Predicting the Future: A ML MI Replication Study

Written by Marshall McArthur, Xavier Serrano, Viktoriia Zakharova

Abstract

The study of electroencephalography (EEG) signals is an important stepping stone to providing neuroscience explanations for various phenomena. In this paper we focus on the interpretation of EEG signals in regards to motor imagery. Specifically, we want to analyze the differences between typical machine algorithms and deep learning algorithms in performing EEG interpretations. Additionally, there have been no unique studies comparing the performance of convolutional neural networks and deep belief networks directly when used on motor imagery data. Therefore, attempting to replicate results in deep learning experiments between the two might give insight into why this research inquiry has been left unanswered.

1 Introduction

1.1 Literature Review

In the context of feature extraction and classification of EEG signals, deep learning has proven to be an effective tool. Convolutional neural networks (CNNs) provide some of the best classification times and accuracy Craik et al. [2019]. CNNs theoretically work so well with EEG signals due to the fact that convolution layers are able to extract EEG effective features quickly across multiple scales Tang et al. [2020]. The EEG signals in Tang et al. provided a useful real world application, that being a brain-to-wheelchair online BCI system. It was found that the classification success of EEG signals within such BCI systems highly depends on the experimental setup and to where the CNN is getting its data from. Offline systems like the one we propose may fare better than the online systems like Tang et al. because our data is not subject to noise within the experiment and can be preprocessed. The conditional empirical mode decomposition algorithm presented in Tang et al. is not necessary for our study. However, it is mentioned that the effectiveness of intrinsic modal component (IMF) selection for classification is highly dependent on researcher experience with EEG classification algorithms. Another study, Abbas et al. used Common Spatial Pattern (CSP) and Fast Fourier Transform Energy Maps (FFTEM) for feature computation and selection whereas a Convolutional Neural Network (CNN) is proposed as a classifier with the novel features. The proposed model has yielded mean kappa value of 0.61 and achieved the best-reported results with lower computational complexity when compared with state-of-the-art methods. As a result, choosing an effective algorithm that correctly selects IMFs is imperative to our study's success.

Deep belief networks have gained large popularity in classifying EEG signals due to their training procedure, which yields great success in deep neural network optimization. Lu et al. [2016] describes this optimization as a combination of feature extraction and classification into one pipeline, and explains that it is more computationally efficient than other existing classification methods. They note that using frequency domain input for the DBN algorithm improves performance specifically with motor imagery classification, and that their results were in fact statistically significant. Additionally, it is stated that fine tuning the parameters of the DBN structure is imperative for robust results. It is explained that the best way to proceed with tuning is by conducting experiments and proper pretraining techniques prior to the benchmark tasks. The use of DBNs in BCI research is still relatively new, and the application of deep learning in classification with motor imagery EEG data

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

can be a pivotal stepping stone for better BCI developments. Application of DBN in EEG-based BCI is still not common. The main difficulty is the enormously high feature dimensionality spanning EEG channel, frequency, and time. An et al study used a deep belief network model for two-class motor imagery classification, and DBN was found to be more successful than conventional SVM An et al. [2014].

1.2 Background

Motor imagery tasks, a subsection of tasks related to brain-computer interfaces, are those in which a subject imagines doing physical movements without actually performing them Craik et al. [2019]. These thoughts can then be measured and classified using EEG data, coupled with neural network classification, with the eventual goal of successfully classifying such thoughts in the use of brain-computer interfaces (see Figure 1). In other words, the eventual goal of such brain-computer interfaces is to provide control over a computer to a person without the requirement of physical movement Park et al. [2021]. This can provide ease and convenience to the user, using all kinds of software, and can be especially beneficial to those for whom physical movement can be a challenge. As an important and emerging field, it is necessary to investigate and compare algorithms that can serve as useful or useless for future research. Many algorithms that focus on EEG classification for motor imagery tasks involve the use of neural networks Craik et al. [2019]. The effectiveness of many such algorithms has been compared and studied in literature reviews Craik et al. [2019] Lotte et al. [2018]. **Problem Statement:** However, as recently as 2019, no comparisons have been made between DBNs (Deep Belief Networks) and CNNs (Convolutional Neural Networks) in their application to motor imagery tasks Craik et al. [2019]. In order to form a more complete view of the performance of various algorithms in motor imagery classification tasks, in order to provide recommendations for algorithm selection for future work in the field, and in order to offer replication recommendations for newer researchers, further research must be done in order to compare relevant algorithms to one another, which will aid in providing a baseline comparison of the algorithms. **Purpose of Study:** As such, the purpose of this experiment was to focus on the interpretation of EEG signals related to motor imagery. Specifically, we offer a replication of motor imagery studies mentioned in Craik et al. [2019] Hersche et al. [2018] Lotte et al. [2018]. We analyze the differences between typical machine algorithms and deep learning algorithms performing EEG interpretations, including the feasibility of doing so given the published studies. Specifically, result replication under the same conditions were attempted in order to form a proper comparison. **Research Question:** Thus, this paper answers the question posed by previous researchers on which type of machine learning algorithm is most feasible for accurate, and/or efficient classification of motor imagery data.

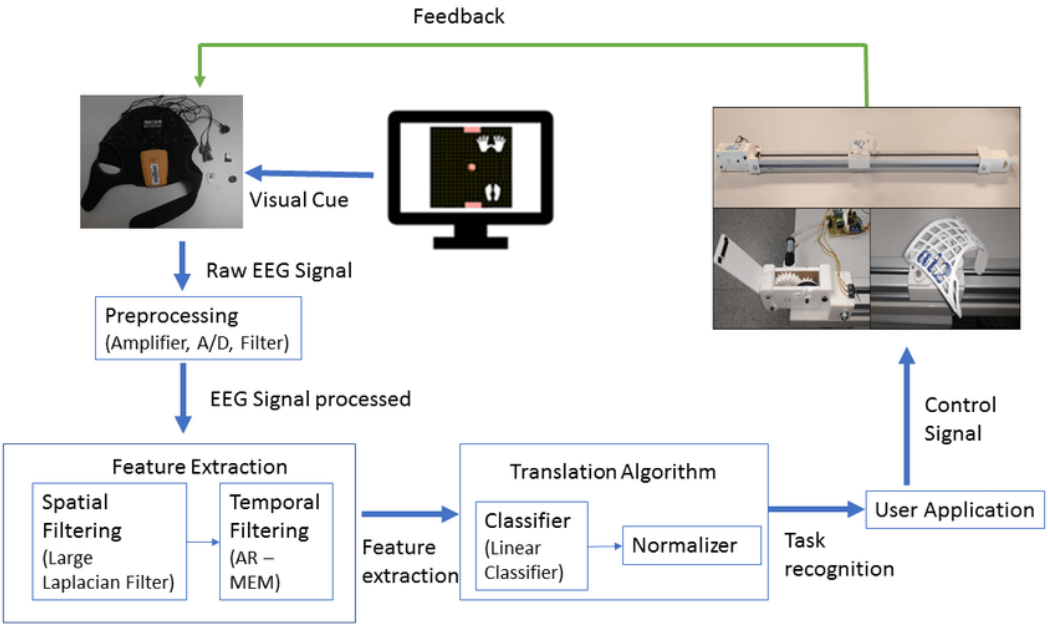

Figure 1: Process of Motor Imagery Classification Quiles et al. [2020]

2 Dataset

We have chosen a dataset that was presented in the 2008 article “BCI Competition 2008 – Graz data set A". We specifically chose the dataset based on the credibility of the BCI competition in their respective research practices, and the number of cited references the dataset has had Craik et al. [2019] Hersche et al. [2018] Lotte et al. [2018]. This data collection contains EEG data from nine people. The cue-based BCI paradigm included four separate motor imagining tasks: imagination of left hand (class 1), right hand (class 2), both feet (class 3), and tongue movement (class 4). Each subject had two sessions filmed on different days. Six runs are separated by short rests in each session. A single run has 48 trials (12 for each of the four classes), for a total of 288 trials each session. A recording of around 5 minutes was made at the start of each session to assess EOG effect. The recording was split into three sections: two minutes with eyes open (looking at a fixation cross on the screen), one minute with eyes closed, and one minute with eye movements. The participants were seated in a plush recliner in front of a computer screen. A fixation cross displayed on the dark screen at the start of each trial (t = 0 s). A brief auditory warning tone was also delivered. A signal in the shape of an arrow pointing left, right, down, or up (corresponding to one of the four classes: left hand, right hand, foot, or tongue) emerged after two seconds (t = 2 s) and lasted on the screen for 1.25 seconds. The individuals were then encouraged to complete the necessary motor imagery task. There were no responses provided. At t = 6 s, the individuals were instructed to complete the motor imagery task until the fixation cross vanished from the screen. The screen went black for a little moment after that.

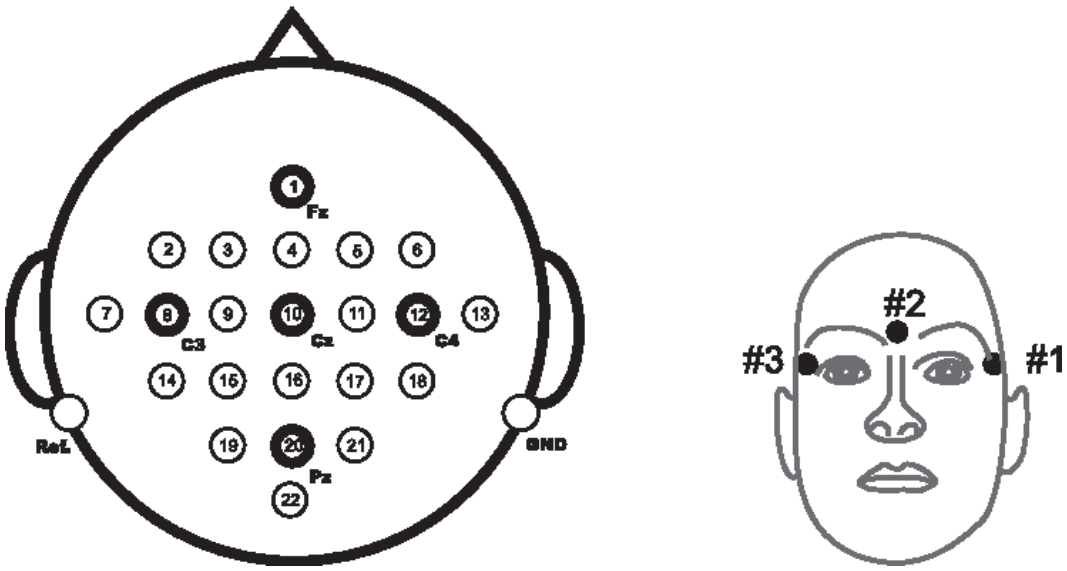

Figure 2: Left: The worldwide 10-20 system is shown via an electrode montage. Right: The three monopolar EOG channels' electrode montage. Brunner et al. [2008]

The EEG was recorded using twenty-two Ag/AgCl electrodes with 3.5 cm inter-electrode intervals (see monatage in Figure 2). All signals were monopolarly recorded, with the left mastoid acting as the reference and the right mastoid serving as the ground. The data was captured at 250 Hz and bandpass filtered between 0.5 and 100 Hz. The amplifier's sensitivity was set at 100 volts. To reduce line noise, an extra 50 Hz notch filter was activated.

3 Experiment Design

Depending on the outcomes of the algorithm, the machine learning algorithms are classified into various categories i.e. supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning algorithms. Classification is a supervised learning method, which means both the input and the desired output data are provided. Classification is a process of identifying the particular. each instance belongs to a class, which is indicated by the value of a special goal attribute or simply the class attribute. The goal attribute can take on categorical values, each of them corresponding to a class. Each example consists of two parts, namely a set of predictor attribute values and a goal attribute value. The former are used to predict the value of the latter. The predictor attributes should be relevant for predicting the class of an instance. In the classification task the set of examples being mined is divided into two mutually exclusive and exhaustive sets, called the training set and the test set. The classification process is divided into two phases: training, when a classification model is built from the training set, and testing, when the model is evaluated on the test set. In the training phase the algorithm has access to the values of both predictor attributes and the goal attribute for all examples of the training set, and it uses that information to build a classification model. This model represents classification knowledge – essentially, a relationship between predictor attribute values and classes – that allows the prediction of the class of an example given its predictor attribute values. For testing, the test set the class values of the examples is not shown. In the testing phase, only after a prediction is made is the algorithm allowed to see the actual class of the just classified example. One of the major goals of a classification algorithm is to maximize the predictive accuracy obtained by the classification model when classifying examples in the test set unseen during training.

3.1 Algorithm Comparison

Typical Machine Learning Algorithms (LDA, SVM) - Linear Discriminant Analysis is a method used for classification and reducing dimensionality. When presented with different categories

of data, LDA maximizes the distance between the categories' data points and minimizes the scatter of the data points withing the category. It then projects the data points onto linear axes (usually two) which presents the category data as nicely separated and easy to differentiate between. LDA can also be used in preprocessing data to reduce the number of features. In BCI, LDA is used to reduce the dimensionality and contrast between the extracted features of cerebral activity. A support vector machine (SVM) is another commonly used machine learning algorithm that aids in classification problems. A support vector machine focuses on linearly separable data. For example, in R2, an SVM would aim to create a line separating two groups of data, with one side of the line corresponding to a certain classification and the other side representing another classification. A good SVM will maximize the distance between points on either side of the linear boundary formed, thus finding the best "middle" separator of the data. More complicated SVMs can also be used on higher dimensional and/or multi-classification problems.

CNNs - Convolutional neural networks are a subcategory of deep learning and neural networks that involve the use of convolutional "layers". The given set of features, often derived from 2D images, is fed through the convolutional neural network. As images, or other data, are fed through the CNN's layers, the important features in determining the class label are weighted more heavily, causing the most attention to be paid to them in future classification. The most unique aspect of CNNs is the "dropout" layers they include, which involve each layer only focusing on a subset of the input features (pixels, etc.). This allows the CNN to accurately label future inputs without overfitting to the training data, allowing it to perform well on unseen data. CNNs also use pooling layers, which group pieces of feature information together to come up with a relationship between them (see Figure 3). This, like the dropout layer, allows CNNs to come up with meaningful relationships between the input data points (pixels, etc.) without overfitting too much to individual pieces of data. Thus, CNNs are able to focus on the most important parts of the input data and apply that knowledge to future inputs.

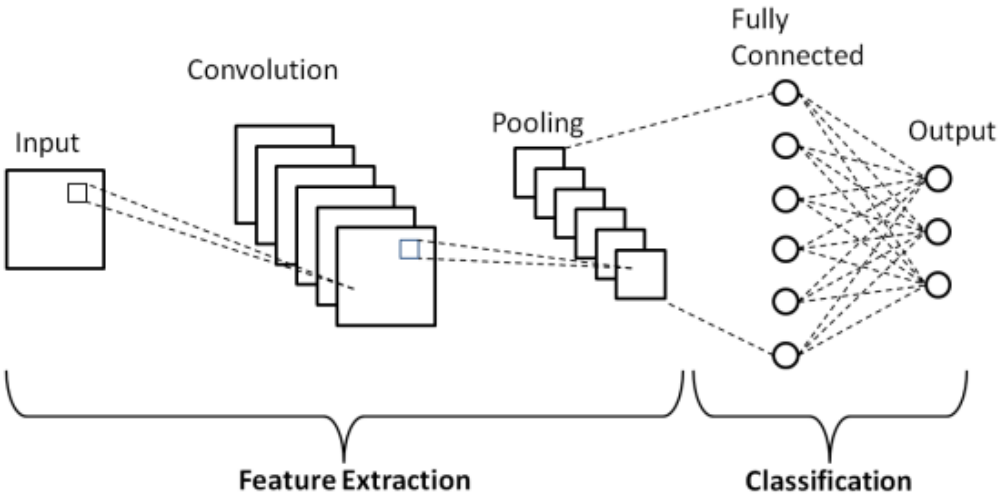

Figure 3: Depiction of CNN Layers Phung and Rhee [2019]

DBNs - Deep belief networks are another subcategory of deep learning and neural networks. DBNs are a generative model which produce the possible outputs for a given feature. DBNs can also be used for both supervised and unsupervised learning. What makes DBNs unique is their method of training, which involves first running through restricted boltzman machine layers (see Figure 4) and training to find important features for classification, as in typical neural networks, and then using "error back-propagation algorithms to fine-tune the parameters of the DBN" after the training is completed (science direct). This allows the DBN to accurately predict labels (or generate them) for future, unknown, inputs.

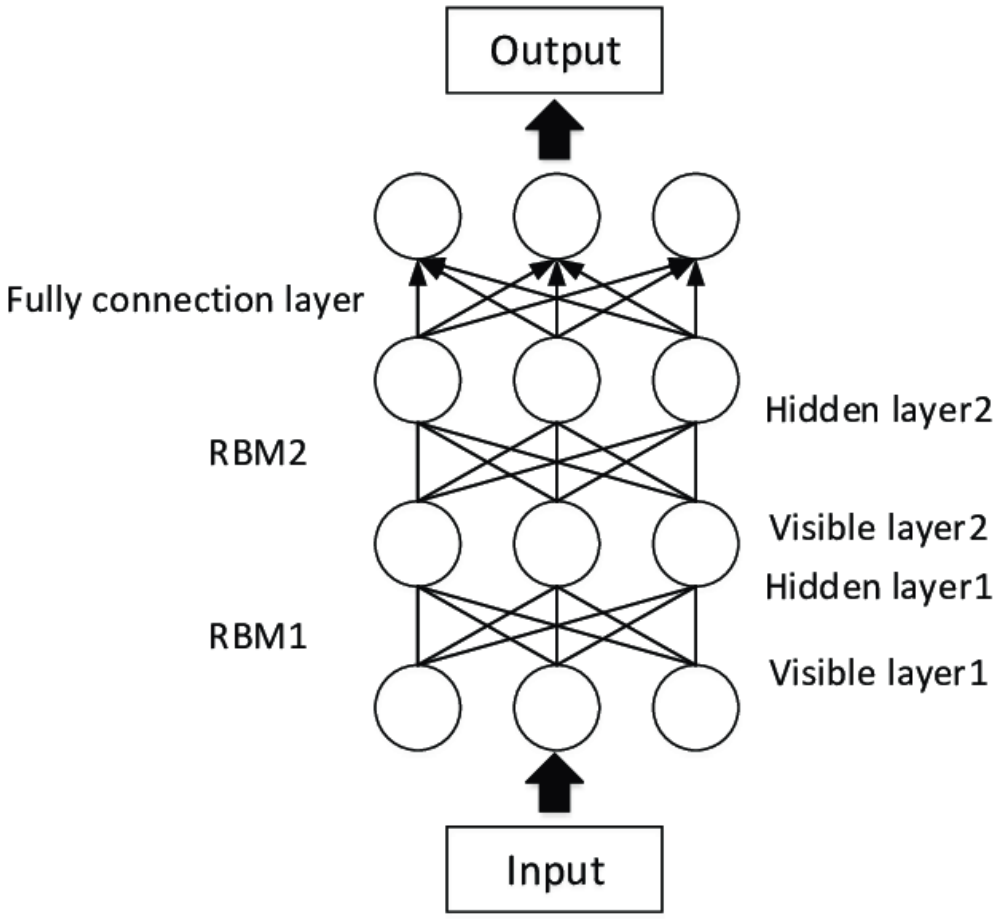

Figure 4: Depiction of DBN Layers Liu and Lang [2019]

3.2 Experiment Design

Our first experiments will be an attempt at replicating the results found in Hersche et al. [2018]. This will include using the same code and packages found within the github page associated with Hersche's paper, in order to classify the 2008 BCI competition motor imagery data (as Hersche did). Then, our neural network experiments are designed to replicate the efficiency and effectiveness already proven with CNN and DBN architectures in classification studies, however this time with the EEG results of human motor imagery tasks. In other words, these architectures will form the basis of two algorithms that will be trained and tested on our dataset under a supervised learning approach. Typically, classification algorithms work along the basis of a linear regression algorithm that slowly alters the input parameters of a prediction model (or formula) in order to achieve desirably classified outputs. In terms of our research, classification in this context refers to our models being able to take the results of our chosen motor imagery EEG data and make predictions for future EEG readings (new inputs) based on the sample. As noted in Craik et al. [2019], neural networks have gained large popularity in these sorts of classification tasks due to the fact they are also trained in a similar way, with inputs, weights, and desired results. Through our experiments, we want to demonstrate which of the CNN or DBN architectures is more accurate and faster in their predictions. To maintain our experimental validity, both architectures are going to be analyzed prior to their choosing, to make sure that they both can support similar numbers of classification inputs. A direct comparison of this kind can then be very informative as to which algorithm architecture may be more useful for future EEG classification research.

Firstly, we will test different CNNs and DBNs to find architectures that both can handle up to nine input features (the number of samples present in our current dataset). We will do sample runs with smaller input values to ensure the networks are functioning properly. Then, we will begin test trials where we train the two networks on our dataset. Lastly we will measure the accuracy of the networks' predictions, and also the runtime of the predictions of the course of the entire experiment. Measuring the accuracy of the models is the most important aspect of our classification experiment, as it lets us know which model is better at making predictions. The prediction accuracy is the focus for driving new methods of interpreting EEG data, which by specialized person alone is often a slow and meticulous process. This is why the speed of the predictions is important as well, as if the CNN and DBN architectures are accurate but give no relative prediction speed gains, their creation and use is not necessary.

4 Baselines

```
def run_CNN(self):
    start_train = time.time()
    if self.useCSP:
        w = generate_projection(self.train_data,self.train_label, self.NO_csp,self.filter_bank,self.time_windows)
    else:
        w = generate_eye(self.train_data,self.train_label,self.filter_bank,self.time_windows)

    feature_mat = extract_feature(self.train_data,w,self.filter_bank,self.time_windows) #numpy array, 219,11352
    eval_feature_mat = extract_feature(self.eval_data,w,self.filter_bank,self.time_windows)
    # feature_mat = feature_mat[np.newaxis, ...]
    # eval_feature_mat = eval_feature_mat[np.newaxis, ...]
    self.train_label = self.train_label
    self.eval_label = self.eval_label
    y_train = np.empty(219, dtype = float)
    y_test = np.empty(219, dtype = float)
    for i in range(219):
        np.append(y_train[i], self.train_label[i])
        np.append(y_test[i], self.eval_label[i])

        #y_test[0][i] = self.eval_label[i]
    print(y_train.shape, y_test.shape)
    print()
    print(feature_mat.shape, eval_feature_mat.shape)

    cnn = keras.models.Sequential()
    cnn.add(keras.layers.Conv1D(16, 3, activation = 'relu', input_shape = (219, 11352)))
    cnn.add(keras.layers.MaxPooling1D(2))#, padding = 'same'))
    cnn.add(keras.layers.Flatten())
    cnn.add(keras.layers.Dense(16, activation='relu'))
    cnn.add(keras.layers.Dense(10))
    cnn.summary()

    cnn.compile(optimizer='adam',
          loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
          metrics=['accuracy'])
    history = cnn.fit(feature_mat, y_train, epochs = 2, validation_data = (eval_feature_mat, y_test))
```

Figure 5: CNN model layers code added to Hersche et. al program.

For testing base level classification examples, we used the support vector machine (SVM), linear discriminant analysis (LDA), and HD models designed by Hersche et al. [2018]. Both the SVM and LDA models consist of 64 layers. The HD model is a more complex model and involves performing arithmetic on large input hyper vectors. The data is projected to a high dimensional Hamming Space, and the model is trained on this data.

Our CNN model was done in addition to the code by (Hersche 2018). Specifically, we added a program to similarly extract features from the IV-2a data and run a CNN on the data (Figure 5). Our CNN model aimed to use tensorflow to create a simple CNN using our prior knowledge and implementation tutorials of CNNs for Tensorflow Abadi et al. [2015]. Ultimately, we were unsuccessful in training and testing CNN on this data. Our problems stemmed from implementing

the algorithm, including formatting the data appropriately for inputting into the CNN, along with computational limitations given our personal computers' memory and time restrictions.

For the DBN model, two different implementations were attempted albertbup [2017] and AmanPriyanshu [2021]. There were complications with running the code from albertbup [2017] due to OS errors. The download path, success, and compatibility were all checked on our windows machine, meaning that the code may not be suitable with all environments. The code from AmanPriyanshu [2021] also was problematic, as it was built to import image classification files and not EEG signals. Even after alteration to the dataset input information the code could not work. Since this implementation used numerous python files for implementation, it was infeasible for changes to be made at that large a scale while maintaining the implementation's key components. Evidently, no viable results for either of these implementations were producible with the IV-2A motor imagery dataset.

5 Results

```
AVG:;    0.6148;          0.6283;          0.6795;
(9, 1, 4)
IV2a
```

Figure 6: Average accuracy results for run 1 of (hersche2018) replication of SVM, LDA, HD respectively.

```
AVG:;    0.6148;          0.6283;          0.7192;
(9, 1, 4)
IV2a
```

Figure 7: Average accuracy results for run 2 of (hersche2018) replication of SVM, LDA, HD respectively.

Table 1: Accuracies of LDA, SVM, and HD model in (hersche2018)

	SVM	LDA	HD
Prediction Accuracy	61.48	62.83	73.5

5.1 Typical Algorithm Performance

Figures 6 and 7 represent our reproduction of the results from (Hersche 2018) on two separate runs. Using the authors' provided instructions and code, we recreated the results with the given results. Table 1 provides the results for the same run provided by the original authors themselves. Notably, our results from our replication runs of SVM and LDA classification on the dataset prove to be the exact same (to four decimal places) as the average accuracies reported by the authors for these algorithms. As for the HD model, created by the authors of the Hersche 2018 paper, we achieved similar results to those reported in Table 1. However, there was more variance here than with the previous 2 models. The models differed by approximately 1.6% in terms of accuracy on the shown run, with a variation maximum of 5.55 percent between runs.

5.2 CNN / DBN Performance

Our CNN implementation is shown for reference in (Figure 5). This implementation attempt, along with our attempt to implement a deep belief network, was unsuccessful in producing meaningful results. Implementation, along with memory issues, were problematic for us, which meant that we had no meaningful results for our attempts to run a CNN or DBN (see Figure 8).

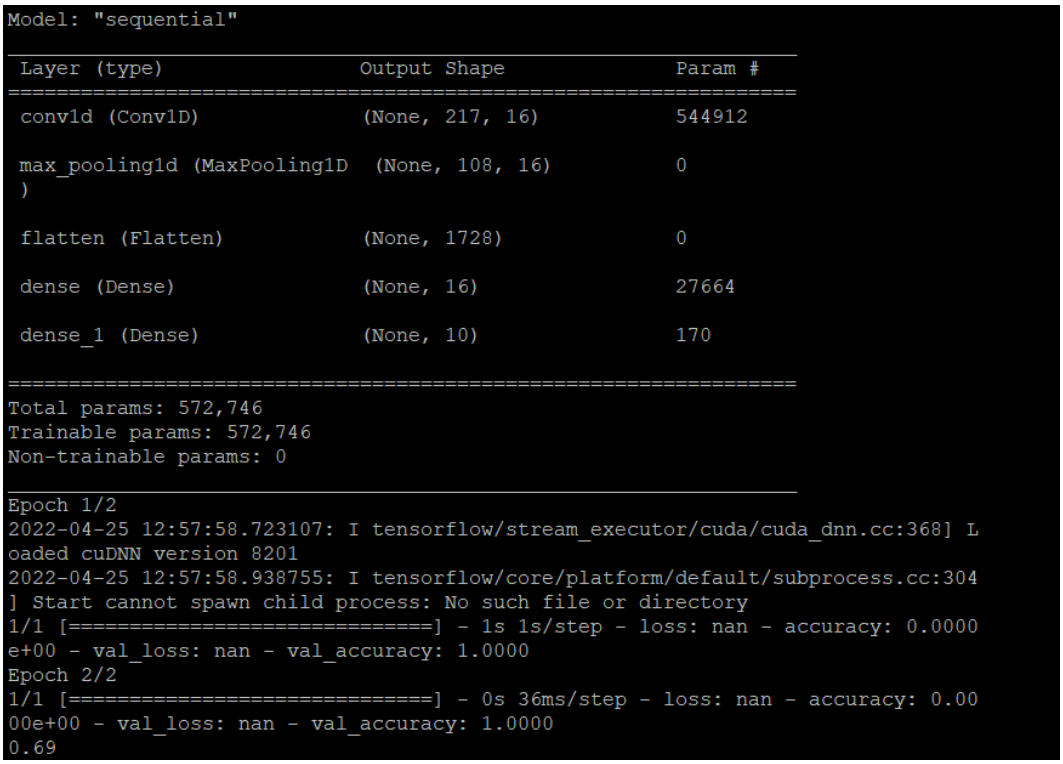

Figure 8: Erroneous Accuracy Results for our CNN Implementation.

6 Discussion

Table 2: Time spent on implementation of algorithms.

	SVM	LDA	HD	CNN	DBN
. Time in implementation (hours)	2	2	2	25	25

Our research is important because it demonstrates the ease of independent researchers in replicating standard machine learning models on motor imagery data, but the expected difficulties encountered in implementing viable neural network implementations. With this in mind, independent researcher such as ourselves can expect to face multiple forms of adversity with replicating the complex neural network applications to motor imagery data, such as machine constraints, research understanding time constraints, and complexities associated with using mixing numerous environment resources into one cohesive implementation. There are multiple tutorials for implementing standard machine learning algorithms on a multitude of different types of datasets and dataset files. These include datasets for motor imagery, emotion recognition, mental workload, seizure detection, sleep stage scoring, and event related potential tasks (Craik 2019). However, with motor imagery specifically there are relatively few research examples for DBNs, alluding that this method of classification might be outdated. The reason being that DBNs are clusters of restricted boltzman machines, which tend to be better optimized for image classification (Larochelle et al. 2012). Nevertheless, it could still be useful, as (An et al. 2014) found their DBN to be more successful than conventional SVM. Meanwhile, CNNs have been found to be useful for motor classification Tang et al. [2020], but still also require lots of time, skill, and experience for novel application to motor imagery studies (see Table 2 for time spent on each algorithm) For these reasons, CNNs comparison to DBNs for motor imagery classification is sparse, as identified in (Craik et al. 2019). Future research should take

these factors in account when attempting neural network implementations for motor imagery EEG classification.

7 Conclusion

This research project allowed us to replicate and verify the more simple algorithms' (LDA and SVM) performance on motor imagery data. We were able to verify the results provided for these standard machine learning models, along with Hersche et al.'s unique HD model. On this admittedly small dataset, it can be concluded that SVMs perform better than LDAs in terms of accuracy. However a binarized model, like the HD model created by Hersche et al., is able to perform the best.
In terms of our CNN and DBN model, our conclusion is that more work must be done relating to these two algorithms (and perhaps deep learning algorithms in general) when it comes to classifying EEG data. Specifically, future researchers should have experience implementing these algorithms from scratch onto new kinds of data, and be prepared for some of the issues we faced in memory and implementation. Future research into deep learning and EEG data classification should focus on these areas. Providing novel implementations of these commonly used algorithms will provide a starting base for further investigation of the use of deep learning algorithms on EEG classification tasks.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

albertbup. A python implementation of deep belief networks built upon numpy and tensorflow with scikit-learn compatibility, 2017. URL `https://github.com/albertbup/deep-belief-network`.

AmanPriyanshu. Deep-belief-networks-in-pytorch, 2021. URL `https://github.com/AmanPriyanshu/Deep-Belief-Networks-in-PyTorch`.

Xiu An, Deping Kuang, Xiaojiao Guo, Yilu Zhao, and Lianghua He. A deep learning method for classification of eeg data based on motor imagery, 2014.

C Brunner, R Leeb, GR Muller-Putz, A Schlogl, and BCI Competition. Graz data set a. *Institute for Knowledge Discovery, and Institute for HumanComputer Interfaces Graz University of Technology, Austria*, 2008.

Alexander Craik, Yongtian He, and Jose L Contreras-Vidal. Deep learning for electroencephalogram (eeg) classification tasks: a review. *Journal of neural engineering*, 16(3):031001, 2019.

Michael Hersche, José del R Millán, Luca Benini, and Abbas Rahimi. Exploring embedding methods in binary hyperdimensional computing: A case study for motor-imagery based brain-computer interfaces. *arXiv preprint arXiv:1812.05705*, 2018.

Hongyu Liu and Bo Lang. Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences*, 9:4396, 10 2019. doi: 10.3390/app9204396.

Fabien Lotte, Laurent Bougrain, Andrzej Cichocki, Maureen Clerc, Marco Congedo, Alain Rakotomamonjy, and Florian Yger. A review of classification algorithms for eeg-based brain–computer interfaces: a 10 year update. *Journal of neural engineering*, 15(3):031005, 2018.

Na Lu, Tengfei Li, Xiaodong Ren, and Hongyu Miao. A deep learning scheme for motor imagery classification based on restricted boltzmann machines. *IEEE transactions on neural systems and rehabilitation engineering*, 25(6):566–576, 2016.

Sangin Park, Jihyeon Ha, Da-Hye Kim, and Laehyun Kim. Improving motor imagery-based brain-computer interface performance based on sensory stimulation training: An approach focused on poorly performing users. *Frontiers in Neuroscience*, page 1526, 2021.

Phung and Rhee. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. *Applied Sciences*, 9:4500, 10 2019. doi: 10.3390/app9214500.

Quiles, Suay, Gemma Candela Garcia, Chio, Jiménez, and Leandro Kurogi. Low-cost robotic guide based on a motor imagery brain–computer interface for arm assisted rehabilitation. *International Journal of Environmental Research and Public Health*, 17:699, 01 2020. doi: 10.3390/ijerph17030699.

Xianlun Tang, Wei Li, Xingchen Li, Weichang Ma, and Xiaoyuan Dang. Motor imagery eeg recognition based on conditional optimization empirical mode decomposition and multi-scale convolutional neural network. *Expert Systems with Applications*, 149:113285, 2020.