ECE C247 Final Project Report

Kuei-Chun Huang
UID: 705668757
kueichun0625@g.ucla.edu

Jo-Hua Wu
UID: 905847065
zoeywu0930@g.ucla.edu

Yi-An Chen
UID: 105945588
anniechen091@g.ucla.edu

Abstract

This paper presents an investigation of various machine learning models for the classification of electroencephalogram (EEG) data. Specifically, three different models, including Convolutional Neural Network (CNN), Convolutional Recurrent Neural Network (CRNN), and Convolutional Recurrent Neural Network with Attention (CRNN with Attention), were implemented, alongside Multi-layer Perceptron (MLP) as the baseline. The models were trained on data from subject 1 and all subjects, and the results showed that the models trained on data from all subjects outperformed those trained on data from subject 1 alone. Additionally, the study examined the impact of time periods on accuracy and found that CNN achieved better results with shorter time lengths, while CRNN and CRNN with Attention were better suited for longer periods. Finally, the study introduced a majority voting ensemble method to enhance the classifiers' performance, resulting in a test accuracy of 0.736 using a total of 64 CRNNs and CNNs.

1. Introduction

We mainly implemented two types of CNN-based models in this project. The first model is a pure CNN model based on the paper [2]. It contains 3 convolutional layers as the feature extractor and a fully connected layer as the classifier. This model architecture reaches great performance, so we did not tune anything about it.

The second model is a combination of CNN and RNN called CRNN. It makes sense to apply the RNN-based model because the input is time series data. We also implemented the network of a paper [1]. However, it turned out that with the same architecture, the accuracy was much worse than that of CNN even though the model was trained to converge. We then tried to make the network deeper and the kernel sizes smaller and found that the modified model performed slightly better than CNN on the validation set. Furthermore, we also introduced an attention mechanism on the output of the RNN layers since RNN with attention usually showed a better performance.

Finally, we elaborated on the ensembling technique in order to investigate the relationship between the performance and the number of models ensembled.

2. Results

This section presents the outcomes of our conducted experiments. Initially, the models were exclusively trained on the data of subject 1, and subsequently, on the data of all subjects. The accuracy of the models was then evaluated with respect to time. Finally, we attempted to improve our results by testing various combinations of model types for ensembling.

2.1. Optimize accuracy for subject 1

To study the accuracy of subject 1 under different training scenarios, We trained the four models - Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), Convolutional Recurrent Neural Network (CRNN), and Convolutional Recurrent Neural Network with Attention (CRNN with Attention) - under two different training sets, data of subject 1 and data of all subjects. Then we evaluated the models' performances on the test set of subject 1.

The models were trained with rigorous precision, each being run 5 times, and the average accuracy was computed to provide a comprehensive overview of their performance. The resulting accuracies are presented in table 1 and 2, which provides a detailed breakdown of each model's accuracy on the validation and test sets.

According to the tables, when we used training data only from subject 1, CRNN exhibited the highest accuracy of 0.488 on the validation set while CNN had the highest accuracy of 0.404 on the test set. However, if we included all subjects in our training data, both CNN and CRNN with Attention demonstrated superior performance on the test set with an accuracy of 0.452.

2.2. Optimize accuracy across all subjects

Furthermore, we extended the analysis to test model performances on data from all subjects. Similar to the approach on subject 1, each model was run 5 times and the average accuracy was computed for each run. The results of this analysis are presented in Table 2, which provides a detailed breakdown of each model's accuracy on the validation and test sets. The analysis reveals that CNN demonstrated the highest accuracies of 0.603 on the validation set and 0.602 on the test set.

2.3. Evaluate accuracy as a function of time

To study the relationship between classification accuracy and time period, we tested different time length settings starting from 300 to 1000 with a step of 50 on our CNN, CRNN, and CRNN with Attention models. The results of validation accuracy and test accuracy are shown in figure 1 and 2 respectively. Based on the result of validation accuracy, the optimal time length for CNN, CRNN, and CRNN with Attention models are 450, 900, and 1000 respectively. We applied these optimal period settings to ensemble models.

2.4. Ensemble

We introduced majority voting as our ensemble method and tested multiple model-type combination settings. The model-type combinations we tried included ensembling by single model type, two model types, and all three model types. For each model type, we applied the optimal time length we mentioned earlier and trained 33 models.

2.4.1 Ensemble by single model type

The result is shown in figure 3, the best accuracy for CNN is 0.688 (with 25 models), the best accuracy for CRNN is 0.661 (with 31 models), and the best accuracy for CRNN with Attention is 0.675 (with 33 models).

2.4.2 Ensemble by two model types

The results are shown in figure 4, 5, and 6. The best accuracy for CRNN with Attention + CRNN is 0.684 (with 56 models), the best accuracy for CRNN with Attention + CNN is 0.731 (with 60 models), and the best accuracy for CRNN + CNN is 0.736 (with 64 models).

2.4.3 Ensemble by all model types

The result is presented in figure 7. With 69 models, it reaches the best accuracy of 0.734.

3. Discussion

In this section, we discussed the impact of the difference in the training data, the performance comparison on different model architectures, and the effect of the ensembling.

3.1. Data of subject 1 vs all subjects

As we can see in table 1 and 2, the accuracy of subject 1 is much lower than that of all subjects. Therefore, we hypothesized that the distribution of data of subject 1 might be more noisy or different from that of other subjects. In addition, except that MLP can not fit the EEG data well, all three CNN-based models' test accuracies of subject 1 increase when training on all subjects. We could thus draw a conclusion that the information from other subjects might be helpful for the prediction of data from subject 1.

3.2. Model comparison

As we can see in table 2, all model types have performance drops on the test set. It means that there might be a bit of difference in the distributions of the train and test set. It is worth noting that CNN reaches the best accuracy on both validation and test sets, which shows that it is the most robust model. Therefore, we assumed that the signals with long time distances might not be critical for models to learn. Instead, an appropriate design of a CNN architecture can be powerful enough to capture most of the essential features. Another interesting finding is that even though CRNN with attention does not outperform CRNN without attention too much on the validation set, it has significantly better accuracy on the test set. We thus concluded that the attention mechanism not only leads to better fitting power but also provides robustness for the CRNN model.

3.3. Different lengths of time

From figure 1 and 2, we observed that the validation accuracy of CNN kept trending down when the time period becomes greater than 500. Similar behavior was also observed in test accuracy. On the other hand, for CRNN and CRNN with Attention, there is no strong correlation between accuracy and time. Still, the accuracy can be slightly improved with longer time periods. We assumed that the reason why CRNN and CRNN with Attention can remain stable accuracy when time length goes up is that the recurrent layers have the advantage to deal with time-related information, which CNN does not.

3.4. Ensemble

As we can see in figure 3, the accuracy of all three types of the model increases as the number of models increases. That is because the randomness during the training process can still produce independent errors even though the model type or the data are the same. This result is consistent with what we have learned from class that the average model error will be reduced with more models ensembled.

Furthermore, we also observed from figures 4, 5, 6, and 7 that with the same number of models, the performance of the model ensembled by more than just one model type beats the model ensembled by only a single model type. That is because there is some difference between the different model types. Therefore, the unique strength of each model type will be aggregated by ensembling. It is also worth noting that both the model ensembled by CRNN and CNN and the model ensembled by CRNN with Attention and CNN are significantly better than the model ensembled by two types of CRNNs. The reason is that the model structures of the two types of CRNNs are much alike compared to CNN. We could thus conclude with the two findings mentioned in this paragraph that the more varied the models are, the more improvement will be with the same number of the models ensembled. This conclusion is also consistent with what we have learned from class.

Finally, another interesting phenomenon we observed is that with the same number of models, ensembling by either type of CRNN with CNN has nearly the same accuracy as by all three model types. That is also contributed by the similarity of CRNN and CRNN with Attention. Hence, if we want to pursue higher accuracy, instead of directly increasing the number of all three model types, we should just pick one of the CRNNs into our ensembling pool, and try to include another model type with a different structure.

References

[1] Soojin Lee et al. A convolutional-recurrent neural network approach to resting-state eeg classification in parkinson's disease. *Journal of Neuroscience Methods*, 361, 2021. 1, 5

[2] Vernon J Lawhern et al. Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces. *Journal of Neural Engineering*, 15, 2018. 1, 5

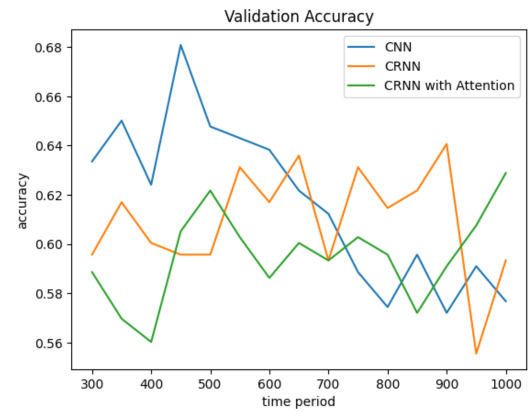

Figure 1. Validation accuracy for the CNN, CRNN, and CRNN with Attention models under different time periods from 300 to 1000.

Table 1. Validation and test accuracy for the MLP, CNN, CRNN, and CRNN with Attention models trained on data of subject 1.

	S1 Valid Accuracy	S1 Test Accuracy
MLP	0.36666	0.32400
CNN	0.41667	**0.40400**
CRNN	**0.48750**	0.35200
CRNN w/ Attention	0.44167	0.37600

Table 2. Validation accuracy and test accuracy for the MLP, CNN, CRNN, and CRNN with Attention models trained on data of all subjects.

	All Valid Accuracy	S1 Test Accuracy	All Test Accuracy
MLP	0.34752	0.28400	0.34086
CNN	**0.60284**	**0.45200**	**0.60226**
CRNN	0.57967	0.44400	0.52235
CRNN w/ Attention	0.58440	**0.45200**	0.56479

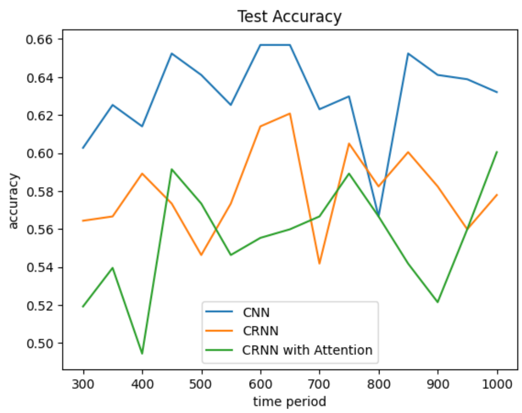

Figure 2. Test accuracy for the CNN, CRNN, and CRNN with Attention models under different time periods from 300 to 1000.

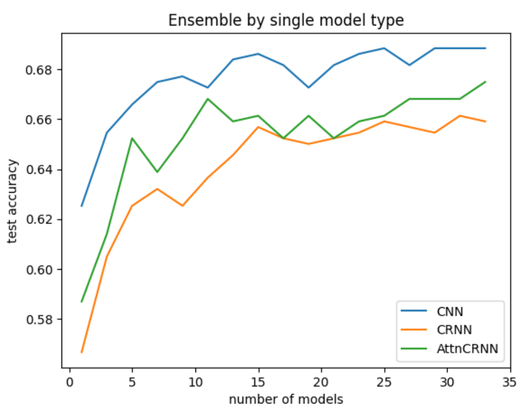

Figure 3. Test accuracy of ensembling by single model type under different numbers of models from 1 to 33.

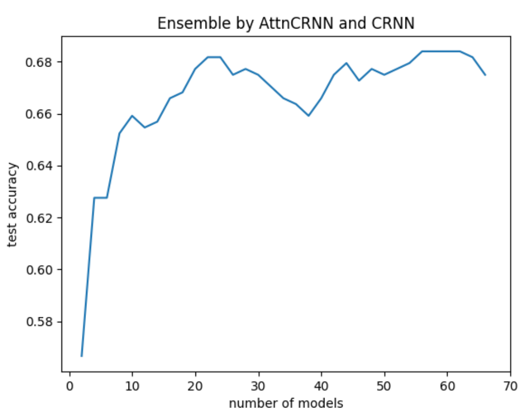

Figure 4. Test accuracy of ensembling by CRNNs with Attention and CRNNs under different numbers of models from 2 to 66.

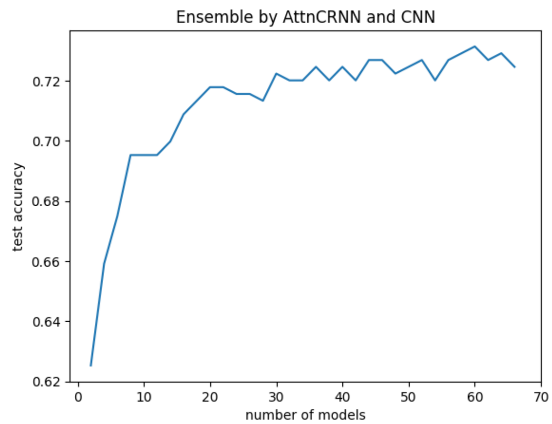

Figure 5. Test accuracy of ensembling by CRNNs with Attention and CNNs under different numbers of models from 2 to 66.

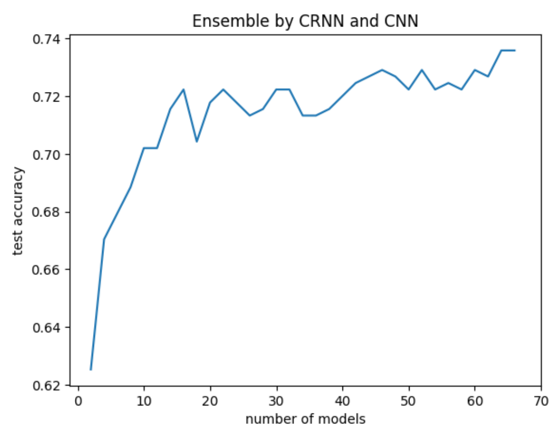

Figure 6. Test accuracy of ensembling by CRNNs and CNNs under different numbers of models from 2 to 66.

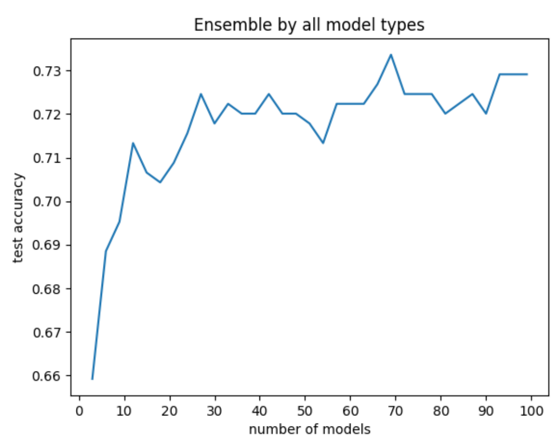

Figure 7. Test accuracy of ensembling by all three model types under different numbers of models from 3 to 99.

4. Appendix

This section shows the architectures of the networks we implemented in this report. In addition, the details of the training process will be illustrated.

4.1. CNN

We implemented the exact model in this paper [2].

4.2. CRNN

We designed a CRNN based on the paper [1]. To achieve better performance, we made the network deeper and kernel sizes smaller. Furthermore, we replaced GRU with Bidirectional LSTM.

Firstly, the model has four convolutional layers. The numbers of output channels are 16, 32, 64, and 128, while the kernel sizes are (1,10), (21,1), (1,10), and (1,10) with stride 1. In addition, all convolutional layers are followed by a batch normalization layer, an ELU activation layer, and a max pooling layer. The kernel sizes of max pooling layers are (1,4), (1,2), (1,2), and (1,2). A dropout layer with a probability of 0.5 is added afterward.

Secondly, the output of convolutional layers will be flattened on channel and height dimensions and fed into a 3-layer Bidirectional LSTM with a hidden size of 128. If not applying attention, the concatenation of the final state of two directions will be taken. Otherwise, the attention-weighted RNN outputs will be further concatenated and projected to the same dimension with a fully connected layer. Also, a dropout layer with a probability of 0.5 is added afterward.

Finally, a fully connected layer with an output dimension of the number of classes acts as the downstream classifier.

4.3. Training Details

We split 20% of the "train_valid" set as the validation set, and trained models on the rest 80% of the data. The number of epochs is 50 and the batch size is 64. We used Adam as our optimizer with a learning rate of 1e-3. Lastly, we chose the model with the best validation accuracy to evaluate on the test set.