BiDirectional Neural Networks and Convolution Neural Networks for EEG Alcoholic Binary Classification

Jiahao Zhang[1]

Austrilia National University
u6921098@anu.edu.au

Abstract. BiDirectional Neural Network (BDNN) is an idea to train the neural network not only forward, but also backward (reverse forward). Convolutional Neural Network (CNN) is a successful model to extract features from 2 dimensional data. The ICU EEG Alcoholic Dataset contains EEG (Electroencephalography) data and corresponding Alcoholic and stimulus labels. In this paper, a method is proposed to apply the BDNN on the ICU EEG Alcoholic Dataset for alcoholic binary classification. This method converts binary labels to multiple labels for training and convert them back for testing. This would increase the invertibility of the model, hence improve the robustness when learning representations on both directions. Comparing with three types of datasets and four types of networks, improvements on accuracy can be found. And we also converted the EEG singals to 2D image-like data, hence be able to apply popular CNN models like AlexNet, VggNet and ResNet. But it seems that CNN models are not suit for this problem, and the reasons and potential improvements are discussed.

Keywords: BiDirectional Neural Networks · Convolution Neural Networks · UCIEEG dataset · Machine Learning · Deep Learning · Pattern Recognition · EEG · BDNN. · CNN.

1 Introduction

EEG signal is complex and time consuming for a human doctor to observe and diagnosis, however it is a simple binary classification task for computers. With proposed methods, we can distinguish whether a subject is alcoholic or not automatically. But limited by non-invertibility of binary labels, traditional BDNNs did not perform well. Our goal is improving performance by multi-label for BDNNs.

Besides that, we also convert the signal data into 2D RGB images, and apply CNNs LeNet [9], AlexNet [7], VggNet [10] and ResNet [4] on it as comparsion.

As shown in the Fig. 1, BiDirectional Neural Network is different from traditional Multilayer Perceptron (MLP). Where BDNNs have reverse forward while training. Training a neural network with forward propagation makes the network learn to do prediction base on the input data. However, with reverse propagation, the network would learn about generating input data with predictions. This is a bio-inspired technique. For example, intuitively given a photo of a dog one can easily tell that is a dog. And given a word dog, it is still not tough to get an image of dog in the brain.

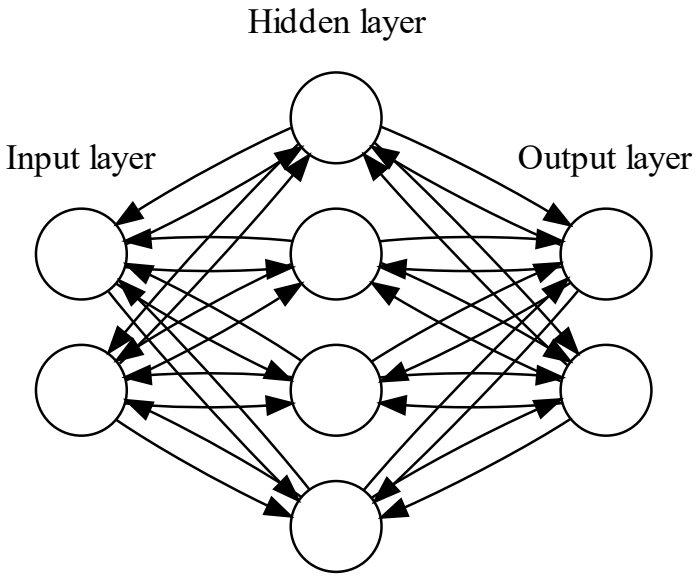


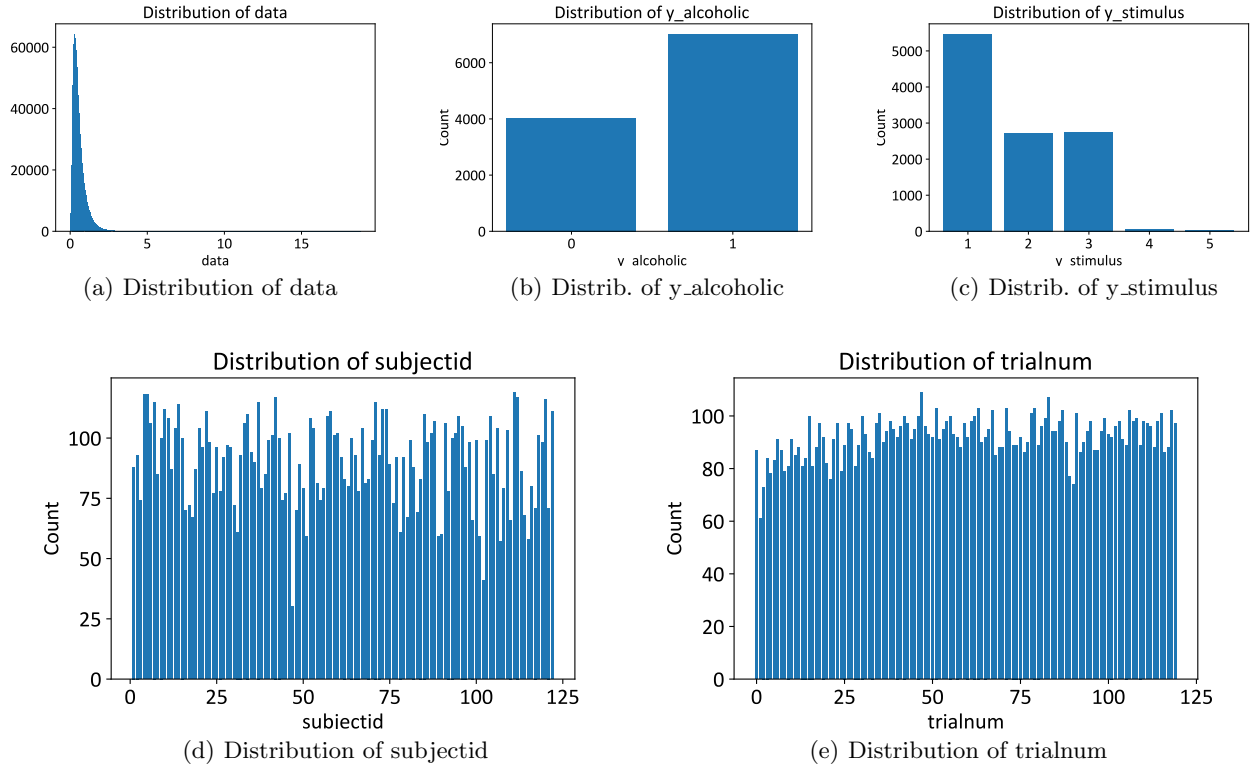
Fig. 1. BDNN Illustration.

The Convolution Neural Network (CNN) has become popular in computer vision field since the LeNet [9]. It uses a convolution layer followed by a pooling layer instead of multi-layer perceptron to extract features from 2d images, such as the handwritten characters classification. This pattern of the LeNet has become standard sub-module for most of later CNN model architectures. After that we got AlexNet [7] which is the champion of ILSVRC-2012. It first uses **Relu** as activation function to replace **Sigmoid**, **MaxPooling** instead of **AvgPooling** and introduces **Dropout** to avoid overfitting. It also proposed Local Response Normalization (LRN) for better model generalization. And is implemented in CUDA which brought GPU's calculation power into model training. As going deeper and deeper, the model is more and more likely to be overfitting, gradient vanishing and explosion. Besides that, more layers mean more parameters, hence more computation power required. This led to Inception series from Google, v1 [13], v2 [5], v3 [14] and v4 [12]. Inception v1 adds 1 x 1 convolution layer before non 1 x 1 convolution layer to a inception module, which reduces the number of channels of feature maps, hence require lower computation power. Inspried by VGG [10], Inception v2 and v3 replace large convolution kernel with multiple smaller ones, e.g. replace one 7 x 7 kernel with a coomposition of one 1 x 7 kernel and one 7 x 1 kernel, for further computational reduction. And introduces Batch Normalization (BM) to the model to overcome overfitting. Inception v4 brought residual connections inspired by ResNet [4]. Residual means that the model has connections between discontinuous layers to make the gradient passes through. This technique is proved to be the solution for very deep convolution neural network design. It will prevent degradation issue which is caused by depth related problem like gradient vanishing and explosion. With residual connections, Inception v4 can go even deeper to get better result and performance.

The dataset is original on UCI Machine Learning Repository [3], and a processed version is provided in [15]. The modification is for each sample, instead of 256 signals for one second, the average is used. The dataset has in total 11057 samples. Each sample is composed of five parts [11].

- *data* is 192 EEG features. It can be divied into three bands in the order of *theta*, *alpha*, *beta*. Each band contains 64 EEG features. Which corresponds to the averaged signal from 64 electrodes [2].
- *y_alcoholic* is the zero or one label that denotes whether the corresponding subject is alcoholic or not.
- *y_stimulus* is the stimulus for the subject. It contains 5 different types denoted by 1, 2, 3, 4, 5.
- *subjectid* is the identifier for each subject. There are in total 122 subjects denoted by 1, 2, ..., 122.

– *trialnum* is the trial identifier. There are around 120 trials for each subject denoted by 0, 1, ..., 119.



(a) Distribution of data (b) Distrib. of y_alcoholic (c) Distrib. of y_stimulus

(d) Distribution of subjectid (e) Distribution of trialnum

Fig. 2. Distribution of UCI EEG Dataset

As shown in Fig. 2, the distribution of $y_alcoholic$ is not balanced. We have almost twice 1s as much of 0s. And for *subjectid* and *trialnum*, they are almost distributed evenly. We then normalize the *data* to the range $[0, 1]$ to prevent overfitting.

2 Method

BDNN structure

– BDNN is a simple BiDirectional Neural Network with 192 input neurons, 50 hidden neurons.
– BDNN2 is a two layer BDNN, where the first layer has 100 hidden neurons and 50 in the second layer.

For each layer, the RELU activation function is used to introduce nonlinearity [1]. The number of neurons in the output layer is determined by the dataset. E.g. the original dataset corresponds to two output neurons, because it has only two labels, True and False.

BDNN training Take BDNN2 as an example. Assume W_1, b_1 denotes weight and bias between input layer and first hidden layer. And W_2, b_2 between two hidden layers, W_3, b_3 for the second layer and output. Additionally, *Input* for input matrix and *Output* for output. The activation function is omitted in the equations.

During forward propagation,

$$Output = ((InputW_1^T)W_2^T + b_2)W_3^T + b_3 \tag{1}$$

For reverse forward propagation,

$$Input = ((OutputW_3)W_2 + b_2)W_1 + b_1 \tag{2}$$

Then two different losses are added up to represent the overall loss. Cross Entropy Loss for forward classification, and Mean Square Error (MSE) Loss for reverse forward generation. The Adam (adaptive moment estimation) [6] optimizer is applied for gradient descent.

BDNN testing As a binary classification task, the accuracy metric is used. What worth noticing is that no matter how many labels a dataset has, it needs to be converted to True or False eventually with Eq. 3.

$$y_{predict} = \begin{cases} False & Argmax(Softmax(Output)) < \frac{\#class}{2} \\ True & Argmax(Softmax(Output)) \geq \frac{\#class}{2} \end{cases} \tag{3}$$

2.1 Four types of dataset

As a binary classification task, the original label is True and False. But for BDNN, two labels would lead to extremely low performance. Due to the fact that, when learning representations in reverse forward, there are not much features the model can extract from two label only input. Intuitively, True and False can not generate EEG signals. Hence, we need to enlarge or expand the labels. For this purpose, two datasets: Alcoholic Stimulus Dataset and Alcoholic Subject Dataset are introduced below.

Alcoholic Dataset This is the original dataset, where the input is 11057 x 192 EEG features and the labels are: 1s for alcoholic, 0s for non-alcoholic. This dataset is introduced as a control variable, it would provide baseline for comparison.

Alcoholic Stimulus Dataset As *y_stimulus* denotes what kind of stimulation that the subject receives. It is Intuitively that EEG signals can inferred from a combination of stimulation type and whether subject is alcoholic or not. The labels are combinations of *y_alcoholic* and *y_stimulus* as shown in Tbl. 1.

Table 1. Labels for Alcoholic Stimulus Dataset.

y_alcoholic	*y_stimulus*	label
0	1	0
0	2	1
0	3	2
0	4	3
0	5	4
1	1	5
1	2	6
1	3	7
1	4	8
1	5	9

Alcoholic Subject Dataset Similar to Alcoholic Stimulus Dataset but replace *y_stimulus* with *subjectid* as shown in Tbl. 2.

Table 2. Labels for Alcoholic Subject Dataset.

y_alcoholic	*subjectid*	label
0	1	0
0	2	1
...	...	...
0	121	120
0	122	121
1	1	122
1	2	123
...	...	...
1	121	242
1	122	243

Alcoholic Image Dataset Inspired by Bashivan [2], we can convert EEG signals into 2D images. As the EEG electrodes are distributed over the scalp in a 3D space, we first project the 3D locations onto a 2D surface using Azimuthal Equidistant Projection (AEP). Then the coorsponding three EEG signal bands are mapped to RGB channels, hence we obtain the image. Now the size of each data sample is $3 \times 32 \times 32$. However for recent CNNs like AlexNet [7], the minimum size is 224×224, hence we scale the image into that size with bilinear interpolation.

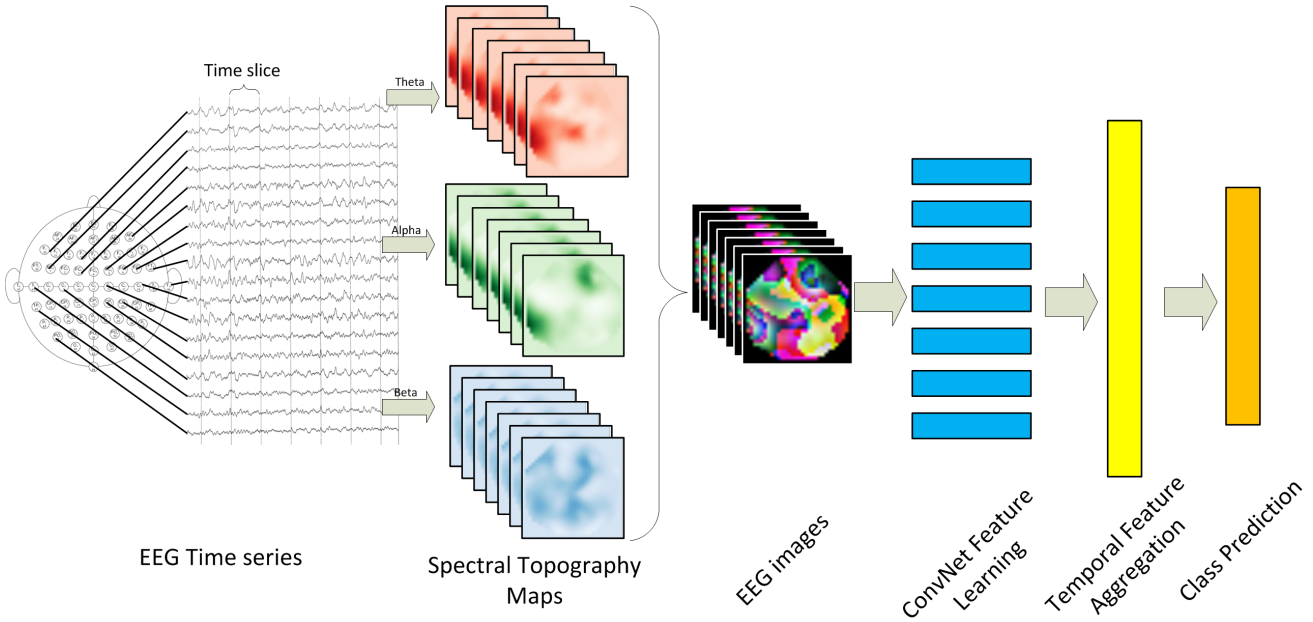


Fig. 3. Illustration of CNN classification on images generated from EEG signals. [2]

3 Results and Discussion

All the combinations of four models (DNN, DNN2, BDNN, BDNN2) and three non-image datasets are tested. And four CNNs (LeNet, AlexNet, Vgg, Resnet) are tested on image dataset, with fine-tune. Some interesting results cross models and cross datasets are found.

Two simple networks called DNN and DNN2 are built and tested to be compared with BDNN and BDNN2. The DNN2 is a fully connected neural network with 192 input neurons, 100 neurons in the first hidden layer and 50 in the second hidden layer. Like BDNN and BDNN2, the number of neurons in the output layer is dynamic determined by the dataset used. The DNN is a simplified version of DNN2 with only one hidden layer contains 100 neurons. And they are trained and tested with same hyperparameters with BDNN and BDNN2.

The dataset split is 7:3 with a constant random seed for reproducible result. The batch size is manually set to be 16, learning rate to be 0.0001 and epoch to be 500. What worth noticing is that, learning rate seems to have great

impact on the final accuracy. Slightly higher learning rate like 0.01 could result in non-converge. The test accuracy does have a trend to go higher after 500 epochs, but it takes to too long to get a significant further improvement.

Table 3. Test accuracy result.

index	dataset	model	accuracy
0	Alcoholic	DNN	0.889693
1	AlcoholicStimulus	DNN	0.839361
2	AlcoholicSubject	DNN	0.605485
3	Alcoholic	DNN2	**0.899940**
4	AlcoholicStimulus	DNN2	0.805606
5	AlcoholicSubject	DNN2	0.574141
6	Alcoholic	BDNN	0.865883
7	AlcoholicStimulus	BDNN	0.870403
8	AlcoholicSubject	BDNN	0.650693
9	Alcoholic	BDNN2	0.867691
10	AlcoholicStimulus	BDNN2	0.837251
11	AlcoholicSubject	BDNN2	0.632911
12	AlcoholicImage	LeNet5	0.848732
13	AlcoholicImage	AlexNet	0.874938
14	AlcoholicImage	Vgg11	0.850214
15	AlcoholicImage	ResNet18	0.836391

3.1 Cross Model

On this specified binary classification, DNN does overperform proposed BDNN with similar network structure no matter it is DNN1 or DNN2. Especially the highest accuracy is obtained by DNN2 on Alcoholic (3). For dataset Alcoholic, increasing the number of hidden layers also increases the performance. This is intuitive, since the model can learn more general and complex features, hence higher accuracy.

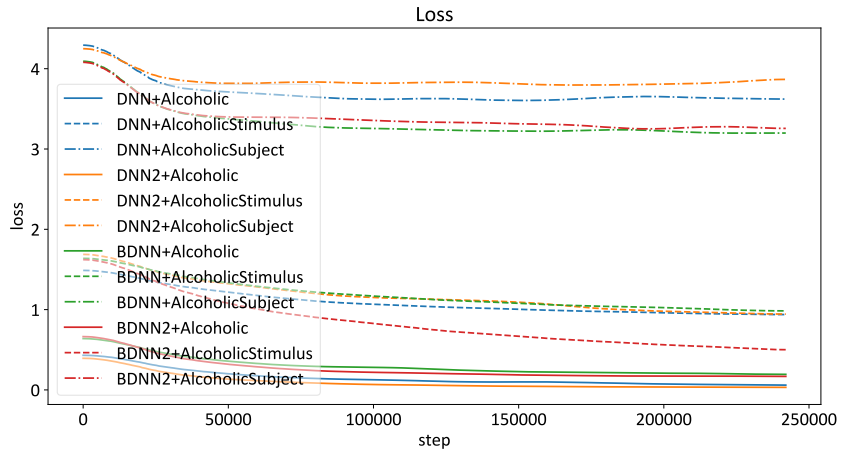
However, for dataset AlcoholicStimulus and AlcoholicSubject, the results are completely opposite. The worst accuracy 0.574141 comes from DNN2 with AlcoholicSubject (5). This is even lower than predict all 1s, which would get 0.636067 correct. This could relate to multiple factors. As shown in Fig. 4(b), the train accuracy peaks at the very beginning, then drops down slowly. It may be trapped in local minimum.

Among the four CNN models, the AlexNet overperforms others. But the even AlexNet (13) does not overperforms the original simple DNN (0). One possible factor could be that, converting EEG signal to images will loss some representations of the original data. Plus the CNN is designed by extracting feature maps from the images, from small features like horizontal lines, vertical lines and corners. But those features are not suitable for EEG signals, and the those small features have no meaning for the EEG itself. Another possible reason is the lack of data, resulting in overfiting, resulting in overfitting. We found that during training, the accuracy of train dataset is already 100%, while the accuracy of test dataset stuck.

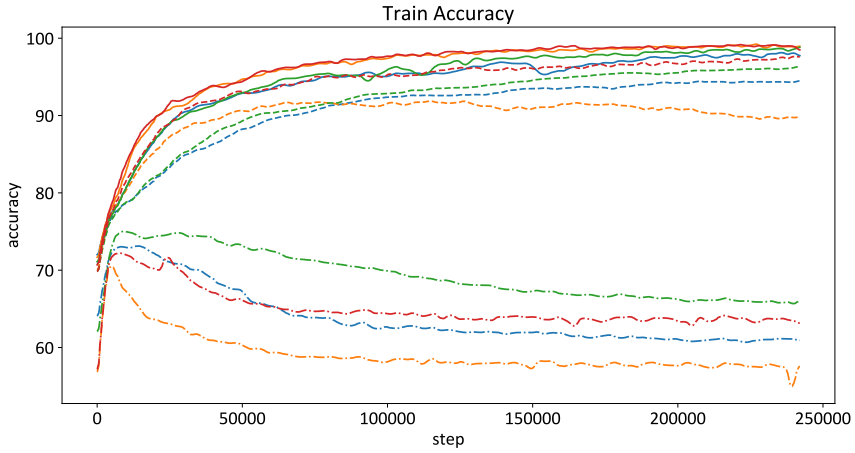
3.2 Cross Dataset

Comparing BDNN with Alcoholic (6) and BDNN with AlcoholicStimulus (7), the latter one achieves slightly better accuracy. In contrast, for DNN with same datasets (0 v.s. 1) and DNN2 (3, 4), there are much larger declines. Hence, we can conclude that doing multi-label does improve the performance of BDNN on a binary classification task. Originally, we want the reverse forward of BDNN to learn to generate EEG signals from True of False label, which is quite impossible for human. When we apply multi-label to the dataset, BDNN now learns to generate from the combination of stimulation and alcoholic flag. Which is equivalent to enlarge the input feature dimensionality, more features can be extracted in reverse propagation. Hence higher accuracy in general.

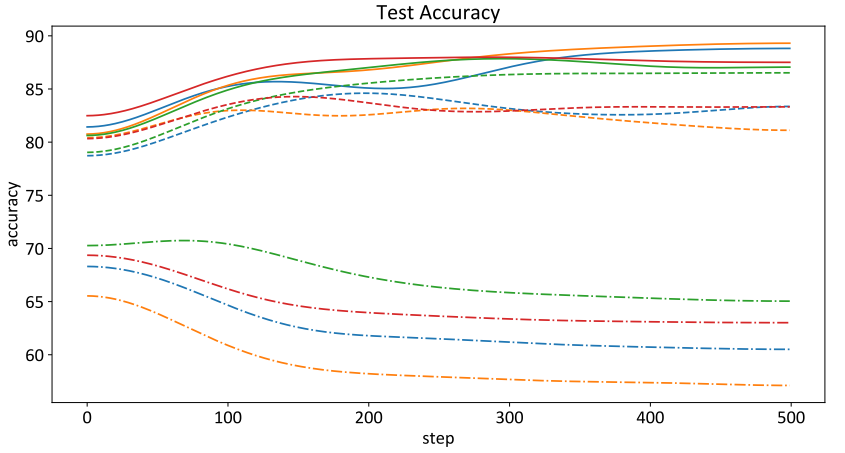
However, for dataset AlcoholicSubject, the training does not work out well. There is a dramatic drop comparing with AlcoholicStimulus for all four models. Too many labels (244) could be the main factor for this issue. Because the hidden layer before the output is 50 with is significantly lower than 244.



(a) Loss duraing training.



(b) Train accuracy.



(c) Test accuracy during trianing.

Fig. 4. Training diagrams for non-image combinations. A Gaussian 1D interpolation ($\sigma = 50$) to y is applied in all figures.

4 Conclusion and Future Work

In this paper, we try all combinations of two DNNs plus two BDNNS and three different types of datasets, and four popular CNNs on converted EEG image dataset to solve alcoholic binary classification task. We found that using multi-label does improve the performance of BDNN. Although for this specific task, DNN seems overperform BDNN natively and even rencent CNNs.

There are so many works to do in the future obviously. On the one hand, for BDNNs there are still many options left to be explored. E.g., as discussed in Sec. 3, increasing the number of neurons in the last hidden layer. Or even go deeper with BDNNs. In this paper, a reverse forward is followed by forward for each batch, which is not good intuitively. Because it may weaken the forward and vice versa. It may benefit on doing several forward propagations first, and then apply equal number of reverse forward propagations.

As for the dataset, we randomly split train and test dataset, this is called within subject split [8]. Cross subject, which means that no seen subjects are in the test set, should be implemented to test the model's generalization.

The original EEG dataset contains 256 data points for each sample, which is averaged in this paper. With that data, we can get more detailed sequential features, which may help us to get better result. For example, using the original data, we can introduce Recursive Neural Network (RNN) like LSTM, and with converted EEG images, we can apply 3D Convolutional Neural Networks (3DCNN) to further improve the classification accuracy.

References

1. Agarap, A.F.: Deep Learning using Rectified Linear Units (ReLU). arXiv:1803.08375 [cs, stat] (Feb 2019), http://arxiv.org/abs/1803.08375, arXiv: 1803.08375
2. Bashivan, P., Rish, I., Yeasin, M., Codella, N.: Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks. arXiv:1511.06448 [cs] (Feb 2016), http://arxiv.org/abs/1511.06448, arXiv: 1511.06448
3. Begleiter, H.: UCI Machine Learning Repository: EEG Database, http://archive.ics.uci.edu/ml/datasets/EEG+Database
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. arXiv:1512.03385 [cs] (Dec 2015), http://arxiv.org/abs/1512.03385, arXiv: 1512.03385
5. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv:1502.03167 [cs] (Mar 2015), http://arxiv.org/abs/1502.03167, arXiv: 1502.03167
6. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs] (Jan 2017), http://arxiv.org/abs/1412.6980, arXiv: 1412.6980
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (May 2017). https://doi.org/10.1145/3065386, https://dl.acm.org/doi/10.1145/3065386
8. Lawhern, V.J., Solon, A.J., Waytowich, N.R., Gordon, S.M., Hung, C.P., Lance, B.J.: EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces. Journal of Neural Engineering **15**(5), 056013 (Oct 2018). https://doi.org/10.1088/1741-2552/aace8c, http://arxiv.org/abs/1611.08024, arXiv: 1611.08024
9. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998). https://doi.org/10.1109/5.726791
10. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556 [cs] (Apr 2015), http://arxiv.org/abs/1409.1556, arXiv: 1409.1556
11. Sykacek, P., Roberts, S.J.: Adaptive Classification by Variational Kalman Filtering p. 8
12. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:1602.07261 [cs] (Aug 2016), http://arxiv.org/abs/1602.07261, arXiv: 1602.07261
13. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going Deeper with Convolutions. arXiv:1409.4842 [cs] (Sep 2014), http://arxiv.org/abs/1409.4842, arXiv: 1409.4842
14. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567 [cs] (Dec 2015), http://arxiv.org/abs/1512.00567, arXiv: 1512.00567
15. Yao, Y., Plested, J., Gedeon, T.: Information-preserving feature filter for short-term eeg signals. Neurocomputing **408**, 91–99 (2020)