Applications of Convolutional Neural Networks and Recurrent Neural Networks in EEG Analysis

Jianwei Zhang, Xiaopei Zhang, Yutong Lu
Department of Electrical and Computer Engineering
University of California, Los Angeles
z18686634810@gmail.com
zxpmirror1994@gmail.com
amylyt@ucla.edu

Abstract

Deep learning (DL) has demonstrated great capability of classifying images and videos. Recently there has been a growing interest in applying deep learning technologies for medical data analysis. In this paper, we aim at classifying the EEG data related to human's neural activities. We mainly implemented three DNN structures, including a deep convolutional neural network (CNN) architecture, a shallow CNN architecture, and a convolutional gated recurrent neural network, to process the EEG data. We also explore different activation functions, optimizers and regularization and pooling strategies that may assist in improving both the training speed and the accuracy. In general, our CNN architectures can be optimized to achieve around 68% accuracy, and our RNN architecture can achieve over 55% accuracy for a single subject, and 45% for all.

1. Introduction

Before implementing DL structures and testing their performances, we cleaned the data and only used 22 electrodes. The first model we have implemented is a deep CNN consisting of five convolution layers, three pooling layers and one fully-connected (FC) layer. In addition, we also added batch normalization after each convolution layer and one dropout layer to prevent over-fitting. The second one is a shallow CNN with only two convolution layers, one pooling layer and one FC layer. We also tried to combine CNN and RNN together, which is formed as C-RNN with five (to classify one subject) or seven (to classify all subjects) convolution layers, four GRU layers and one FC layer with one batch norm layer as the first layer. Exact structures for these three architectures are shown as Fig 1, 2, 3.

As for the pooling layers, there are two choices: max pooling or mean pooling and we tested the classification performance for both choices. To train these networks, different activations can be applied, and in our setup, we trained our models using ReLU, tanh, sigmoid and ELU. In addition, we tested on all nine EEG datasets as well as tested one classifier across all subjects. Also, based on different percentage of partition of the dataset to form testing dataset and training dataset, the testing accuracies also vary.

2. Results

2.1. Deep CNN and Shallow CNN

The result for deep convolution net is shown in Fig 4 and for shallow convolution net is shown in Fig 5. We trained the model for 500 epochs, as we can see in the plots (Fig 4 and 5). Despite some oscillation on validation accuracy, theses plot show increases in both train and validation accuracy. The final test accuracy on a separate test set is 0.6831 for deep convolution net and 0.5613 for shallow convolution net. These results are trained on the whole dataset on all 9 subject data. We have also tried to train the model on one subject data. We found that these models were not suitable for one subject since with only 288 trails in the data, CNN models easily over-fit. We also used different time steps to train the model (we used 1000,800,500,200 time steps). The results show that 200 time steps model underperforms. We will analyze these problems and results in the discussion section.

2.2. RNN

In recent years, recurrent neural networks have gained more popularity in dealing with sequential and temporal data. Given the input as a time series of signals and the task of classification, an immediate approach is to stack multiple robust RNN layers, such as GRU or LSTM that avoids

vanishing/exploding gradients, followed by a softmax classifier. However, the testing results of the RNN stack on the EEG data is bad, achieving only 20-30% accuracy, which is similar to random guessing.

2.3. Convolutional Gated RNN (C-RNN)

In fact, simple stacking of RNN layers has two obvious drawbacks to handle relatively long time-series data. First of all, it is computationally expensive due to the complexity of RNN layers as well as the large dimension of input data. Second, without pre-processing the input data, RNN layers may overlook the dependencies hidden in local pieces of data. However, these difficulties can be simultaneously overcome by inserting a stack of Conv1D layers prior to RNN layers. These extra convolutional layers can not only shrink the size of feature maps, but also extract local information about temporal dependencies. Following Conv1D layers, RNN layers are capable of capturing both long- and short-term dependencies of sequential data [1].

Fig 3 displays the architecture of our convolutional gated recurrent neural network specially designed for the EEG data. Here, we choose GRU rather than LSTM, but there are actually no significant differences between them. The result of our C-RNN model on Subject 1 is shown in Fig 6 and the result on all subjects is shown in Fig 7. We see that it achieves around 55% validation accuracy for Subject 1, but only 45% validation accuracy for all subjects. The inconsistent performance of our model can be partially explained by the variation of neural activities coming from different subjects. In Section 3, we delve in optimization measures that are effective for the classification task of either a single subject or all subjects.

3. Discussion

3.1. Single Subject Optimization

3.1.1 CNN

For deep Conv net, we have tried two training strategies. First one, we used data from a single subject (288 trials). Second one, we combine all the data together and train across all the subjects. We found that for training on one subject, the model overfits easily (train accuracy increases to 1 while validation and test accuracy remain at around 0.3). Yet, when we train across all the data, the model can achieve accuracy around 0.65. We believe this is due to two reasons. First, 288 trails compose a very small dataset which makes over-fitting easily. Second, since training on all dataset has better performance, the model is not learning features that is general enough for classification task when trained on only one subject. We also test these two strategies on shallow Conv net. Different from the deep net, shallow net has a consistent accuracy on one subject and all subject. This makes sense since the shallow net structure is modified from Filter bank common spatial patterns (FBCSP). This structure utilizes concepts from traditional signal processing method. Therefore, its consistent performance reflects the concepts behinds it since it is learning a certain pattern in data. In comparison between two models, the best test accuracy achieved by deep Conv net is around 0.65 and 0.55 for shallow net. Cross all datasets, deep net has better performance, yet in terms of consistency, shallow net has a rather stable performance on all data and one subject data.

3.1.2 C-RNN

For the convolutional gated RNN, there are several network settings and hyper-parameters that we can play around to improve its performance. First, we can vary the number of Conv1D layers, With each Conv1D layer having a filter size of 4×1 and a stride of 2, the size of feature vectors is halved after every passing of a Conv1D layer. If we include too many Conv1D layers, the size of the output of the last Conv1D layer, i.e. the input to the GRU stack, will decrease exponentially, thus impairing the strength of GRU layers. Based on our computations and experiments, we eventually decide to have five Conv1D layers, each with 32 different filters, so that the input to the GRU stack has a size of 32×32. This is an appropriate setting that results in great performance on the data of Subject 1, and guarantees a relatively fast training process.

Second, we can adjust the activation function that our GRU layers use. In early RNN structures, the problem of vanishing/exploding gradients is very common. If we use the sigmoid or the tanh which are prone to saturation, the gradient anomaly will be getting worse. But this does not bother GRU or LSTM at all. Conversely, the output by the sigmoid or the tanh has special interpretations - it can tell how much a gate is activated or how much a previous state is preserved. By default, the activation function used by the GRU layer in Keras is the tanh. According to our experiments, even though using ReLU makes the loss converge the fastest, tanh gives the best prediction accuracy.

3.2. All Subjects Optimization

3.2.1 CNN

In the end, the best accuracy achieved is by deep convolution net at around 0.65 across all data. We started from a clean structure with only convolution layers, max pooling layers, and FC layers. This first model over fits extensively. Then, we add Batch norm, ELU activation and dropout layers. We used ELU here mainly because ELU, unlike RELU, allows negative number to pass through, which given the signal nature of the data should be more fit. We used a

dropout layer at 0.7 to prevent model from over fitting. With the addition of these layers, the validation accuracy rises to around 0.4. Then, when tuning these parameters, we found that the size of filter on the temporal axis had a significant impact on the final test accuracy. The initial filter size on temporal (time) axis is 5, which in comparison to 1000 time steps, is rather small. We increased the size to 30 and the performance reached 0.65. We also had a similar discovery in RNN and C-RNN (to be discussed below).

We hereby speculate that increasing filter size on temporal axis increases accuracy could indicate that the classification of signals depends on the local correlation of temporal axis. The signal in a certain window together might encode some necessary information to classify the data. As the filter size on temporal axis reaches 50, the performance starts decreasing. This further supports the theory that data in a certain time window encodes classification information and the window size effectively for good performance is around 30.

For one subject data, deep Conv net is heavily over fitting. We have tried to tune the size of filter, dropout rate and activation function, but the model keeps on over fitting. we believe that this is due to the fact that one subject has only 288 data points and it is a really small size for a 5-layer deep Conv net. Another reason could be that the model is not learning information necessary for classification since one subject and small data size lacks generality for model learning.

For shallow net, we follow the same path of adding batch norm, ELU activation and dropout. The best test accuracy it achieves is around 0.55. The shallow model's performance is also increasing to 0.55 by changing the filter size on temporal axis to 30. However, unlike deep net, shallow net has a consistent performance on one subject data at 0.55. As the structure of shallow one is analogous to the structure of Filter bank common spatial patterns (FBCSP). This structure is designed to learn a certain pattern similar in signal processing, therefore, its consistent behavior is expected.

3.2.2 C-RNN

To optimize for all subjects, the first update applied upon the C-RNN model that we design for single subject classification is adding two more Conv1D layers before the GRU stack. If so, by the end of the Conv1D stack, the size of feature maps is going to be reduced to around 8×32. This strategy compresses the temporal information more densely, and may counteract the variation among subjects. It raises the validation accuracy from 25-30% to 40%.

Furthermore, unlike CNN structures, we exclude most of Batch Normalization layers except the one used for input normalization for our C-RNN model, because we do not introduce any nonlinearity during the convolution operations (Conv1D).

For C-RNN, we include dropout and recurrent dropout in every GRU layer to minimize the effect of over-fitting on the small EEG dataset. Based on our experiments, the proper values of dropout and recurrent dropout are 0.2 and 0.5, respectively.

3.3. Classification Accuracy as a Function of Time

For all CNN and C-RNN, using longer times series will surely increase the time of training. The classification accuracy of CNNs will increase as the models are learning more local features and will automatically assign weights to each time point. If it believes some time point is relatively important for making a decision, it will assign more weights to this time point.

For the C-RNN model, we alter the number of Conv1D layers to maintain the size of the input to the GRU stack. Since each of our Conv1D layers reduce the dimension of feature vectors by 2, if the input is shortened by half, we just get rid of one Conv1D layer. We see from the results in Table 1 that the classification accuracy increases as we use data over long periods of time.

3.4. Insight from CNN and C-RNN structures

When classifying the given EEG data, it is found that CNN could give a better classification accuracy than RNN generally. CNN is proved to have great prediction after learning the features of of training data since it is finding local features. We can imagine the 22×1000 data for each trial as an image, and based on the filter size we have chosen for each convolution layer, we believe that there exists some relationships between each electrode. For C-RNN structures, among all experiments, testing accuracies are not as satisfactory as CNN's and if we just use RNN without convolution layers in advance, the testing results will be even worse. We therefore suspect that maybe there are too many noises in the time-series data that many of them are irrelevant for classifying purpose. Prediction based on a lot of noise will just make RNN be at a loss. This is why we convolute the training data several times before feeding them to GRU that we want to firstly extract some useful information and reduce the length of time series.

Except the above major observations, other minor observations such as how to split the dataset into training, validation and testing are also available. For cross-validation, we set split ratio to be 0.1 and each time we sample different size of trials for testing and it is found that with the increase of size of test set, the classification accuracy will go down.

References

[1] S. Roy, I. Kiral-Kornek, and S. Harrer. Chrononet: A deep recurrent neural network for abnormal eeg identification. *CoRR*, abs/1802.00308, 2018.

Appendix

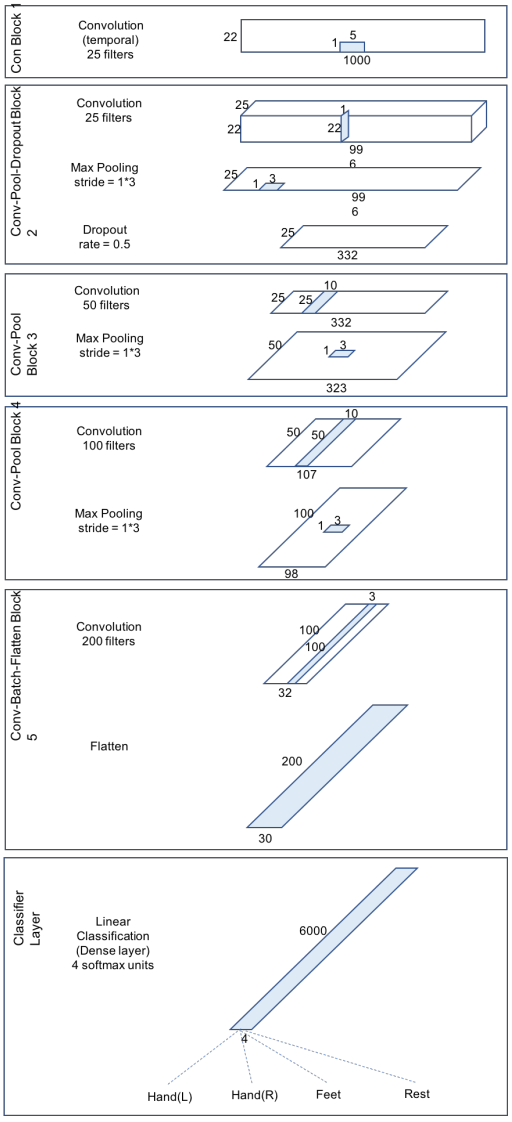

Figure 1. Architecture of deep CNN

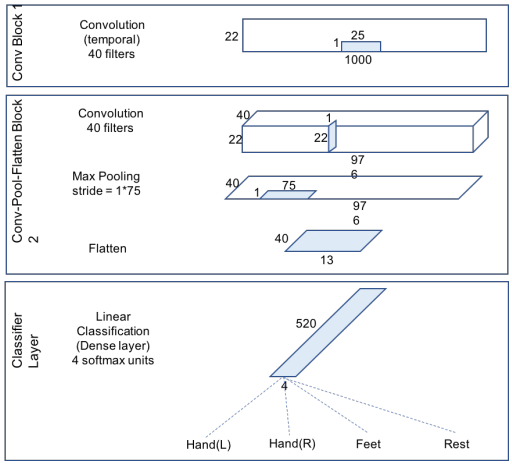

Figure 2. Architecture of shallow CNN

	CNN	SCNN	C-RNN
$t = 250$	54.24%	43.13%	34.35%
$t = 500$	60.11%	55.24%	40.71%
$t = 1000$	68.31%	55.25%	43.89%

Table 1. Performance comparison of the three deep neural networks described in Section 2 under various durations (=[1,t]) of EEG data of all subjects.

	CNN	SCNN	C-RNN
one subject	31.14%	54.13%	55.28%
all subjects	68.31%	55.25%	43.89%

Table 2. Performance comparison of the three deep neural networks described using EEG data of one subject and all subjects.

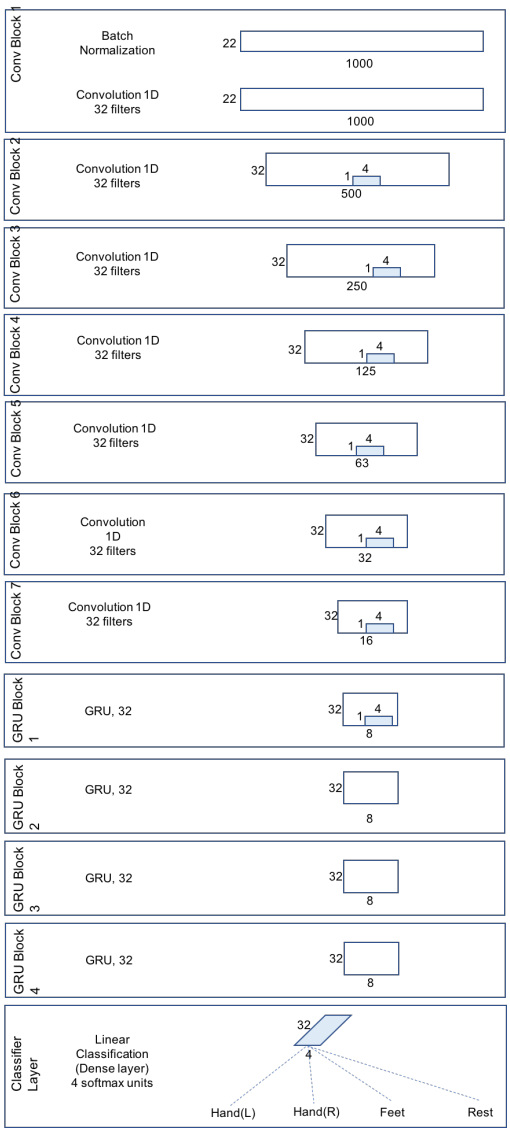

Figure 3. Architecture of C-RNN for all subjects

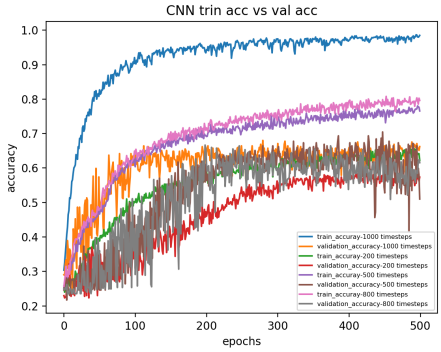

Figure 4. The plot shows the training accuracy against validation accuracy. The plot grows as the epoch increases.

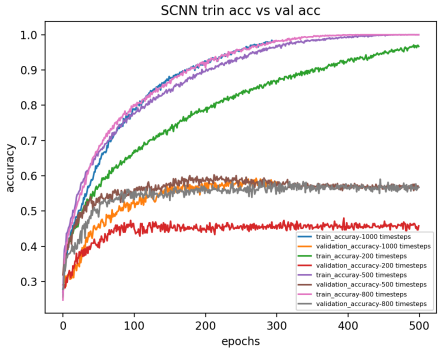

Figure 5. The plot shows the training accuracy against validation accuracy. The plot grows as the epoch increases.

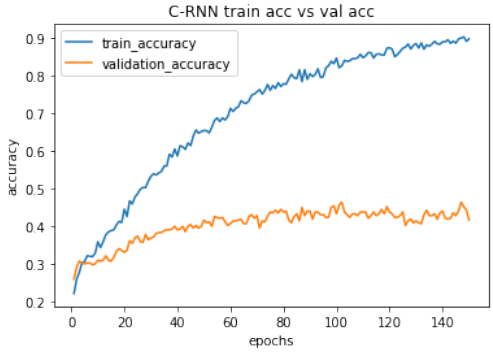

Figure 6. The plot displays the training accuracy and the validation accuracy versus the training epoch, on all subjects.

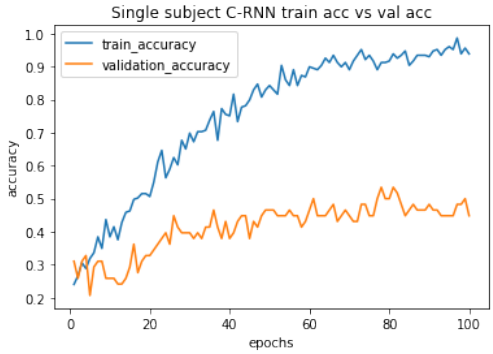

Figure 7. The plot displays the training accuracy and the validation accuracy versus the training epoch, on only Subject 1.