Investigating Convolutional and Recurrent Neural Network Performance on classifying motion-related tasks from EEG signals

Atharva Deo
MAE Department, UCLA
deoatharva97@g.ucla.edu

Helene Levy
MAE Department, UCLA
hjlevy@ucla.edu

Marie Payne
MAE Department, UCLA
mpayne6@g.ucla.edu

Abstract

In this project we develop various neural network architectures to make predictions on electroencephalography (EEG) datasets, recordings of neuronal activity by a non-invasive scalp electrodes. The purpose of the networks is to accurately classify which of 4 different actions the subjects are performing. Neural networks are powerful predictors of classification tasks such as these because they are able to identify not only linear features of a dataset, but also spatial and temporal features. We developed both a shallow and deep Convolutional Neural Network (CNN) as well as two recurrent neural networks (RNNs), a Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). In addition, we explored the combination of both RNNs with a CNN. Each architecture was trained and optimized over permutations of the training data. Overall, the best performance came from the bidirectional LSTM with a training and testing accuracy of 93.61% and 70.88%, respectively. Neural networks offer a high degree of flexibility to make predictions on a variety of data, making them powerful computational tools.

1. Introduction

Previous research has shown the ability of CNNs to predict EEG data with a high degree of accuracy [2]. For this dataset, deep networks typically improve predictive performance over shallow networks, however both are limited in they are unable to capture temporal features inherent to the data signal. Therefore, we explored implementing different RNN architectures such as LSTM and GRU because they are well suited for data with time series characteristics. RNNs solve a core issue in CNNs in that RNNs contain recurrent connectivity. CNNs will always produce the same output regardless of the input, however RNNs can record information from past inputs to make a dynamic output prediction. Furthermore, the combination of RNNs and CNNs could potentially extract more useful features than either network implementation on its own.

In the next section, we provide an overview of each network we developed, trends we noticed during training, and network diagrams. We also explore training over one subject vs all subjects on testing and evaluating the classification accuracy over time.

2. Dataset

All EEG data comes from the 2008 BCI competition data [1]. The data was pre-split into training and testing data for all subjects as well as subject-specific training and testing. We randomly split our training data into 80% for training and 20% for validation. The test set was generated independent of training and validation. There are 4 class labels which correspond to moving either the left hand (class 1), right hand (class 2), both feet (class 3), or tongue (class 4).

3. Network Architectures

The following sections detail the seven different network implementations tested on the EEG dataset: Shallow CNN, Deep CNN, LSTM, GRU, unidirectional GRU with CNN, bidirectional LSTM with CNN, and bidirectional GRU with CNN. An explanation for the architectural choices of each network will be provided.

3.1. Shallow CNN

The Shallow CNN structure is based on Schirrmeister et al. , where we merged the temporal and spatial Convolutional layers for optimal training and testing results [2]. Our shallow network has one convolutional layer and hence has less parameters making it faster to train and less computationally expensive. We optimized the network by introducing batch normalization, dropout layers, and L1L2 regularization. We varied these parameters in a range of values and selected the best performing network. Because of the simplicity, our goal was to understand the influences of different variations in the network, and use this information to achieve more accurate results for the following, more complex networks.

3.2. Deep CNN

The Deep CNN architecture includes 4 convolutional blocks, where each block includes a convolutional layer with max-pooling, batch normalization and dropout layers, with the output fed into a fully-connected layer. We also varied the learning rate and the dropout rate to optimize our network performance. Deeper CNN allowed for more features to be extracted from the data, which lead to better accuracy compared to Shallow CNN.

3.3. LSTM

The LSTM is composed of one LSTM layer with batch normalization and dropout. The output was fed through a flatten and dense layer for classification. The LSTM alone was the worst performing out of all the models for testing and training. Possible reasoning includes exploding and vanishing gradients and the LSTM's inability to build memory beyond the length of the subsequence. Hence, we developed the CNN + LSTM architecture.

3.4. CNN + LSTM

The CNN + LSTM includes three convolutional layers to reduce the time and spatial dimensions of the dataset. Each convolutional layer is followed by the ELU activation, batch normalization layer and dropout layer. This was followed by a permute and reshape layer to make the feature vector compatible with LSTM layers. We used two LSTM layers to learn the time information, after which it was sent to a fully-connected layer to generate the four categorical scores. We implemented two types of LSTM on the same architecture: bidirectional and unidirectional LSTM and compared the performances of the two. All together, the CNN + LSTM produced a high degree of accuracy (see Table 1) and takes advantage of the LSTM's ability to incorporate information from past time points.

3.5. GRU

GRU is another type of recurrent neural network (RNN). Similar to LSTM, we found the GRU network performs better in series following a CNN. The GRU network was made up of a GRU layer with batch normalization and dropout paired with a dense classification layer.

3.6. CNN + GRU

Both a unidirectional and a bidirectional CNN + GRU Network were optimized for the EEG data. The architecture is identical to CNN + LSTM with the GRU layers replacing the LSTM layers. The bidirectional GRU, similar to the bidirectional LSTM, utilizes information from past time steps as well as future time steps to make predictions about the current state. Because of this, the bidirectional implementation was able to achieve a higher testing accuracy for all three training and testing trials in Section 4. The architectures for both networks can be found in the appendix.

4. Results

Each network was optimized for all subject data as well as for the first subject. Time series data was also incorporated into the analysis to validate whether training over different time points affects model outcomes.

4.1. Training and Testing on All Subjects

The best results for all networks came from training and testing on all subjects of the EEG data. This was to be expected as with more training and testing data, the model is less likely to overfit. The best model in this scenario is CNN-LSTM, with a test accuracy of 70.88%.

Network	Training	Validation	Testing
Shallow CNN	89.08%	64.15%	68.17%
Deep CNN	89.12%	70.75%	69.97%
LSTM	81.42%	36.40%	36.79%
GRU	74.11%	39.50%	38.60%
CNN + LSTM (Bi)	93.61%	72.17%	70.88%
CNN + LSTM	87.89%	67.92%	68.62%
CNN + GRU (Bi)	89.73%	69.81%	67.27%
CNN + GRU	87.42%	68.87%	67.27%

Table 1. All Subject Training and Testing Accuracy ((Bi) represents bidirectional LSTM / GRU)

4.2. Training and Testing on One Subject

Training and testing on one subject gave lower accuracy than the case of training and testing all subjects. There was a 10-20% decrease in the testing accuracy in comparison to the corresponding networks in Table 1. The best model in this scenario is CNN-GRU, with a test accuracy of 60%.

Network	Training	Validation	Testing
Shallow CNN	95.97%	60.00%	51.99%
Deep CNN	95.48%	55.00%	50.00%
CNN + LSTM (Bi)	96.48%	65.00%	54.00%
CNN + LSTM	96.48%	65.00%	54.00%
CNN + GRU (Bi)	96.98%	70.00%	56.00%
CNN + GRU	96.98%	75.00%	60.00%

Table 2. Subject 1 Training and Testing Accuracy ((Bi) represents bidirectional LSTM / GRU)

4.3. Training on All Subjects, Testing on One Subject

The networks trained on all subjects were less prone to overfitting in comparison to training on one subject. Thus,

this trial had better performance than that of training and testing on one subject. The best model in this scenario is Deep CNN, with a test accuracy of 68%.

Network	Training	Validation	Testing
Shallow CNN	90.02%	66.51%	57.99%
Deep CNN	91.11%	73.11%	68.00%
CNN+LSTM (Bi)	95.03%	69.81%	57.99%
CNN+LSTM	86.62%	67.45%	63.99%
CNN+GRU (Bi)	90.63%	67.92%	64.00%
CNN+GRU	87.84%	69.34%	62.00%

Table 3. All Subject Training and Subject 1 Testing Accuracy ((Bi) represents bidirectional LSTM / GRU)

4.4. Hyperparameter Optimization of CNN

For the Shallow CNN, we built a model providing the flexibility of adding Batch Normalization, L1L2 regularization and dropout layers to the bare shallow CNN. In addition, we varied the learning rates, dropout rates and L1L2 regularization constants to obtain an optimized model. The best model consisted of batch normalization, dropout and L1L2 regularization. This was used as a base for constructing the Deep CNN architecture which also included batch normalization and dropout layers. For Deep CNN, we varied the learning rates and dropout rates to get an optimized model.

4.5. Temporal Evaluation

The classification accuracy for each network was evaluated as a function of time. In general, the classification accuracy increased with data for longer periods of time. However, some fluctuations from the trend can be observed in the 600-700 time points range. It was found that 500-700 time points of data were required to get a reasonable classification accuracy (See Figures 1-4).

5. Discussion and Conclusion

In the case of shallow CNN, the batch normalization, regularization and dropout individually did not help improve the performance significantly, but together, improved the performance by around 10%. Batch normalization improved the architecture by standardizing the statistics of the output of each layer, and dropout and regularization were used to prevent overfitting. This helped us achieve approximately 68% testing accuracy. The insight gained from the Shallow CNN construction was then used to develop a deep CNN architecture.

The deep CNN was able to achieve around 70% testing accuracy by using the findings from the shallow CNN and optimizing the hyperparameters.

The best results came from the deep CNN and the CNN + LSTM. This result is reasonable since the deep CNN takes in information from many nodes in the network and can learn less common features than a shallow network. The LSTM did not perform well on this dataset because of the large number of parameters, however by adding a CNN to reduce the parameters entering the LSTM, a testing accuracy over 70% was realized. The GRU outperformed the LSTM on its own, which was expected since GRUs incorporate less parameters. However, when combined with a CNN the LSTM + CNN combination slightly outperformed the GRU in training and testing. A possible explanation for this might be that the cell states of LSTM layers were beneficial for retaining the important features of the data.

We also provided analysis of training and/or testing on one subject vs. the entire dataset. Across the board, training over all subjects gave the best performance, as training over one subject made the models overfit the data, generalizing poorly on other subjects. We believe that with further fine tuning of the models, their performance can be improved. Pre-processing the data using transformation or subsampling could have improved the accuracy of all the networks.

Some models not explored in this project were data augmentation architectures such as Variational Auto-Encoders (VAEs) and Generative Adversarial Networks (GANs). VAEs and GANs are powerful models that make predictions or alterations to a dataset by generating fictional data. The purpose of one of these networks is less to find a training and testing accuracy but more to explore whether we can make similar class predictions from fictional data. In future work, this data might be interesting to gain insights into what features of an EEG give clues about the ensuing motions.

References

[1] C. Brunner, R. Leeb, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller. Bci competition 2008–graz data set a. *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology*, 16:1–6, 2008.

[2] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball. Deep learning with convolutional neural networks for eeg decoding and visualization. *Human Brain Mapping*, 38(11):5391–5420, 2017.

Appendices

A. Temporal Evaluation of 4 Architectures

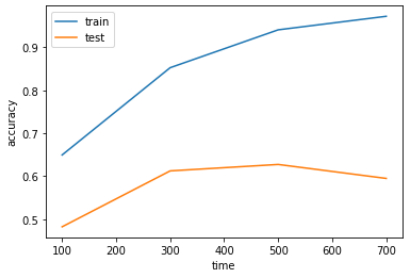

Figure 1. Shallow CNN Temporal Evaluation

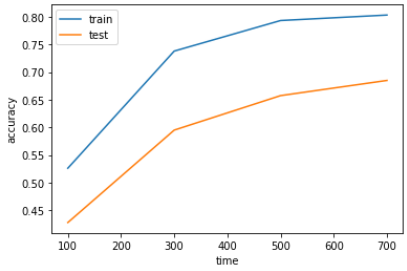

Figure 2. Deep CNN Temporal Evaluation

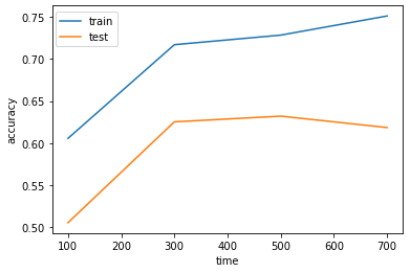

Figure 3. CNN-GRU Bidirectional Temporal Evaluation

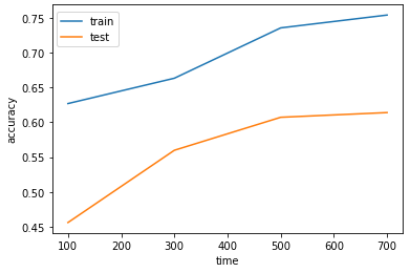

Figure 4. CNN-LSTM Bidirectional Temporal Evaluation

B. Architectures

Shallow CNN:

1. Convolution(filters=22, kernel size=(10,1))
2. Batch Normalization
3. Max Pooling(pool size=(10,1), strides = (10,1))
4. Dropout (0.7)
5. Linear Classification (Dense Layer)

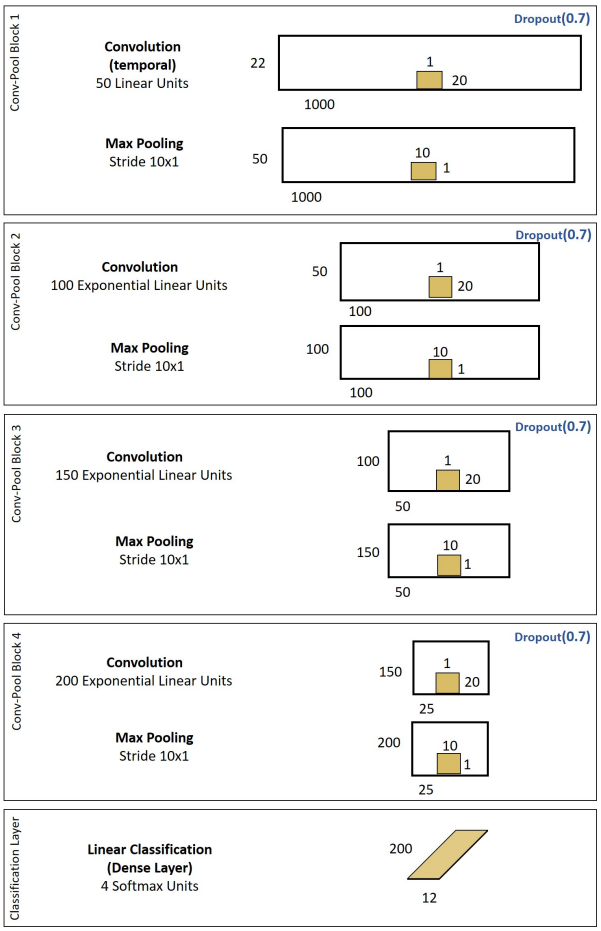

Figure 5. Deep CNN Architecture

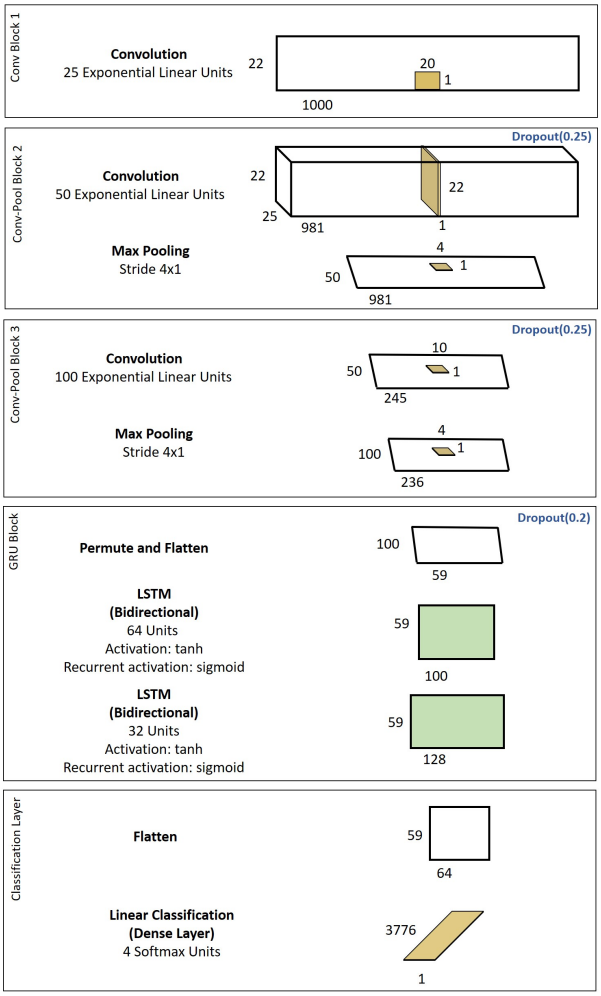

Figure 6. CNN-LSTM Bidirectional Architecture

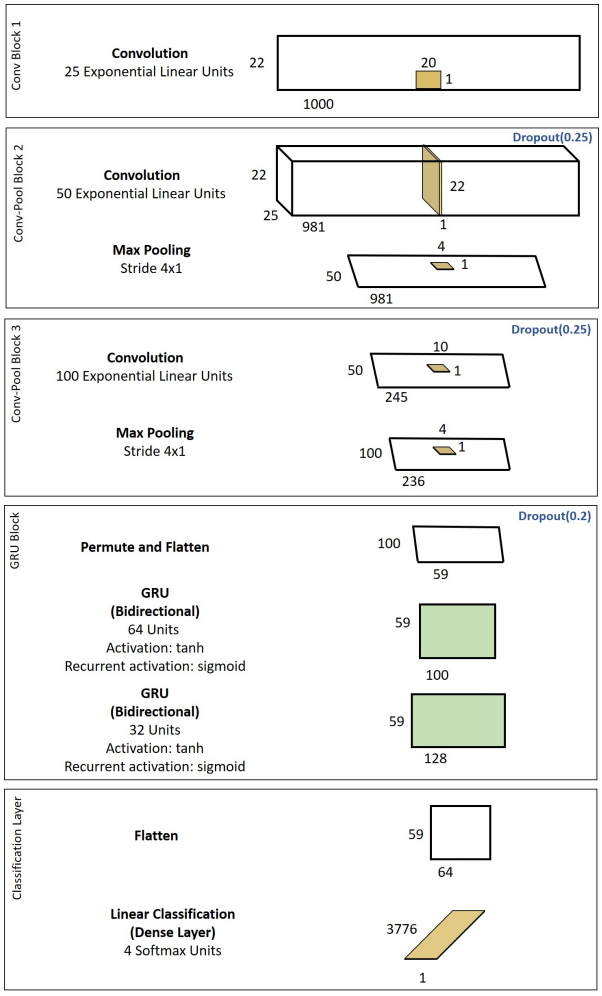

Figure 7. CNN-GRU Bidirectional Architecture