Brain Activity Recognition with Deep Convolutional Neural Networks

Zhengxing Yan

June 2021

Abstract

Brain computer interface (BCI) is a system that captures brain signals and provides a pathway to connecting human brains and external devices. With the improvement in safety and convenience of BCI, it becomes accessible to ordinary business vendors and individuals. BCI has extensive practical use in our daily life, with applications ranging from personal assistant to entertainment equipment. Despite the popularity, analyzing and utilizing BCI is a nontrivial topic. In general, the recorded signals reflect the cortical electrical activity and are present in Electroencephalography (EEG) waves. Such raw signals can be quite noisy and difficult to analyze. As such, a more robust approach is needed to achieve satisfactory efficacy as well as efficiency. To this end, we propose a novel deep learning based approach for brain activity recognition from EEG signals. Deep learning has been revolutionizing the field of artificial intelligence and has made remarkable achievements in tasks such as image recognition and machine translation. Deep learning is extremely suitable for representation learning for complex input, and is flexible in the sense that several components can be integrated to build a more powerful system.

In this work, we present a deep convolutional neural network based approach for brain activity recognition. Specifically, our model is composed of multiple 1-D convolutional neural networks for signal process and a multilayered nonlinear classifier. The advantages of using such a model is that (1) 1-D dimensional CNN can easily process the lengthy EEG signals and extract reasonable features for further use; (2) The nonlinear layer with regularization method such as weight decay and dropout can learn from the signal abstraction effectively, thus leading to performance boost.

To verify the proposed approach, we conducted extensive experiments on real world EEG signals. We are concerned with learning cross-subject brain activity recognition where signals from different subjects are mixed and an individual independent classifier is built based on it. In some sub-tasks, our model can achieve an exciting accuracy hitting 95%, which demonstrates the effectiveness and promise of this approach.

Keywords: **Brain Activity Recognition, Electroencephalography (EEG), Convolutional Neural Network (CNN), Multilayer Perceptron (MLP)**

Contents

1 Introduction

Brain-Computer Interface (BCI) is a system that captures brain signals and provides a pathway to connect human brains and external devices. With improvements in the safety and convenience of BCI, it may become valuable to ordinary business and individuals.

In 1929, Hans Berger successfully extracted brain electrical waves with the method he named as Electroencephalography (EEG) in [1]. The method was noninvasive, which made it more accepted by the public. Since then, researchers started to look deeper into processing the brain waves which is now seen as a key to make BCI become a true invention rather than a pure conception.

Since EEG has grown more popular over time, the concept of EEG-based brain activity recognition was drawn out. Such concept has high expectations when combined with BCI. However, there are still some severe challenges that needs to be solved:

1. Accuracy concerning brain activity recognition generally stays at a low level. To make EEG-based brain activity recognition valuable in the future stages of researches, people have to come up with higher accuracy;

2. Many researches focus on intra-subject brain signal recognition where a classifier on one person can not be transferred to another person, which limits the generalizability and flexibility of the built classifiers.

Therefore, in this research paper, we will focus on the binary recognition of EEG signals and we will propose an algorithm we call as the DL-Brain which accomplishes the following:

1. Combines CNN layers together with Multilayer perceptron (MLP) layers to learn from raw EEG signals, which enhances the model's robustness and efficiency;

2. Conducts extensive experiments on real-world EEG datasets, with satisfying results that show promising performances across different challenging tasks.

2 Related Work

Especially in this millennium, researchers have started to look deeper into the topic of EEG, Neural Networks, and understanding our brains. For example, Meisheri et al., in their article [2], proposed a categorizing model combining Multi-class Common Spatial Pattern (mCSP) with Self-Regulated Interval Type-2 Neuro-Fuzzy Inference System (SRIT2NFIS) for classification of four motor imagery (MI) classes, and achieved a successful rate of 54.63%; Dellaert et al. built a model based on K-Nearest Neighbors (KNN) to recognize the emotions in our speeches and achieved a 72% accuracy [3]; Abdulhamit Subasi has came up with an algorithm combining Artificial Neural Network (ANN) and wavelet coefficient to recognize the alertness level in [4]. He picked on 12 specific subjects and reached the highest accuracy of 98%. All these works have build strong and harsh basis for the further researches in brain activity recognition; Lawrence et al. proposed a CNN approach in facial recognition task and achieved an error rate of 3.8% [5]; In research [6], Zhang et al. developed a model with 7 layers of Recurrent Neural Network (RNN) in order to recognize the subjects' intents. Their best accuracy was 95.53%.

In our research, we returned to the basic binary brain activity recognition on multiple subjects rather than multiple classes. The reason was we wanted to try and build stronger models that can ensure better accuracy in simple recognition, so that the future researches can be more promising and we can enlighten them with more inspirations.

3 The Proposed Methodology

In this section, we will present and describe our proposed approach. Our model operates in an end-to-end manner, which means it does not need a separate model for feature extractions.

3.1 The Architecture of DL-Brain

The structure of this model is as shown in Figure 2.1.1. We define the main functions of our codes as:

- Creates a model regarding CNN combined with multiple layers of MLP to recognize the brain activity by predicting the contents in the test label;
- Compares the actual binary labels with the prediction ones after every epoch, then train and update the model to make it more accurate.

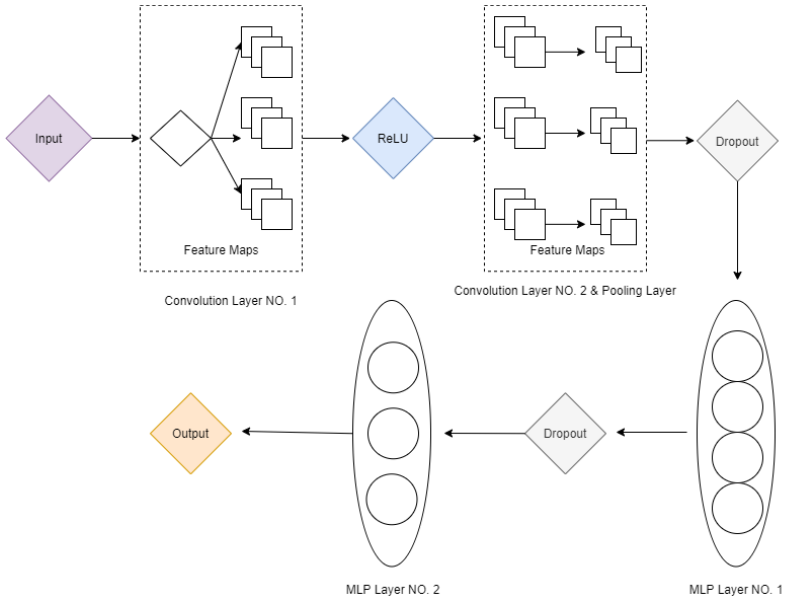


Figure 3.1.1: Prediction Process of DL-Brain

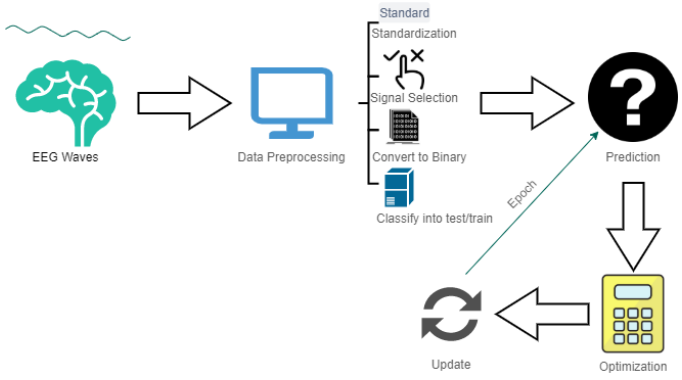


Figure 3.1.2: Workflow of the Algorithm

Below we will give more detailed explanations on each component used in the above architecture.

3.2 CNN

In 1989, LeCun et al. firstly proposed the Convolutional Neural Network (CNN) in [7]. CNN stands out when compared to the other neural networks. With extractions of significant features, CNN can more easily capture the more obvious parts of a label, which enhances its accuracy while learning.

3.2.1 Convolution Layer

The Convolution Layer belongs to the hidden layer in a CNN that enlarges the special features using a fixed-dimension kernel. The mathematical representation of this process with only one stride for each move of the kernel is shown below where f denotes the input, h denotes the kernel, a denotes the number of rows of the result matrix, and b denotes the number of columns of the result matrix:

$$G[a,b] = (f * h)[a,b] = \sum_j \sum_k h[j,k] f[a-j, b-k] \tag{1}$$

As we can see both from the equation above and the figure below (Figure 2.2.1), a convolution layer can reduce some dimensions for the input data, however, it serves more to detection of salient features from the input data and it makes these features stand out more.

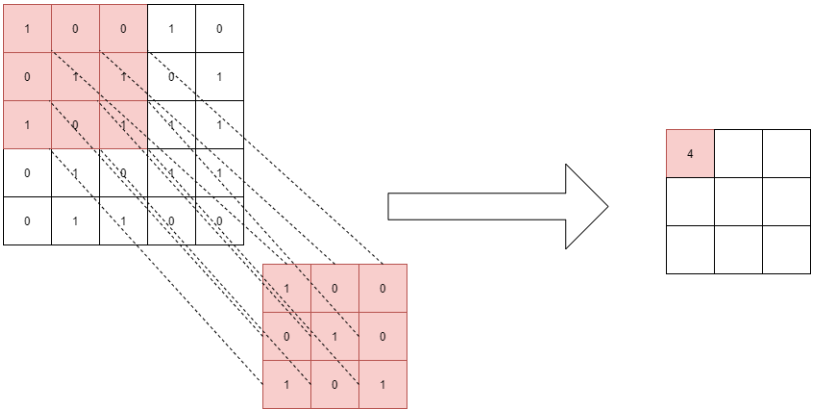


Figure 3.2.1: Example of 2-D Convolution with 5*5 input, 3*3 kernel

3.2.2 Pooling Layer

The pooling layer is used to downsample the high-dimensional input data and keep the most significant features. Such layer provides significant helps in deducting the dimension of the input, and can also extract special features from the input data.

Max-pooling is a function that can identify the most obvious feature within the filter that has a set dimension like it is shown in Figure 2.2.2. This dimension of the output matrices can be mathematically represented as below as the dimension of the filter is defined as p*q:

Firstly, there are two variables that we need to define first:

$$r = 0, 1, 2, 3, ..., p-1 \tag{2}$$

$$s = 0, 1, 2, 3, ..., q - 1 \tag{3}$$

And the output should be shown like below as x remarks the input matrix:

$$y_{ij} = max(x_{i+r,j+s}) \tag{4}$$

Additionally, we shall be aware of some limitations in the ranges of i and j as below when the input matrix's size is defined as m*n:

$$i \leq m - p \tag{5}$$

$$j \leq n - q \tag{6}$$

Therefore, we can see that if we set the dimension of the filter applied in the max-pooling function is relatively big, the dimension of the input can be deducted significantly.

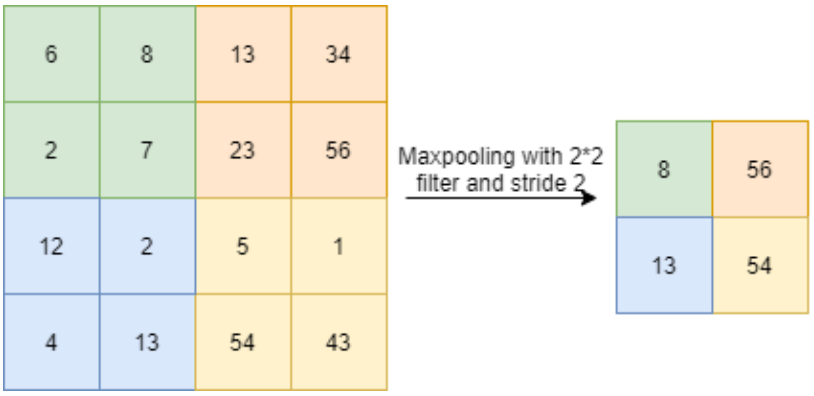


Figure 3.2.2: An Example of 2-D Maxpooling

3.3 MLP

MLP was firstly invented by Frank Rosenblatt [8], it belongs to the class of Feedforward Artificial Neural Network (ANN). It includes the three most basic layers: the input layer, the hidden layer, and the output layer. The architecture of MLP is shown below in Figure 2.3.1.

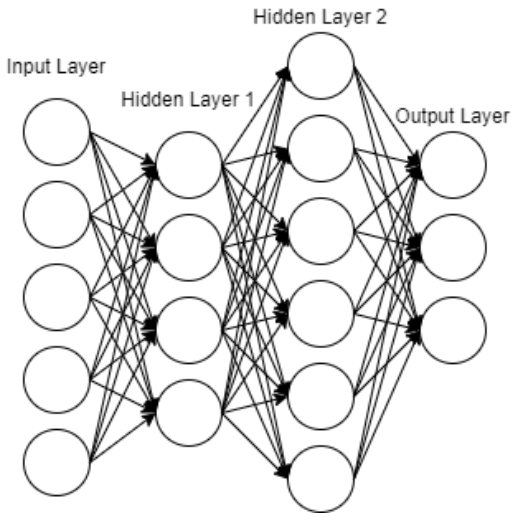


Figure 3.3.1: Concept of MLP

If we use b_k to represent the bias of the data, and use W_k to denote weights, which are the coefficients of the input matrices, we can get the following mathematical model to explain this kind of neural networks:

$$y(x) = W_k(...(W_2(W_1x + b_1) + b_2)... + b_{k-1}) + b_k \tag{7}$$

3.4 ReLU

In most neural networks, only part of the output from the previous layer can become the input to the next layer of nodes. We need to do some calculations to certify which data should be transmitted. We call such calculations the activation function. In this research paper, we adapted the Rectified Linear Unit (ReLU) function invented by Nair et al [9]. ReLU function is a nonlinear function, and the way it works can be shown:

$$f(x) = max\{0, x\} \tag{8}$$

The ReLU function creates non-saturation of the gradient, which results in faster convergence in the Stochastic Gradient Descent (SGD) that was used in the model and will be introduced in Section 2.5.2.

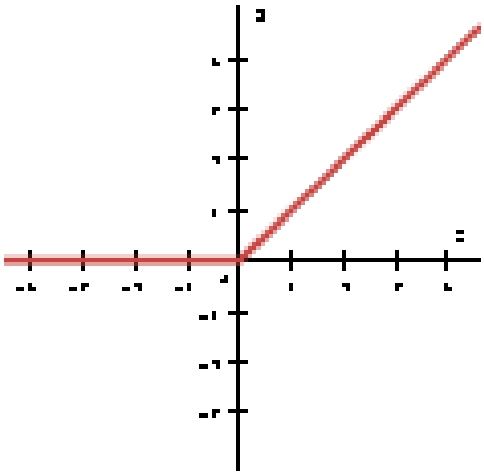


Figure 3.4.2: The ReLU Function

3.5 Model Optimization

Optimization is also an important part of a Deep Learning algorithm. Since the codes are also trying to do the right predictions, the problems in the model must be found, fixed and updated all the time.

In each epoch, inevitably, there are always mistakes and differences between the true label and the predicted label. And our goal is to make such difference to its minimum.

3.5.1 Loss Function

To compute the loss, which are the differences between the output labels and desired labels, we can utilize variety of loss functions. Here, we chose the Cross-Entropy Loss.

To apply the cross-entropy loss function, there is one requirement, which is that the outcomes must be binary since this function was derived from the Bernoulli Distribution [10]. The mathematical representation is the following when x denotes the features of the input, ŷ denotes the predicted label and y denotes the actual label:

$$p(y|x) = \hat{y}^y(1-\hat{y})^{1-y} \tag{9}$$

Therefore we can add logarithms on both sides of the equation, and utilize the logarithm laws, which gives us:

$$logp(y|x) = ylog\hat{y} + (1-y)log(1-\hat{y}) \tag{10}$$

For this possibility, we wish to make it as high as it can be. So, we can make a slight change to the above function to have a minimum value. If L_{CE} marks the Cross-Entropy Loss Function and θ represents model parameters, the Cross-Entropy Loss can be written as:

$$L_{CE}(\theta) = -logp(y|x) \tag{11}$$

3.5.2 Gradient Descent

The main method to minimize the loss is Gradient Descent [11]. We utilized the Stochastic Gradient Descent invented by Robbins et al. in [12].

In SGD, it recalculates the function each time it gets an output sample. But it only uses one instance to represent the whole sample. Therefore it is more efficient when one's dataset has a large set of data. The update rule for learning parameters where α means the learning rate which scales the stepping length of each update is:

$$\theta_j = \theta_j - \alpha\frac{\partial L_{CE}(\theta)}{\partial\theta_j} \tag{12}$$

3.6 Combination

When joining all of the sections we have mentioned above together in a sequence shown in Figure 2.1.2, our model, the DL-Brain, is built.

4 Experiment

In this section, we are going to introduce the experiment settings and results. For the experiment, we decided to focus on 4-6 subjects' EEG waves' analysis.

4.1 Datasets

For freshly extracted EEG impulses, they are classified accordingly to their frequencies as the table below has shown:

Types	Frequency Band	Participant's State
Beta Waves	12 Hz - 30 Hz	active and awake
Alpha Waves	8 Hz - 12 Hz	inactive and awake
Theta Waves	4 Hz - 7 Hz	sleeping
Delta Waves	0 Hz - 4 Hz	deep sleeping

Table 1: Classification of EEG Signals

In this experiment, the EEG files we used are called *"EEG Motor Movement/Imagery Dataset"* which we have obtained from *PhysioNet*[13][14]. Each of the 109 healthy volunteers was asked to perform 14 experimental runs (2 baselines and 3 repetitions of task 3-6) as shown in Table 2.

Task 1	Eyes open
Task 2	Eyes closed
Task 3	Clenches a fist accordingly to the direction of the icon on the screen
Task 4	Imagines to clench a fist accordingly to the direction of the icon on the screen
Task 5	Clenches both fists when the icon appears on the top; Close both feet when the icon appears on the bottom
Task 6	Imagines to clench both fists when the icon appears on the top; Imagines to close both feet when the icon appears on the bottom

Table 2: Testing Instructions

For this experiment, we were only going to test on task 3-6 since task 1 and task 2 were simply some control sets.

A variety of annotations were added artificially to represent the different periods of situations:

1. T0 which means the subject is at a state of resting;

2. T1 which means the subject is at the state of doing the following:

    - Clenching the left fist (which correlated to Task 3 and Task 4);
    - Clenching both fists (which correlated to Task 5 and Task 6);

3. T2 which means the subject is at the state of doing the following:

    - Clenching the right fist (which correlated to Task 3 and Task 4);
    - Putting both feet together (which correlated to Task 5 and Task 6);

The benefits of using this dataset was that for each task, they only had two kinds of reaction and they have been marked with symbols and annotations, which gave a strong basis to our model since we were focusing on binary brain activity recognition.

4.2 Data Preprocessing

In order to make the model fit into the raw instances, few changes must be applied to the signals.We classify the changes into three main categories:

4.2.1 Read and Selection

We firstly selected and read the raw EDF+ files, then we concatenated the raw instance together. After that, we decided to adopt a band-pass filter which was proposed by Hans Berger [15] which allows us to select and extract signals within a set range of frequencies. We designated the frequency range as 7.-30. so that any delta waves were unaccepted (since we don't expect the subject in state of deep sleeping). After that, we applied a function named "pick_types"[16], which can easily help us exclude the signals that were recorded by bad electrodes.

4.2.2 Standardization & Convert

After we've done with fundamental data preprocess, we defined a dictionary with a key named correspondent to the contents in annotations T1 and T2 (T0 was no longer taken into considerations because none of the 4 experiments contains this annotation).

Afterward, we utilized an "epochs" class [16] whose purpose was to read the detailed information more precisely through epochs. Then, we copied the data from 0 seconds to 3 seconds from the epochs so that we could include fewer samples since the amount of it can potentially be huge for 3 2-minutes routines. Then, we turned the values in the dictionary into a big vector with 45 samples.

4.2.3 Training & Test Sets

After finished with processing with the raw data, we moved on to classify them into testing and training subsets. We completed this by using another function named "train_test_split" [17] which randomly split the processed data into training sets and testing sets once we type in the data

we want to split and the size of testing data/training data (the other one would defaulted as the complement ratio). We designated that the size of the testing data should be 0.1, which means the rest 0.9 would be used for training.

4.3 Implementation Details

For basic parameter in the algorithm, the batch size was 64, the number of epochs was 750, and the Learning Rate of the SGD was 0.0001.

For the layers of models, we set the parameters as follow:

1. For the first layer of CNN, the number of input channels was 64 and the number of output channels was 32, with a kernel size of 10, and a stride of 5;
2. For the second layer of CNN, the number of input channels was 32 and the number of output channels was 16, with a kernel size of 5, and a stride of 3;
3. For the first dropout layer, we set the abandon rate as 0.5;
4. For the first MLP layer, the input feature was 16*15, and the output feature was 16.
5. For the second dropout layer, we set the dropout rate as 0.2;
6. For the second MLP layer, the input feature was 16, the output feature was 3.

4.4 Baseline

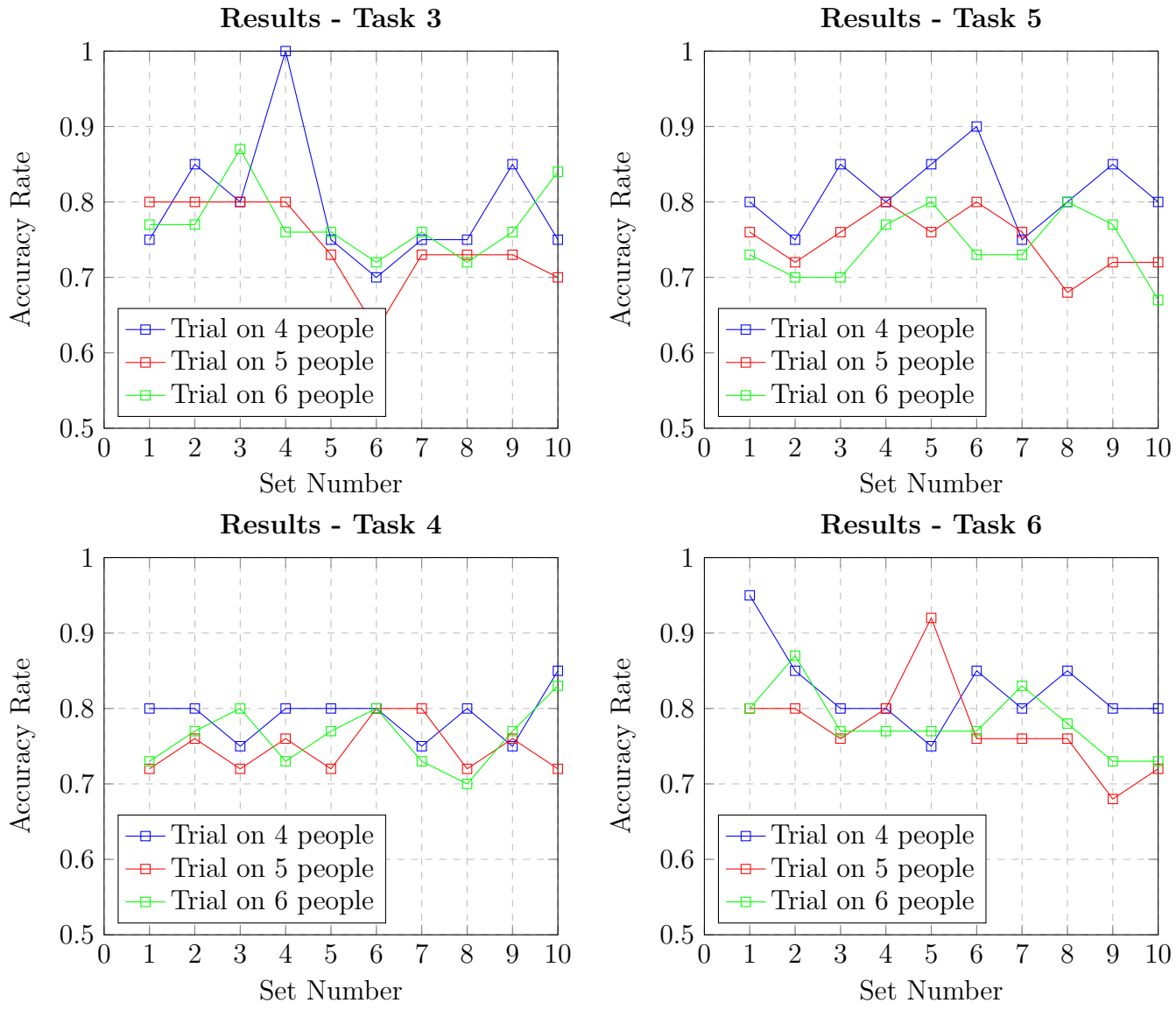
4.4.1 Long-Short Term Memory(LSTM)

LSTM is an artificial Recurrent Neural Network (RNN) architecture and it was invented by Hochreiter et al. [18]. LSTM is also widely used in many fields of research[19][20]. It used to be the basis of our model. The comparison will be shown in Section 3.5 "Results Analysis".

4.5 Results Analysis

In this section, we will show our results of running this algorithm on different amount of people in forms as line graphs so that we can show the results in a more visualized way.

For the following graphs, each reflects the results when the algorithm was ran on an individual task (see in each title), with different colored lines representing different numbers of subjects ranging from 4 to 6.



In order to show the statistics more clearly, some key information were picked up and recalculated. The results are shown below in the table:

BP	MBA: DL-Brain (2 s.f)	MBA: LSTM (2 s.f)
1	0.78	0.65

Table 3: Deep Analysis on Results

"BP" denotes "Best Performance"; "MBA" denotes the "Mean of Best Accuracy" for the two models when they were ran on different tasks and number of subjects

As we can all see, we have out-performed the baseline (LSTM). The 100 percent accuracy rate shall be counted as the highlight of this algorithm.

5 Conclusion

Throughout this research paper, we especially focused on creating a novel model of recognizing the brain signal by predicting the next list of binary codes of the signal and comparing it with the actual ones.

As tables have shown in Section 3.5, the model had the best accuracy when ran on task 3 (100% accuracy) and task 6 (95% accuracy). Such rates have potentials in helping to achieve more possibilities. For example, Task No. 6 included signal collections of the subjects when they were imagining to clench both of their fists or closing their feet. If we can develop more of this model, it may recognize more clearly whether the people are thinking about using both fists or both feet, which would potentially become a great help for the handicapped people to control their prosthesis.

For future works, we will look deeper into more complicated classifications and predictions. We need to keep digging into the theme of brain activity recognition so that computers can have a boost in the understanding of our brains and bodies.

References

[1] Hans Berger. Über das elektroenkephalogramm des menschen. *Archiv für psychiatrie und nervenkrankheiten*, 87(1):527–570, 1929.

[2] Hardik Meisheri, Nagaraj Ramrao, and Suman K Mitra. Multiclass common spatial pattern with artifacts removal methodology for eeg signals. In *2016 4th International Symposium on Computational and Business Intelligence (ISCBI)*, pages 90–93. IEEE, 2016.

[3] Frank Dellaert, Thomas Polzin, and Alex Waibel. Recognizing emotion in speech. In *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, volume 3, pages 1970–1973. IEEE, 1996.

[4] Abdulhamit Subasi. Automatic recognition of alertness level from eeg by using neural network and wavelet coefficients. *Expert systems with applications*, 28(4):701–711, 2005.

[5] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[6] Xiang Zhang, Lina Yao, Chaoran Huang, Quan Z Sheng, and Xianzhi Wang. Intent recognition in smart living through deep recurrent neural networks. In *International Conference on Neural Information Processing*, pages 748–758. Springer, 2017.

[7] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[8] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[9] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[10] James Victor Uspensky. Introduction to mathematical probability. 1937.

[11] Augustin Cauchy. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.

[12] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[13] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

[14] Gerwin Schalk, Dennis J McFarland, Thilo Hinterberger, Niels Birbaumer, and Jonathan R Wolpaw. Bci2000: a general-purpose brain-computer interface (bci) system. *IEEE Transactions on biomedical engineering*, 51(6):1034–1043, 2004.

[15] TJ La Vaque. The history of eeg hans berger: psychophysiologist. a historical vignette. *Journal of Neurotherapy*, 3(2):1–9, 1999.

[16] Alexandre Gramfort, Martin Luessi, Eric Larson, Denis A Engemann, Daniel Strohmeier, Christian Brodbeck, Roman Goj, Mainak Jas, Teon Brooks, Lauri Parkkonen, et al. Meg and eeg data analysis with mne-python. *Frontiers in neuroscience*, 7:267, 2013.

[17] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[19] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber. Learning precise timing with lstm recurrent networks. *Journal of machine learning research*, 3(Aug):115–143, 2002.

[20] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015.