Neural Networks for Movement Classification using EEG data

Ketan Kapre
UCLA
405 Hilgard Avenue, Los Angeles, CA 90095
kkapre10@ucla.edu

Abstract

Neural networks are a growing technique in machine learning and have a wide range of applications. This paper explores and compares the ability of convolutional neural networks (CNNs), recurrent neural networks (RNNs), and convolutional recurrent neural networks (CRNNs) to classify distinct imagined movement types using EEG data. The best performing network was a CNN with spatial and temporal filters in separate layers and achieved an accuracy of 70.2% on the test data. A temporal convolutional layer was found to perform better than a long short-term memory (LSTM) layer at learning temporal features. It is hypothesized that the importance of the frequency power spectrum for EEG data could explain this. The best performing network was able to classify individual subjects using their EEG data with 99% accuracy. This suggests individuals have distinct EEG signatures. This distinctness could also explain the finding that models trained on one subject were unable to generalize movement predictions to other subjects with more than 40% accuracy. The best performing network was able to maintain greater than 70% accuracy even when it was provided only 1.2 seconds of data instead of 4 seconds. This suggests the network is learning robust features within the data and that brain waves may contain significant redundancy in their representation of distinct imagined movements. Overall, the results help inform the design of neural networks for EEG data, provide some information about EEG data characteristics, and may be useful in applications like brain-computer interfaces (BCI).

1. Introduction

The main goal of this project was to compare the performance of different neural network architectures at decoding movement actions from EEG data. There were four main architectures compared: two CNNs, an RNN, and a CRNN. Some comparisons on variations within each type were made. All architectures were created through the API provided by tensorflow-keras. All models had a comparable number of trainable parameters in the range of $400,000$ to $500,000$ if 1000 time bins were available.

A simple CNN architecture was created as a baseline and based off an architecture used to classify images from the CIFAR-10 dataset. It will be referred to as Image CNN in the rest of the paper. This architecture is of the form [(conv - relu - pool - dropout - batch norm) x 2 - dense - softmax] and can be viewed in more detail in the Architectures section. The EEG data is treated analogous to 22x1000x1 image as it gets passed through three convolutional layers with filter size 3x3.

To improve performance, a CNN that was more adapted to EEG data was created using the paper Schirrmeister et al. as a guide [1]. It will be referred to as EEG CNN in the rest of the paper. This CNN consisted of a temporal convolution layer, spatial convolution layer, pooling layer, dropout layer, batch normalization layer, and dense layer with softmax activation for classification. The layer orders permuted and tested for impact on performance

An recurrent network architecture (RNN) was created with one LSTM layer and a dense layer with softmax activation for classification. An RNN with two LSTM layers instead of one was also compared with.

A convolutional recurrent network (CRNN) was also created. It was designed to be comparable with the EEG CNN. The first neuronal layer uses a spatial convolutions like the EEG CNN. The second neuronal layer is an LSTM layer to learn temporal features. This layer can be compared with the temporal convolution layer of the EEG CNN.

Each model was trained for 20 epochs with a learning rate of $1e{-}3$. The model was then trained further for 20 epochs with a learning rate of $1e{-}4$. This was compared with running the model for 40 epochs with learning of rate of $1e{-}3$.

Dataset augmentation through upsampling to increase the number of training examples was also tested.

Models were trained and tested on data using all the time bins from all subjects in the first experiment. The best performing model was also tested using data with less time bins available to determine when performance declined signifi-

cantly.

In the second experiment, models were trained on data from subject 1 and tested on data from subject 1 and the remaining subjects separately.

In the third experiment, the best performing overall model was also trained on the subject IDs to determine whether the model could identify a person based on their EEG data.

Hyperparameters and architecture details were optimized and finalized on the validation dataset. After optimization, accuracy values were obtained from the test data. Since batch selection introduces randomness to accuracy values, the accuracy for any particular model type was taken from an average of test accuracy results for 4 trained models of any particular type.

2. Results

The models in order of performance were the EEG CNN, CRNN, RNN, and the Image CNN. Refer to Figure 1 for an accuracy comparison between all models. The best test accuracy was 70.2%. This EEG CNN was found to have lower test accuracy if the first convolution layer had a relu activation function. Accuracy was also found to be higher if the spatial convolution layer came before the temporal convolution layer.

The EEG CNN was able to perform with greater than 70% accuracy even with about a third of the time bins. Performance only dropped below 60% when less than 200 time bins were used. Accuracy was actually about 2% higher when 500 time bins were used instead of 1000. View Figure 2 for a visualization of the results.

No model trained only on data from subject 1 was able to achieve greater than 60% accuracy on data from subject 1. Performance dropped from the all subject models by 2%-10%. All models also had lower than 40% accuracy when they were trained on data from subject 1 and tested on data from other subjects. View Figure 1 for a detailed comparison.

The EEG CNN had an accuracy of 99% when identifying a subject off their EEG data. It achieved an 96% accuracy for subject identification with only 100 time bins of data.

Dataset augmentation with upsampling was not found to significantly improve validation accuracy.

The best learning rate schedule was found to be running for 20 epochs, annealing the learning rate to $1e{-4}$, and training for another 20 epochs. This learning rate adjustment was found to have a benefit of several percent on validation accuracy.

Models with three convolution layers were attempted but not found to improve performance .

An RNN with two LSTM layers performed worse than an RNN with one LSTM layer.

Using a gated recurrent unit (GRU) layer instead of an LSTM layer was not found to increase performance.

Adjusting batch size did not significantly impact validation accuracy.

The RNN was unable to learn if the LSTM layer only returned the last state. The layer used instead sent the full sequence to the dense layer, specifically a tensor of shape (batch size, number of time steps, number of units)

SpatialDropout2D was found to perform worse than regular dropout.

Using a relu activation after both EEG CNN layers led to overtraining and lower performance when compared with the EEG CNN with relu activation only after the second convolutional layer.

Performance was improved in EEG CNN when the spatial filter layer came before the temporal filter layer.

Using 100 filters in the first convolutional layer and 200 filters in the second convolutional layer instead of 50 and 100 respectively gave worse performance for the EEG CNN.

3. Discussion

Since the EEG CNN performed better than the CRNN, the temporal convolution layer appears to be more effective at learning temporal patterns than the LSTM. This could be due the fact that key information is likely contained in the band power spectrum of EEG data. The temporal convolutional layer learns 100 filters that emphasize different frequency bands of this power spectrum and passes that information on to the rest of the layers. It is more difficult to analyze the LSTM layers output, but it appears to be less effective at extracting relevant temporal features for this dataset when compared with the convolutional layer.

This analysis could also explain why the RNN did not perform as well as the EEG CNN and why the EEG CNN performed better than the Image CNN. The Image CNN may not be learning the frequency spectrum as effectively since its uses both spatial and temporal convolutions in each layer and has a small kernel size.

The CRNN did perform better than the RNN. This is likely because the extra spatial convolution layer allowed for the learning of spatial features which were then passed on to the LSTM layer.

The ability of the EEG CNN to classify movements with greater than 70% accuracy using only 300 time bins or 1.2 seconds of data suggests that the network is learning robust patterns and that brain waves may have significant redundancy in their representation of movements. The robustness and ability of the network to work on short time intervals suggests it could have potential BCI applications.

Interestingly, using 500 time bins gave slightly better performance than using 1000 time bins. This could be due to the curse of dimensionality. More specifically, due to

the limited number of training examples it may be more difficult to learn the patterns within the higher dimensional space of 1000 time bins than the space with 500 time bins.

The high accuracy of the EEG CNN at identifying subjects from EEG recordings suggests people have very distinct EEG signatures. It could be interesting to explore what features the network has learned to better understand what characteristics make people's brain waves so distinct.

This distinctness could also explain why a network trained on one subject had low accuracy when evaluated on other subjects.

In sum, these results show that understanding the intrinsic features of data (EEG in this case) and tailoring neural network architectures to these features can help optimize model performance. The results also suggest that convolutional layers may be better adapted than LSTM layers for use with EEG data and perhaps with other data types that have important information stored in the frequency spectrum. The ability of the EEG CNN to also identify subjects suggests people have very distinct brain waves and that the network architecture may be generalizable to a variety of classification tasks. The model shows robustness by being able to maintain an accuracy of 70% with about 1.2 seconds of data instead of the full 4 seconds. With some accuracy improvement, a BCI system that determines an imagined movement could use this output to act on the world or send a communication for those who normally cannot easily do so. Further extensions could be to create a deeper network that is well adapted to EEG data, visualize network weights and learned features, and test the networks ability to classify different characteristics like emotions.

References

[1] 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, Aug. 2017.

Performance Summary

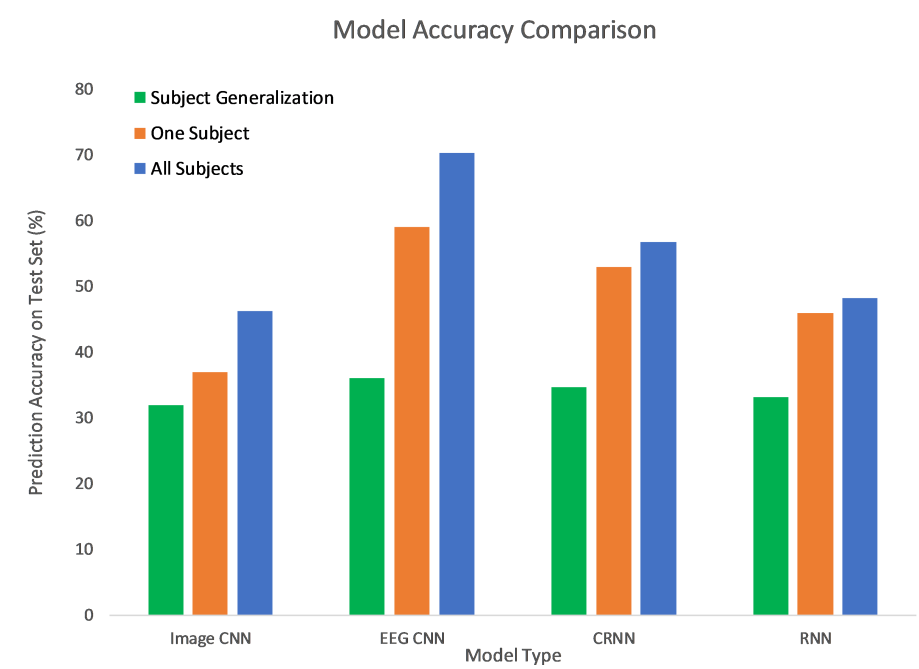

Figure 1: Plot of Model Accuracies. Blue bar represents accuracy for model trained on all subjects and tested on all subjects. Orange bar represents accuracy for model trained on one subject and tested on the same subject. Green bar represents accuracy for model trained on one subject and tested on the 8 other subjects

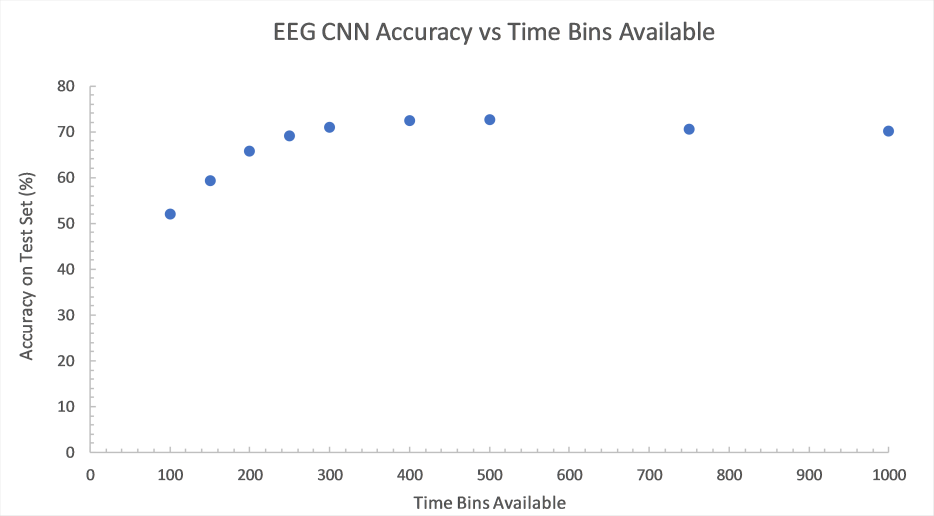

Figure 2: EEG CNN Accuracy vs Time Bins Available. Shows accuracy only begins to drop below 70% when less than 300 time bins are available. Accuracy dips below 60% when less than 200 time bins are available

Architectures

All Models

All models used the Adam optimizer. Models were trained for 20 epochs with an initial learning rate of 1e−3 and for another 20 epochs with an initial learning rate of 1e−4. The batch size was 64 and the validation data was a 10% split from the train_valid data provided. Hyperparameters were optimized on this validation data. Accuracy values were found after architecture optimization by evaluating the model on the test set.

All final layers consisted of a dense layer with the number of neurons equal to the number of classes and a softmax activation. Labels were transformed from class numbers to one-hot encodings. The loss function was categorical cross entropy.

Every layer with neurons uses L2 norm regularization with a default regularization value of 1e−3. If dropout layers were used, the dropout rate was a default of 0.5.

EEG CNN

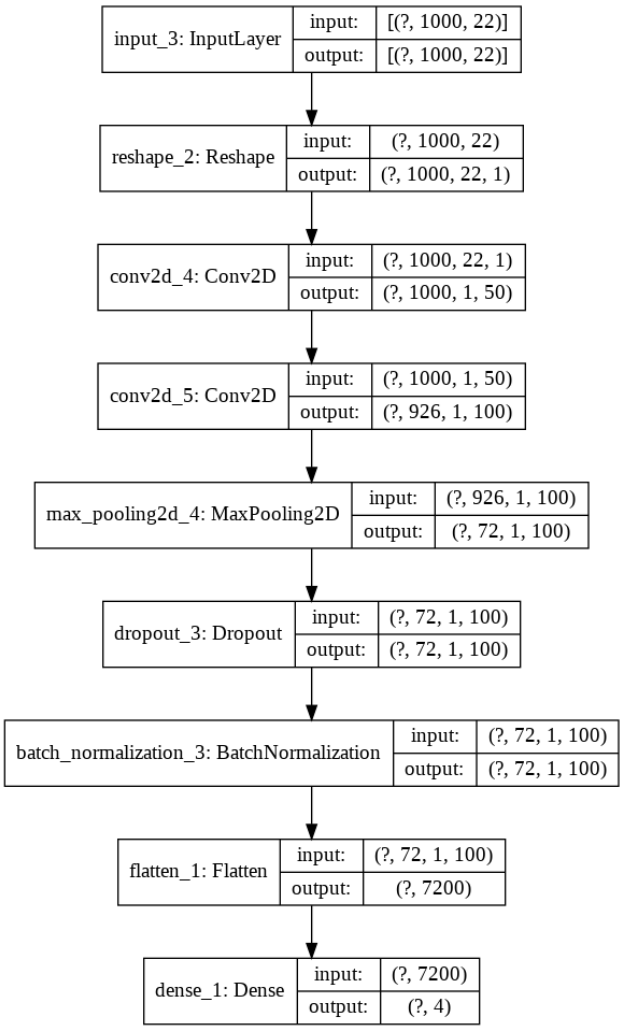

Figure 3: Flow Diagram EEG CNN. Its architecture was guided by the paper Schirrmeister et al [1]. The question marks represent the batch size which can be adjusted. The reshape layer adds an extra dimension to allow the input to be sent through the two convolutional layers that follow it.
The first of these has 50 spatial convolution filters with a kernel size of (1, number of channels) or (1, 22) for this data. It has no activation function. The next layer has 100 temporal convolution filters with a kernel size of (75, 1) that was treated as a hyperparameter. It has a relu activation function. The next layer is a max pooling layer with a size of (70, 1) and strides of (12, 1). The next two layers are a dropout layer and a batch normalization layer to provide regularization. Finally, the output is flattened and sent to the dense layer.

Image CNN

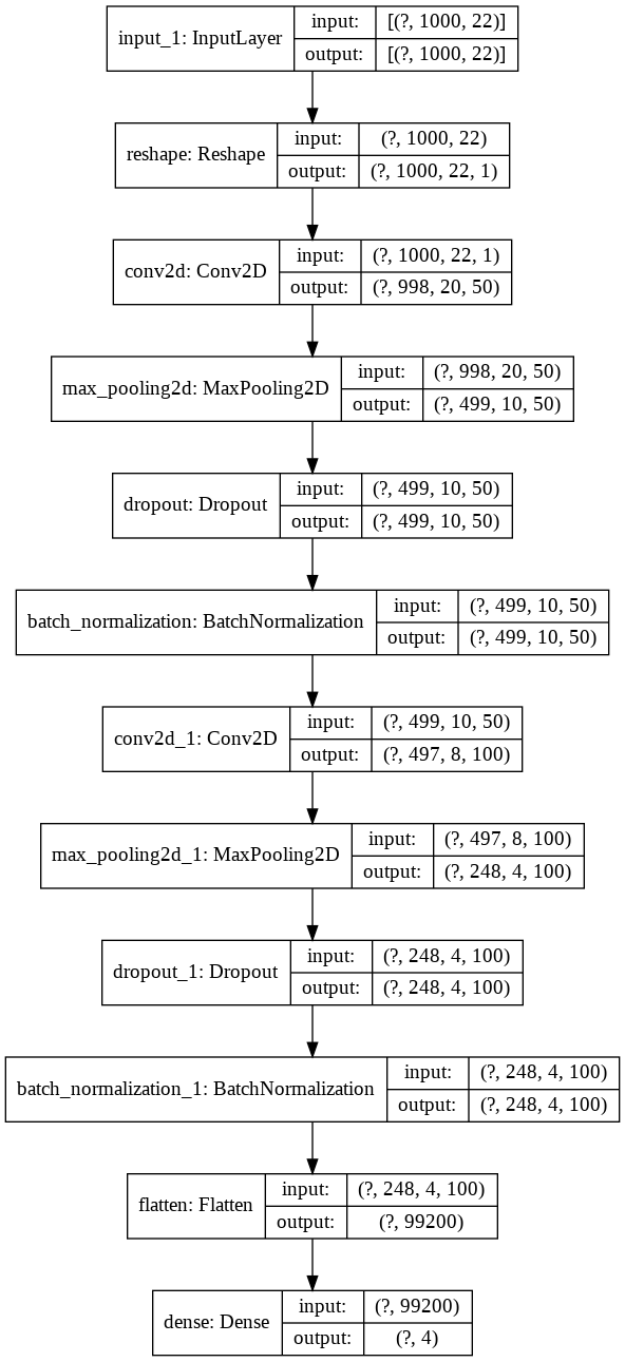

Figure 4: Flow Diagram Image CNN. The reshape layer adds a dimension to send the input through the convolutional layers. All the convolutional filters are followed by relu activation functions. The first block consists of a convolutional layer, max pooling layer, dropout layer, and batch normalization layer. The convolutional layer has a kernel size of 3x3. The max pooling layer has shape (2, 2) with a stride of (2, 2). The next block is the same as the first block but has 100 convolutional layers instead of 50. The output is flattened and sent to the dense layer.

RNN

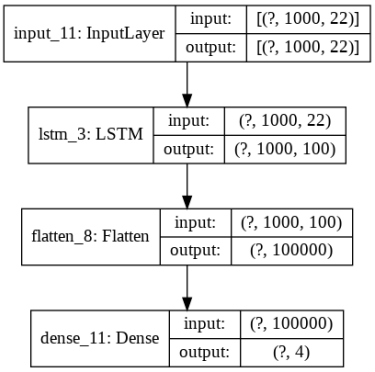

Figure 5: Flow Diagram RNN. The input is sent directly to an LSTM layer with 100 units. The LSTM layer is set to return sequences and not just the final state. An input dropout of 0.5 was incorporated within the LSTM layer. The output is flattened and sent to the dense layer.

CRNN

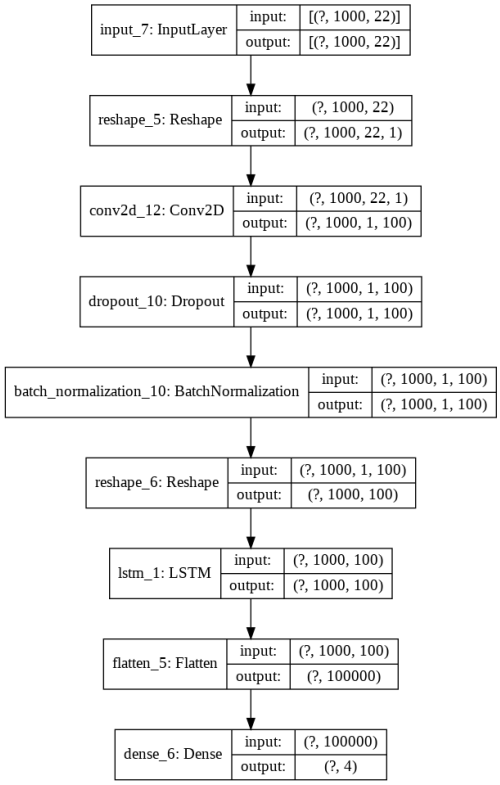

Figure 6: Flow Diagram CRNN. The reshape layer adds an extra dimension to allow the input to be sent through the two that come next. The layers that follows is a spatial convolutional layer that is analogous to the one used in the EEG CNN. The kernel size is also 22 but the number of filters is 100. Dropout and batch normalization layers follow this layer. The tensor is then reshaped to remove dimension of size 1. This forms an array of shape (number of time bins, number of filters) which gets sent to the LSTM layer. The LSTM layer has 100 units. An input dropout of 0.5 was incorporated within the LSTM layer. Its output gets flattened and sent to the dense layer.