EEG Decoding with Convolutional Neural Network (CNN) and Convolutional Recurrent Neural Network (CRNN)

Zhengxiang Yi UID: 304761570
University of California, Los Angeles
roveryi@g.ucla.edu

Bing-Chen Yang UID: 805427866
University of California, Los Angeles
victorbcyang96@g.ucla.edu

Abstract

The main purpose of this project is to classify the electroencephalograph (EEG) data into four different motor imaginary tasks, movement of left-hand, right-hand, feet, and tongue. The EEG data's time duration varies, so the variation is considered. Two structures used in this classification are Convolutional Neural Network (CNN) and Convolutional Recurrent Neural Network (CRNN). This project shows the comparison of the performances of these two structures.

1. Introduction

1.1. Dataset overview

EGG dataset contains recorded activities of neurons near a non-invasive scalp electrode. Two sets, training-validation and testing set are provided for the project. The data from 4 different classes are plotted as grey color map in Fig 1 with data in each horizontal axis is data from one of 22 electrodes and data in each vertical axis is all 22 electrodes recorded data at one time step. As observed from Fig 1, from the perspective of time-series data, the data recorded at each time step is strongly correlated the data in the previous steps such that dark and light bands form in the images. This observation can help to inform the model structure. Another point needed to be iterated is all target labels are of almost equal number of samples, so no further consideration regarding unbalanced classification is required.

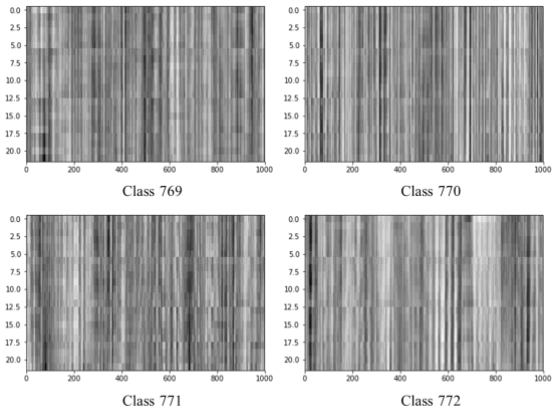

Figure 1: Image of EGG data for 4 difference classes

1.2. CNN Structure

The input data is from multiple trials of 22 electrodes over 1000 time-bins. Signals of one time-bin from one of the 22 electrodes can be analogized as the RGB channels of images such that 1-dimensional convolutional manipulation can be conducted to the raw data. Consequently, a CNN structure could be a reasonable starting point to investigate.

To obtain a reasonable model, the training set is split into 80% for training and 20% for validation set. For each investigated model, cross validation is performed to select the best hyperparameters. By comparing the validation accuracies of different models, the best model is determined with the highest validation accuracy. The testing accuracy is used for determining the classification ability of CNN based models for EGG dataset.

There are three main parts of modeling CNN models. First of all, a baseline model is developed. Baseline model means the model can basically handle the multiclass classification task, and it will be used for checking whether optimizing for subject 1 would be helpful across all subjects or not. Then, to train a better model with more powerful classification accuracy, further modifications would be made on the baseline model and compare different model performances. The investigated models are listed in Appendix A and the detailing of training process and model performance can also be found in Appendix A. The basic motivation of investigating these structures is checking it is deeper or shallower CNN structure would help the classification and whether the introduction of residual block would help improving the accuracy. At last, the best model is selected based on highest validation accuracy, and the best model is also used for evaluating the training time impacts on the model performance.

1.3. CRNN Structure

The correlation between adjacent time steps features are high, and the dimension of time is large. Therefore, it is reasonable to apply convolution on time-dimension to address the difficulty of Recurrent Neural Network (RNN) to learn in such condition.

Several structures of Convolutional Recurrent Neural Network (CRNN), with different number of Long Short Term Memory (LSTM) layers, are implemented in this

project. The testing accuracy is also used for determining the classification ability.

The structure of CRNN, including four convolutional layers, followed by three LSTM layers, a dropout layer, and a fully connected layer, provides the highest accuracy. Between the LSTM layers and fully connected layer, a dropout layer is developed for avoiding absent information. As the structure capture information along time for training, one can expect a higher accuracy than CNN.

2. Results

2.1. CNN Results

A 4 convolutional layer model is selected as baseline model to investigate the impact on the overall accuracy of training on a single subject. To train the model with only one subject, different weight tensors are applied when forming the loss function. Different weights from 1.0 to 0.25 are assigned to subject 1 and other subjects share the same weight to make the total weights sum up to 1. Fig 3 presents the training and testing accuracy when different weighs are put on subject 1. As can be observed from Fig 3, when all weights put on subject 1 in the loss function, training the model does not help the overall accuracy, because the loss function only considers misclassification of subject 1. In this case, even all subject 1 label is correctly classified, the model still performs like random guess (25% accuracy). As the weight on subject 1 decrease, the model performance increases. When the weight subject 1 reaches 0.7 (weights on other subject are 0.1), the model performance is stable, which means even higher weight is assigned to subject 1 training also helps the accuracy over other subjects. This is easily to be interpreted since during this stage, the loss function also takes misclassification from other subjects into consideration even the weights are low.

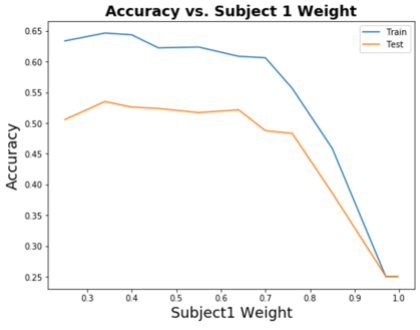

Figure 3: Model Accuracy vs. Subject 1 Weight

The baseline model is of validation and testing accuracy 47% and 42%, respectively. There seems exist improvement in terms of the accuracy. Both complicated and simpler CNN structures are investigated to find a better model. The investigations include adding and reducing the number of convolutional layers, increasing the number of fully connected layers and introducing residual blocks in deeper CNN structures. During the try and error process, it has been found that deeper CNN structure cannot help improving the accuracy. As the number of convolutional layers increase to 5 layers, the model performs closely to random guess, and overfitting is observed. Also, increasing the number of fully connected layers does not help improve the model performance given the same number of convolutional layers. The model accuracy increases as the number of convolutional layers reduces and reaches to a peak of 65% to 70% testing accuracy at 2 convolutional layers. However, when there is only 1 convolutional layer, the model also does not work. There are multiple reasons contribute to the observation. First, the date set is not suitable for very deep convolutional layers since the original data in one channel is of shape [1,1000], and after several convolution manipulation the dimension could be even lower than the kernel size. Consequently, deep convolutional layers could lead to large information loss. This judgement can be furtherly enhanced by the introduction of residual block. When two residual blocks are adopted in the 5 convolutional layer network, it performs closely to the baseline model, which contains 3 convolutional layers. Residual block helps passing the original information to deeper network such that avoiding information loss. Secondly, fully connected layer cannot contribute to the prediction of original data. This can be proved by simple neural network structure consisted by only fully connected layers. Thirdly, using 80% of data for training only gives around 1,700 individuals, which is not sufficient for training a deep CNN. This could be the reason leads to overfitting in the 5 convolutional layer model. Data augmentation could be used to increase the training set size (but not adopted here). Lastly, from the perspective of time-series data, the original data at each time step is strongly correlated to the data in the previous steps as observed from Fig 1. Convolution computation considers this correlation by adopting kernels to localize the data. Consequently, two convolutional layers with one fully connected layer perfectly models the correlation and lead to the best model performance.

Using the selected best model (2 layers CNN), a simple experiment is conducted to investigate the training time's impacts on the model accuracy. The model accuracy against number of epochs for the proposed model is presented in Fig 4. As the training time increases, the training, validation and testing accuracy increase. In the beginning, the improvement in the accuracy increases sharply as longer training time. After about 200 epochs, validation and testing accuracy reach a plateau and training accuracy still keeps increasing. Training a greater number of epochs can help improve the model performance especially given the training set size is limited. However, this may lead to overfitting. Another thing to be noted is the CNN with residual blocks (does not shown in the plot), the

introduction of residual block speeds up the training process significantly. Almost after 5 epochs the training loss reduces to zero. For such models, training time does not have an observable impact on the model performance.

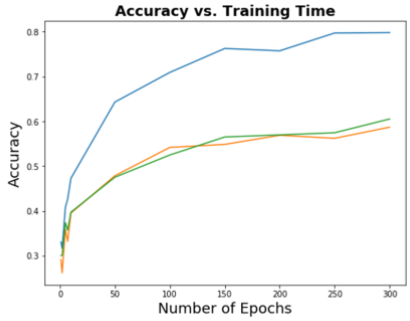

Figure 4: Model Accuracy vs. Training Time

2.2. CRNN Results

Several CRNN models are used for training on the whole data set.

Frist of all, different numbers of LSTM layers are tried in this project. The validation accuracy and testing accuracy of applying four layers of LSTM is not better than three layers is used on same number of epochs setting (epoch = 40). Table 1. and Table 2. show the training accuracies and testing accuracies of different numbers of layers of LSTM. The CRNN model with three LSTM layers has the highest resting accuracy, so this structure is selected for the later investigation. It makes sense that CRNN model with four LSTM layers has lower testing accuracy than CRNN structure with three LSTM layers, since structure with more LSTM layers requires more complex calculation, i.e., more epochs are needed.

Table 1. Accuracies of Different Numbers of LSTM Layers

	One Layer of LSTM	Two Layers of LSTM
Validation Accuracy	96.2%	94.9%
Testing Accuracy	65.9%	64.8%

Table 2. Accuracies of Different Numbers of LSTM Layers

	Three Layers of LSTM	Four Layers of LSTM
Validation Accuracy	92.3%	82.3%
Testing Accuracy	66.4%	58.0%

CRNN model contains three LSTM layers is applied for training on data of nine single subjects and the whole data set. The result of testing accuracy of single subject shows in Figure 5. As can be seen in the result, accuracies of different subjects are different, some have the higher accuracy than the accuracy of whole data set, while some have lower accuracies. The reason of this variation of accuracies of different subjects is that the amount of information single subject can provide is too small for CRNN for accurate classification, i.e., CRNN model is overfitting for single subject data. This overfitting problem can be addressed by applying regularization on single subject data set and adding more "dropout layers."

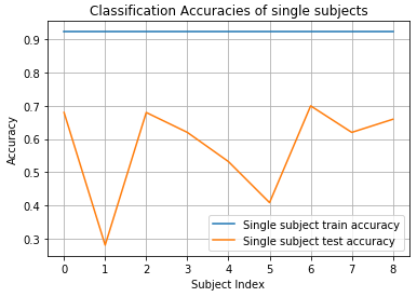

Figure 5: Accuracy of Single Subject

The best CRNN model is also used for investigating data over different time periods. In Figure 6., the testing accuracies in the beginning of time are low, and the accuracies are getting higher as time duration increases. The testing accuracies are low in the beginning of time, since time duration is small at the beginning, the amount of information is not adequate for training. This is why CRNN model with fewer convolution layers is applied when time duration is smaller than 300. As can be seen in Figure 6., after time duration around 400, the testing accuracies are higher than 60%. Due to the amount of features the model can learn increases, the CRNN model can classify the data more accurately. When time duration is 1000, the model reach its highest testing accuracy of 92.340428%

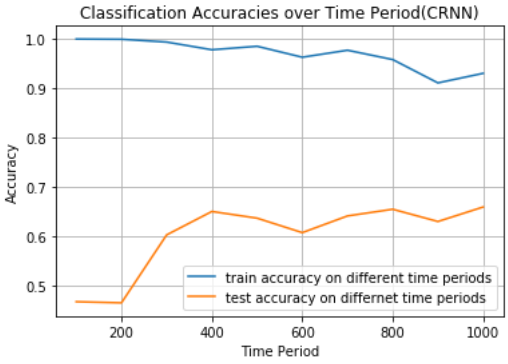

Figure 6. Testing Accuracies Among Different Time Period

References

[1]Brunner, C, et al. BCI Competition 2008 Graz Data Set A

APPENDIX A. CNN Structures and Performances

Table A1. Investigated Models

Model	No. of Layers	No. of Conv Layers	No. of FC Layers	No. of Parameters	Optimizer	Learning Rate	Momentum	Weight Decay	Validation Acc	Testing Acc
Baseline	5	4	1	2272196	SGD	0.0008	0.9	0.12	47%	42%
2-CNN	3	2	1	701572	SGD	0.0001	0.9	0.1	60%	59%
5-CNN	6	5	1	4888516	SGD	0.0008	0.9	0.1	27%	35%
5-CNN-2	7	5	2	6991812	SGD	0.0008	0.9	0.1	29%	27%
5-ResNet	6	5	1	4169704	SGD	0.0008	0.9	0.5	48%	44%

Table A2. Model Details

Model	Baseline	2-CNN	5-CNN	5-CNN-2	5-ResNet
Layer1	Conv1d(22, 64, kernel = 3)	Conv1d(22, 128, kernel = 7)	Conv1d(22, 64, kernel = 3)	Conv1d(22, 64, kernel = 3)	Conv1d(22, 64, kernel = 3)
	BatchNorm1d	BatchNorm1d	BatchNorm1d	BatchNorm1d	BatchNorm1d
	Relu	Relu	Relu	Relu	Relu
	Dropout(0.6)	Dropout(0.7)	Dropout(0.6)	Dropout(0.6)	
	MaxPool1d(3)	AvgPool1d(3)	MaxPool1d(3)	MaxPool1d(3)	
Layer2	Conv1d(64, 128, kernel = 7)	Conv1d(128, 512, kernel = 7)	Conv1d(64, 128, kernel = 7)	Conv1d(64, 128, kernel = 7)	Conv1d(64, 256, kernel = 3)
	BatchNorm1d	Dropout(0.7)	BatchNorm1d	BatchNorm1d	BatchNorm1d
	Relu	Relu	Relu	Relu	Relu
	Dropout(0.6)	BatchNorm1d	Dropout(0.6)	Dropout(0.6)	Conv1d(256,256, kernel=3, padding =2)
	MaxPool1d(3)	AvgPool1d(3)	MaxPool1d(3)	MaxPool1d(3)	BatchNorm1d + Residual
Layer3	Conv1d(128, 256, kernel = 7)	Linear(55296,4)	Conv1d(128, 256, kernel = 7)	Conv1d(128, 256, kernel = 7)	Conv1d(256, 1024, kernel = 3)
	BatchNorm1d	NA	BatchNorm1d	BatchNorm1d	BatchNorm1d
	Relu		Relu	Relu	Relu
	Dropout(0.6)		Dropout(0.6)	Dropout(0.6)	Conv1d(1024,1024, kernel=3, padding =2)
	MaxPool1d(3)		MaxPool1d(3)	MaxPool1d(3)	BatchNorm1d + Residual
Layer4	Conv1d(256, 512, kernel = 15)		Conv1d(256, 512, kernel = 7)	Conv1d(256, 512, kernel = 7)	
	BatchNorm1d		BatchNorm1d	BatchNorm1d	
	Relu		Relu	Relu	
	Dropout(0.6)		Dropout(0.6)	Dropout(0.6)	
	AvgPool1d(3)		MaxPool1d(3)	MaxPool1d(3)	
Layer5	Linear(3072,4)		Conv1d(512, 1024, kernel = 7)	Conv1d(512, 1024, kernel = 7)	
			BatchNorm1d	BatchNorm1d	
			Relu	Relu	
			Dropout(0.6)	Dropout(0.6)	
			AvgPool1d(3)	AvgPool1d(3)	
Layer6	NA		Linear(1024,4)	Linear(1024,2048)	
				Relu	
Layer7			NA	Linear(2048,4)	

APPENDIX B. CRNN Structures and Performances

Table B1. Investigated Models

Model	No. of Layers	No. of Conv. Layers	No. of FC Layers	No. of LSTM Layers	No. of Parameters	Optimizer	Validation Accuracy	Testing Accuracy
CRNN(1-LSTM)	7	4 (3 before time = 300)	1	1	508,932	Adam	96.2%	65.9%
CRNN(2-LSTM)	8	4 (3 before time = 300)	1	2	672,772	Adam	94.9%	64.8%
CRNN(3-LSTM)	9	4 (3 before time = 300)	1	3	713,732	Adam	92.3%	66.4%
CRNN(4-LSTM)	10	4 (3 before time = 300)	1	4	723,972	Adam	82.3%	62.8%

Table B2. Model Details

	CRNN(1-LSTM)		CRNN(2-LSTM)		CRNN(3-LSTM)		CRNN(4-LSTM)
Layer1	Conv2d(1, 10, kernel = 16) elu BatchNorm	Layer1	Conv2d(1, 10, kernel = 16) elu BatchNorm	Layer1	Conv2d(1, 10, kernel = 16) elu BatchNorm	Layer1	Conv2d(1, 10, kernel = 16) elu BatchNorm
Layer2	Conv2d(21, 1, kernel = 32) elu BatchNorm MaxPooling2d(1, 4)	Layer2	Conv2d(21, 1, kernel = 32) elu BatchNorm MaxPooling2d(1, 4)	Layer2	Conv2d(21, 1, kernel = 32) elu BatchNorm MaxPooling2d(1, 4)	Layer2	Conv2d(21, 1, kernel = 32) elu BatchNorm MaxPooling2d(1, 4)
Layer3	Conv2d(1, 10, kernel = 64) BatchNorm MaxPooling2d(1,4)	Layer3	Conv2d(1, 10, kernel = 64) BatchNorm MaxPooling2d(1,4)	Layer3	Conv2d(1, 10, kernel = 64) BatchNorm MaxPooling2d(1,4)	Layer3	Conv2d(1, 10, kernel = 64) BatchNorm MaxPooling2d(1,4)
Layer4	Conv2d(1, 10, kernel = 128) BatchNorm MaxPool2d(1, 4) Dropout(0.4)	Layer4	Conv2d(1, 10, kernel = 128) BatchNorm MaxPool2d(1, 4) Dropout(0.4)	Layer4	Conv2d(1, 10, kernel = 128) BatchNorm MaxPool2d(1, 4) Dropout(0.4)	Layer4	Conv2d(1, 10, kernel = 128) BatchNorm MaxPool2d(1, 4) Dropout(0.4)
Layer5	Permute(2, 3, 1) Flatten	Layer5 Layer6	Permute(2, 3, 1) Flatten LSTM(128)	Layer5 Layer6 Layer7	Permute(2, 3, 1) Flatten LSTM(128) LSTM(64)	Layer5 Layer6 Layer7 Layer8	Permute(2, 3, 1) Flatten LSTM(128) LSTM(64) LSTM(32)
Layer6	LSTM(128) Dropout(0.4)	Layer7	LSTM(64) Dropout(0.4)	Layer8	LSTM(32) Dropout(0.4)	Layer9	LSTM(16) Dropout(0.4)
Layer7	Dense(4)	Layer8	Dense(4)	Layer9	Dense(4)	Layer10	Dense(4)