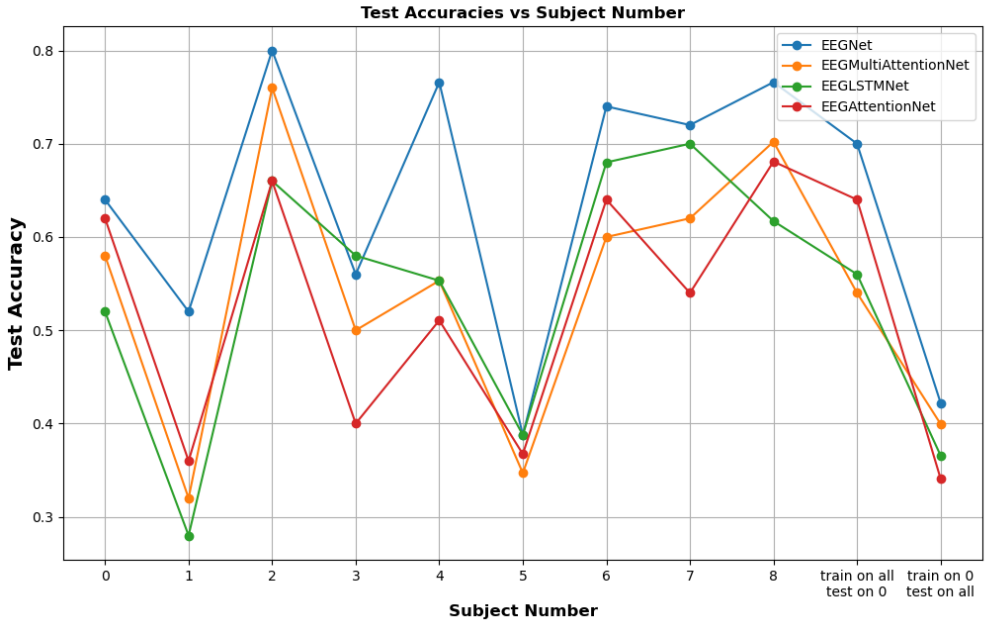

Figure 1. Accuracy comparison on each subject (Exp 2).

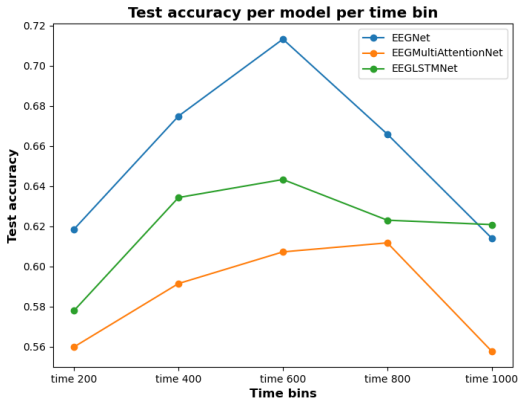

Figure 2. Time Bin Accuracy (Exp 3).

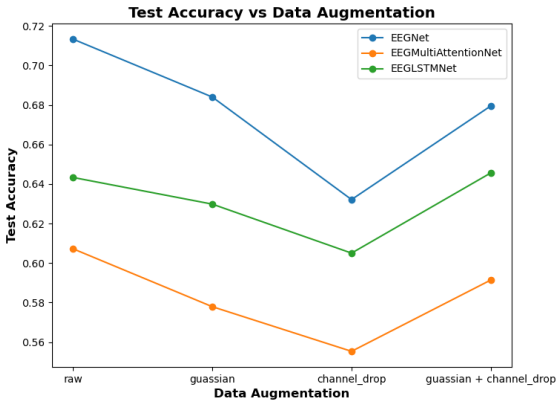

Figure 3. Data Augmentation Accuracy (Exp 4).