# Implementing CNN and RNN Deep Neural Network architectures for EEG Data Classification

Radhika Nayar
16 March, 2020

## Abstract

*The aim of this project is to evaluate the effectiveness of Convolutional Neural Networks (CNN) alongside Recurrent Neural Network (RNN) such as Long Short Term Memory (LSTM) in the classification of the electroencephalograph (EEG) data provided in Brain-Computer Interaction (BIC) competition. The EEG data can be classified into four different motor imaginary tasks such as movement of the left hand (class 1), right hand (class 2), both feet (class 3), and tongue (class 4). The goal is to optimize the performance of decoding a particular task with high accuracy. The architectures used consists of multiple layers of convolutional layers and both the architectures are trained on varying data to correctly classify the task. Techniques of regularization such as batch normalization and dropout are used to improve the generalization accuracy of the models. This project compares the performance of both models, CNN and RNN, by optimizing the classification accuracy across all subjects and a single subject. The classification accuracy is also evaluated as a function of time for both models.*

## 1. Introduction

To tackle the task of classification on EEG data, both the Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) architectures have been compared and evaluated. For both models, I have compared their classification accuracy across all subjects, a single subject and as a function of time.

### 1.1. Convolutional Neural Network (CNN)

The architecture for the Convolution Neural Network (CNN) is shown in Figure 1. This CNN architecture was modeled closer to the LeNet-5 but with the addition of extra convolutional layers. The LeNet architecture follows [CONV-POOL]x2 - CONV - FC - OUT. However, taking inspiration from the more layered architectures like ResNet, which significantly improved the accuracy compared to LeNet, the number of stacked convolutional layers with ReLU activation function were increased to four, followed by a batch normalization layer, a 2D max pooling layer and a dropout layer. Thus making the CNN architecture as [CONV-NORM-POOL-DROPOUT]x4 - FC - OUT. The first three layers of the convolution perform temporal and spatial convolution aiming to reduce the time dimension. The last layer aims to reduce the feature dimension. The additional batch normalization and dropout layers were added to improve the efficiency of the architecture. Finally the fully connected layer outputs the scores for the softmax layer as a one-hot vector used for classification purposes. This architecture also closely resembles the deep ConvNet architecture proposed by Schirrmeister et al [2] to tackle the task of EEG decoding.

### 1.2. Recurrent Neural Network (RNN)

The key missing feature of Convolutional Neural Network (CNN) is that they lack recurrent connectivity and hence have no dynamics. Recurrent Neural Networks (RNN) have recurrent connectivity that define a dynamical system that governs how it evolves through time. The architecture of the implemented Recurrent Neural Network (RNN) is shown in Figure 2. To be able to compare the performance between the two architectures, I used a similar setup as the CNN with four stacked convolutional layers using ReLU activation function, followed by a batch normalization layer and a 2D max pooling layer. To avoid missing important information, the dropout layer was only added after the last convolution layer and after the last bidirectional Long Short Term Memory (LSTM) layer. The convolutions are still performed over the time dimension because features from adjacent time steps are highly correlated and the increased length in time dimension makes it difficult to learn data for RNN. Following the four convolution layers, there are three bidirectional LSTM layers. The use of bidirectional recurrent structure maintains recurrent connections going forward and backwards in time. Since this is a time series dataset, I hypothesize for the optimal prediction of a label at a certain time step, it will be beneficial to have data along the time dimension from both the past and future and thus bidirectional RNN should make a good choice as a model. LSTM is a particular RNN architecture that is

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 22, 996, 16)       96
_________________________________________________________________
batch_normalization (BatchNo (None, 22, 996, 16)       88
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 22, 498, 16)       0
_________________________________________________________________
dropout (Dropout)            (None, 22, 498, 16)       0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 22, 494, 32)       2592
_________________________________________________________________
batch_normalization_1 (Batch (None, 22, 494, 32)       88
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 22, 247, 32)       0
_________________________________________________________________
dropout_1 (Dropout)          (None, 22, 247, 32)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 22, 243, 64)       10304
_________________________________________________________________
batch_normalization_2 (Batch (None, 22, 243, 64)       88
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 22, 121, 64)       0
_________________________________________________________________
dropout_2 (Dropout)          (None, 22, 121, 64)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 2, 121, 128)       172160
_________________________________________________________________
batch_normalization_3 (Batch (None, 2, 121, 128)       8
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 2, 30, 128)        0
_________________________________________________________________
dropout_3 (Dropout)          (None, 2, 30, 128)        0
_________________________________________________________________
flatten (Flatten)            (None, 7680)              0
_________________________________________________________________
dense (Dense)                (None, 4)                 30724
=================================================================
Total params: 216,148
Trainable params: 216,012
Non-trainable params: 136
_________________________________________________________________
```

Figure 1. Architecture of Convolutional Neural Network (CNN)

well-suited for addressing the problem of vanishing and exploding gradient making this model a more attractive choice compared to the CNN architecture. Note, again the fully connected layer is the last layer with softmax activation to get the one-hot vector for classification purposes.

## 2. Dataset

The data collected from electroencephalography (EEG) is from the Brain-Computer competition data (2008) [1]. For each subject, data is recorded using 22 EEG electrodes while the user imagines performing on the four different motor imagery tasks, namely the imagination of the movement of the left hand (class 1), right hand (class 2), both feet (class 3), and tongue (class 4)) making this a classification task. The dataset consists of EEG data collected from 9 subjects. Two sessions on different days were recorded for each subject. Each run consists of 48 trials (i.e. 12 trials for each of the four possible classes), yielding a total of 288 trials per session. There are total of 2115 trials; each trial has corresponding EEG data from 22 electrodes over 1000 time bins (i.e. the data is presented as 22 features changing across 1000 timesteps).

## 3. Results

Comparison of the classification accuracy across all subjects for both CNN and RNN is shown in Table 1.

The classification accuracy of each of the nine subjects for both CNN and RNN is shown in Table 2 and Table 3 respectively.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_76 (Conv2D)           (None, 22, 991, 16)       176
_________________________________________________________________
batch_normalization_76 (Batc (None, 22, 991, 16)       88
_________________________________________________________________
conv2d_77 (Conv2D)           (None, 2, 991, 32)        10784
_________________________________________________________________
batch_normalization_77 (Batc (None, 2, 991, 32)        8
_________________________________________________________________
max_pooling2d_76 (MaxPooling (None, 2, 247, 32)        0
_________________________________________________________________
conv2d_78 (Conv2D)           (None, 2, 238, 64)        20544
_________________________________________________________________
batch_normalization_78 (Batc (None, 2, 238, 64)        8
_________________________________________________________________
max_pooling2d_77 (MaxPooling (None, 2, 59, 64)         0
_________________________________________________________________
conv2d_79 (Conv2D)           (None, 2, 50, 128)        82048
_________________________________________________________________
batch_normalization_79 (Batc (None, 2, 50, 128)        8
_________________________________________________________________
max_pooling2d_78 (MaxPooling (None, 2, 12, 128)        0
_________________________________________________________________
dropout_76 (Dropout)         (None, 2, 12, 128)        0
_________________________________________________________________
permute (Permute)            (None, 12, 128, 2)        0
_________________________________________________________________
time_distributed (TimeDistri (None, 12, 256)           0
_________________________________________________________________
bidirectional (Bidirectional (None, 12, 256)           395264
_________________________________________________________________
bidirectional_1 (Bidirection (None, 12, 128)           164864
_________________________________________________________________
bidirectional_2 (Bidirection (None, 64)                41472
_________________________________________________________________
dropout_77 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_20 (Dense)             (None, 4)                 260
=================================================================
Total params: 715,524
Trainable params: 715,468
Non-trainable params: 56
_________________________________________________________________
```

Figure 2. Architecture of Bidirectional Recurrent Neural Network (RNN)

| Model | Accuracy |
|---|---|
| CNN | 49 |
| RNN | 60 |

Table 1. Classification accuracy across all subjects for CNN and RNN with epoch=40

| SubjectID | Train/Test split | Accuracy |
|---|---|---|
| 0 | 237/50 | 36 |
| 1 | 236/50 | 34 |
| 2 | 236/50 | 60 |
| 3 | 234/50 | 30 |
| 4 | 235/47 | 45 |
| 5 | 236/49 | 37 |
| 6 | 238/50 | 54 |
| 7 | 232/50 | 46 |
| 8 | 231/47 | 55 |

Table 2. Classification accuracy of each subject for CNN

Classification accuracy over a range of time period from 100 to 1000 with a timestep of 100 is plotted for both CNN and RNN and shown in Figure 3 and Figure 4 respectively.

The best accuracy at the optimal time duration across all subjects for both CNN and RNN is shown in Table 4.

| SubjectID | Train/Test split | Accuracy |
|---|---|---|
| 0 | 237/50 | 64 |
| 1 | 236/50 | 30 |
| 2 | 236/50 | 60 |
| 3 | 234/50 | 36 |
| 4 | 235/47 | 43 |
| 5 | 236/49 | 37 |
| 6 | 238/50 | 56 |
| 7 | 232/50 | 56 |
| 8 | 231/47 | 66 |

Table 3. Classification accuracy of each subject for RNN

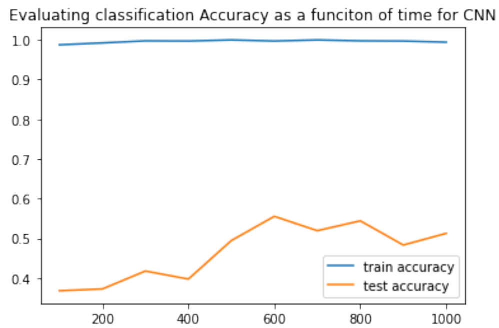


Figure 3. Classification accuracy as a function of time for CNN

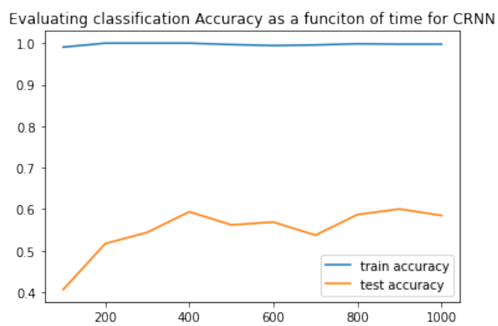


Figure 4. Classification Accuracy as a function of time for RNN

| Model | Best Accuracy | Optimal Time |
|---|---|---|
| CNN | 55 | 600 |
| RNN | 60 | 900 |

Table 4. Best Classification Accuracy of CNN and RNN at certain time period

## 4. Discussion

In this project, I evaluate the CNN architecture and RNN architecture across all subjects, individual subject and over the entire time period. The classification accuracy across all subjects indicate that the RNN model performs 11% better at classification compared to the CNN model as shown in Table 1. This validates my hypothesis and proves that the LSTM RNN architecture works better with the time series dataset. However, comparing the accuracy across all subjects versus individual subject, it was found that some subjects can reach the same or better accuracy whereas other subjects cannot reach such a high accuracy. This is because for a single subject dataset, a much simpler model is required with steps to prevent overfitting such as higher dropout probability and regularization.

Plotting the classification accuracy as function of time indicates that the best accuracy for CNN occurs at a much faster time duration compared to the RNN model as shown in Table 4. Since the same Adam optimizer is used in both models, the longer training time is due to the complexity of RNN compared to the CNN. The three LSTM layers in RNN model requires significantly more time to train and find the optimal classification accuracy.

The general trend of classification accuracy as a function of time is similar in both CNN and RNN models. As the data is trained over longer period of time, the test accuracy increases. After time duration of around 500, the test accuracy seems to stabilize for both the models. Though the test accuracy stabilizes for both at around same time, they have best accuracy at different times. For CNN, the best test accuracy is 55% at time 600 whereas for RNN, the best test accuracy is 60% at time 900.

The models can be further improved using grid-search to find the optimal values of different hyperparameters. Perhaps both models did not need to be trained for 40 epochs to find the model accuracy across all subjects. As observed in Figure 3 and Figure 4, the training accuracy is close to 100% while the testing accuracy is much lower for both models. Different dropout probabilities could be experimented to check whether the classification accuracy improves or drops. Higher dropout probability prevents overfitting but can run the risk of eliminating important information such that the model underfits. L1 or L2 regularization could have been applied in each of the convolutional layers to prevent overfitting. Also, different model architectures both shallow and more deep and different ordering of layers could have been explored to reflect the effects on classification accuracy.

## References

[1] G. M.-P. A. S. C. Brunner, R. Leeb and G. Pfurtscheller. Bci competition 2008–graz data set. 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 brain mapping and decoding of movement-related information from the human EEG? *CoRR*, abs/1703.05051, 2017.