Random Forest Classification for Training a Brain Computer Interface (BCI)

Abstract

Brain–computer interfaces (BCIs) aim at providing a non-muscular channel for sending commands to the external world using brain activity. Most existing BCIs detect specific mental activity in a so-called synchronous paradigm. Unlike synchronous systems that are operational at specific system-defined periods, self-paced interfaces have the advantage of being operational at all times. Existing BCI systems rely on feature extraction followed by a classification scheme to detect intentions from the brain signal. In this paper, we propose a novel self-paced BCI system that employs Random Forest (RF) algorithm for the classification of brain signal. Unlike the conventional BCI systems, the proposed system does not have a feature extraction step and tries to implicitly learn features from the raw brain signals. We also employ a Bayesian optimization framework to tune the parameters of the RF algorithm and the BCI system. The performances of the proposed novel BCI system and a grid search method are compared on dataset I of BCI competition IV. On the calibration data our optimization method outperformed the grid search method by at least 11% accuracy. As expected, the results of both methods on the evaluation dataset were not promising as the brain signal recordings in the calibration and evaluation sessions followed two different paradigms

1 Introduction

A Brain Computer Interface is a system that discovers patterns in a person's brain activity and relates them to the person's intention to control a device [1]. The objective of BCI is to convert the electrical signals generated by the brain to meaningful signals to control an external system. The most important application of BCI is to help the disabled people to control different devices.

Electroencephalography (EEG) is one of the methods of measuring electrical activity of the brain along the scalp. EEG recordings are taken from multiple positions from the surface of the scalp by putting sensors (electrodes) on the scalp. The main source of the brain signal is in the brain itself and therefore measuring this activity on the scalp introduces potentially unwanted noise. The signal to noise ratio of the EEG signal is low, i.e., signals have very low amplitude (i.e. about 10 to 100 micro volts) compared to the background noise. Therefore, detection of intentions from the measured brain signals is a challenging task and has been at the forefront of research [1].

BCI systems can be categorized into two different paradigms, namely synchronous and self-paced systems [2]. The majority of the research in BCI is concentrated on the synchronous systems. In synchronous BCIs, the subjects are limited to control the BCI output in system-defined periods and therefore, they cannot control the output in other times. On the other hand, in self-paced BCIs, the subjects have the option of controlling the system output whenever they intend to do so and the system is inactive in other times. The periods which the user is not controlling the system are called No-Control (NC) states. The response of the system to NC states would be neutral output. Compared to the synchronous BCI systems, designing self-paced BCIs is an extremely challenging task. The efforts so far have been promising but significant amount of work is needed to achieve a system that can be used in real life.

From another perspective BCI systems are categorized based on the Electrophysiological activity of the brain. Different electrophysiological activities of the brain produce differing patterns in the brain signal. For example, P300-based BCI systems operate based on the introduction of visual stimuli [3] and some other BCI systems work based on the sensory motor rhythm (SMR) activity. When the subjects try to move their limbs, a circumscribed desynchronization in their brain signal occurs. This desynchronization is referred to as event-related desynchronization (ERD). It is shown that motor imagery activity (i.e., imagined movements) generates movement related brain signal patterns similar to those that are generated by actual movements [1]. In SMR-based BCIs, which is the focus of this research, the goal is to detect ERD patterns related to real or imagined movements.

Over the past years, researchers have developed various signal processing algorithms to solve the BCI task. Feature extraction (feature engineering) is at the center of developments in the BCI community. The extracted features are then fed to a classifier to translate them to control commands. Publications during the past years have focused on a combination of feature engineering and classification to detect patterns from brain signals [1] and to the author's best knowledge, there is not any BCI system that works by applying learning algorithms on the raw brain signals.

So far, using a combination of cumbersome feature engineering and simple classification methods has not resulted in a satisfactory performance in the existing self-paced BCI systems and these methods are far from being suitable to use in the real life applications. On the other hand, in machine learning community, there is an ongoing research on learning algorithms which are capable of learning the features implicitly. Among these methods we can mention Deep Learning and Random Forest methods. [4] has reported several applications of deep learning in speech processing and computer vision. These methods applied the learning algorithm on the raw datasets without extracting features and outperformed the state of the art speech processing and computer vision algorithms. The focus of this research, in contrast to most of the literature in BCI community, is on applying the learning algorithm on raw brain signals in self-paced BCIs. In addition, we have also used a challenging dataset for which no publications with acceptable performances were found. It is also noteworthy to mention that in the past, we had applied the combination of several feature extraction and classification methods on this dataset; however none of those methods resulted in acceptable performance on evaluation dataset.

2 Materials and Methods

In this project, Random Forest classifier has been applied on the raw brain signals and Bayesian optimization is employed for tuning the parameters. In the following, we first describe the dataset used in this project. Then, we describe the random forest algorithm and Bayesian optimization in sections 2.2 and 2.3. Finally, in section 2.4 we describe our proposed method.

2.1 Dataset

A well-known publically available dataset, dataset I of BCI competition IV [5], is used to evaluate the methodology proposed in this paper. This dataset was recorded from 4 subjects performing motor imagery task (left hand, right hand, or foot imagery). Each subject participated in two sessions of brain signal recording. The first session, namely the calibration phase of recording is used for training the BCI system. The second session of signal recording is used for evaluation of the BCI system.

The data consists of 59 EEG channels (corresponding to 59 sensors) that were spread around the sensory motor area of the brain. In the calibration phase, each subject was assigned to perform two classes of motor imagery tasks from the left hand, right hand, or foot imagery movements. There were 200 trials of imagery movement that were balanced between two classes. The structure of each trial is illustrated in Figure 1. Each trial was 8 seconds (s) long in which at t=2s of each trial a visual cue on the computer screen was shown to the subject. Depending on the visual cue, the subjects were instructed to perform the assigned motor imagery task for 4s after observing the cue. The motor imagery tasks were interleaved with 4s intervals in which the subject should not have controlled the device, i.e., the subject should have been in the No-Control state.

The evaluation phase followed a different procedure. Instead of showing a visual cue to every subject, motor imagery tasks were cued by soft acoustic stimuli (words left, right, and foot) and the subject was instructed to perform the corresponding motor imagery task. As opposed to the calibration stage in which each trial was 8 seconds long, the length of the motor imagery intervals in the evaluation session varied between 1.5 and 8s. The NC intervals were also between 1.5 and 8s.

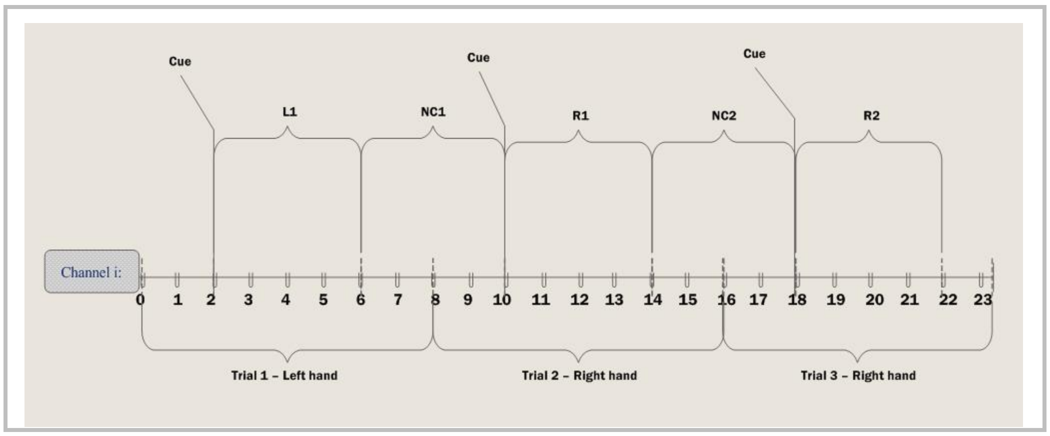

Figure 1: An example of sequences of trials in BCI calibration phase in which each trial is 8s. In the first trial subject performed imagery movement of left hand (L1) followed by a No-Control (NC1) interval of 4s. In the second trial the subject performed right hand imagery movement (R1) followed by a 4s long NC interval.

An important consideration about the EEG signal is that in two different sessions of brain signal recordings, the EEG signals of a subject may vary while performing the same task in both sessions. Studies have shown that session-to-session variability is an issue in BCI designs and therefore, most evaluation data are collected following the same protocols that were used for collecting calibration data. In the dataset at hand, the EEG data were collected in two different sessions with two different stimulation paradigms (visual versus acoustic cues). As a result, even though the data from the calibration and evaluation sessions represent motor imagery tasks, we believe that the data from both sessions have potentially different characteristics and as stated in [6] this is probably the reason that there are no published studies with acceptable performance on this dataset.

Another consideration in the brain signals used for BCI task is discarding some parts of the signal before and after each interval of the movement or NC. This is because of the fact that in transition from one brain state (e.g. NC) to another state (e.g. left hand imagery movement), the ERD may begin after different periods. As a result, it is better to discard the beginning and the ending part of each controlling interval. In section 2.4 of this report this procedure will be described in more details.

2.2 Random Forest Classifier

Random Forest [7] is an ensemble learning algorithm that is constructed by combining multiple decision trees at training time and produces a result that is the average of the output of individual trees. This powerful learning algorithm injects randomness into each tree in two ways. First it uses bootstrapping to sample from the original dataset. The second way of injecting randomness into the data is through selecting a subset of the features to split each node of the tree. As a result injection of randomness in the process of building Random

Forests, these classifiers are robust and have a good performance in cases where there are many outliers in data. Another consequence of injecting randomness in random forests is the ability to rank different features and acquiring a measure for feature importance.

Random Forest algorithm takes a bootstrap of the original training data to build each individual decision tree. Therefore, in each tree a subset of training data remains out of the bootstrap and can be used to measure the generalization power of the Random Forest algorithm. The part of the training data that remains out of the bootstrap are called the out of bag (OOB) samples. By keeping track of the predictions of each individual tree on its OOB samples, we can measure the prediction accuracy for the random forest. OOB score is almost identical to that obtained by K-fold cross validation [8]. Thereby, the accuracy on OOB samples (i.e. OOB score) can be utilized to tune parameters of the Random Forest algorithm. Another claim is that increasing number of trees does not cause the random forest to overfit [8]. Thereby, we set the number of trees to the maximum possible, based on our computational power. Some of the parameters of the random forest which can affect its performance significantly are the number of features to split a node, the maximum depth of the each individual tree, and the minimum number of samples in each leaf node of the tree. In this manuscript we call these parameters as RF parameters.

2.3 Bayesian Optimization

Bayesian Optimization [9] is a powerful algorithm that has outperformed state of the art global optimization algorithms on a number of challenging optimization benchmarks. This method is especially suitable for black box optimization (i.e. when we do not have an expression for the objective function); in which there is no information about the gradient of the objective function. Gaussian Processes are the most widely used tools for Bayesian Optimization. Assuming the objective function is sampled from a Gaussian process, the algorithm keeps a posterior distribution over the observed values of the objective function. To pick the parameters for the next experiment, the algorithm optimizes an acquisition function which is generated from the Gaussian process. A significant property of Bayesian Optimization is that it constructs a probabilistic model of the objective function. This algorithm proposes a new candidate point by integrating out the uncertainty. To perform Bayesian Optimization, two major choices should be made. First, the covariance function of the prior over the optimization function should be specified. The second choice for Bayesian Optimization is the choice of acquisition function.

For Bayesian Optimization in this research the method proposed in [10] is employed. The kernel function of the Gaussian prior was Matern 5/2 Kernel. This covariance function results in functions which are twice differentiable, an assumption that is used to perform quasi-newton optimization. The behavior of the prior function is governed by the choice of hyper-parameters. The most common approach to derive an appropriate value for hyper-parameters is to use a point estimate (e.g. Maximum Likelihood) of these values. However, in [10] a fully Bayesian approach is used to obtain a marginalized acquisition function. In other words, [10] uses a Monte Carlo estimation to evaluate the expected acquisition function over the posterior distribution of the hyper-parameters. Also, the acquisition function used here is the expected improvement (EI), which can be written in a closed form.

Another major contribution of [10] is its ability to perform parallel Bayesian Optimization. Assuming that N evaluations of the cost function have completed and J evaluations are pending. The algorithm Proposes a new candidate point based on the expected acquisition function over the possible results of the pending evaluations.

2.4 Proposed Method

Brain signal classification with Random Forests

To detect patterns from time series data (e.g., EEG signal), the data are usually divided into overlapping sliding windows (e.g., 2s windows with 96% overlap between consecutive windows). The features are then extracted from each window and a classifier is built using the extracted features. In case of multivariate signals (multiple input channels), features are extracted from the same window across all channels and the features are combined to build a single feature vector. For instance, if input signal has *n* channels and *m* features are extracted from each window, the final feature vector will be an $n \times m$ matrix. Eventually the output of a time series classification system is another time series for which the output at any time corresponds to the outcome of the classification task corresponding to the window that ends at that time.

Classifying the brain signals follows the same paradigm. Each interval of movement or NC is partitioned into overlapping windows of data. However, in contrast to the other works on BCI systems, in this project instead of extracting features from the brain signal, the raw signal is directly fed to the classifier. In other words, the final feature vector which is fed to the classifier (random forest in our case) is built by concatenating the windows of different channels. In the dataset at hand, we aim to detect NC states from motor imagery on a continuous basis. In other words, we intend to design a system that continuously classifies the input signal to either a movement imagery or NC state.

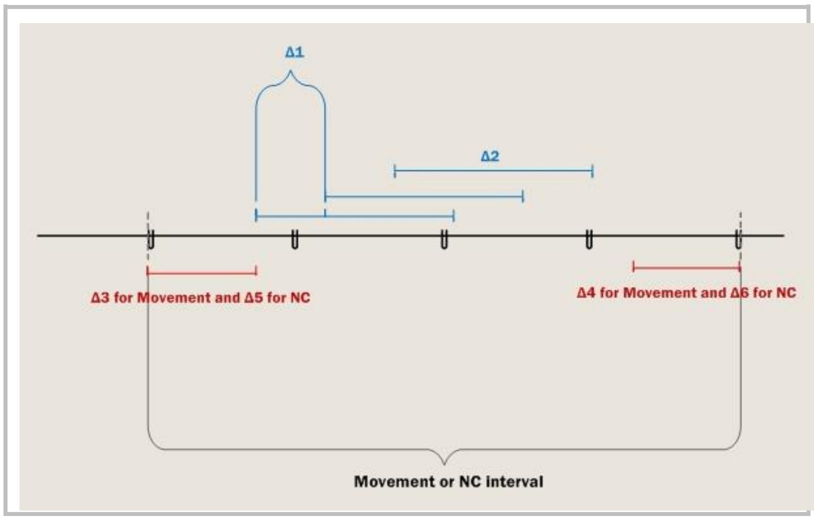

Figure 2: Demonstration of parameters in each interval

Figure 2 illustrates overlapping windows on a sample interval. In Figure 2, $\Delta 2$ corresponds to the size of each sliding window and $\Delta 1$ corresponds to the size of the overlap between two consecutive windows. In this figure, there are also four other parameters (i.e. $\Delta 3$, $\Delta 4$, $\Delta 5$ and $\Delta 6$) which correspond to the parts of each interval which would be discarded (as it was discussed in section 2.1). Note that partitioning each interval is done after discarding the samples from the beginning and the end of each trial. $\Delta 3$ seconds from the beginning movement interval and $\Delta 4$ seconds from the end of the movement interval are discarded. The counterparts of $\Delta 3$ and $\Delta 4$ for NC interval are $\Delta 5$ and $\Delta 6$. Choosing different values of $\Delta 1$, $\Delta 2$ … $\Delta 6$ for discarding unwanted EEG data is critical as the nature of NC and movement imagery are different and the exact times of NC and movement imagery are not known. Essentially, we need to make sure that the data that are fed to the classifier in fact represent the corresponding classes. Feeding data with inaccurate classes (labels) will lead to a poor performance. We call $\Delta 1$, $\Delta 2$ … $\Delta 6$ as BCI parameters in this manuscript. As a result of using raw brain signals the number of features for each training sample would be $59 \times |\Delta 2|$, where $|\Delta 2|$ is the size of the sliding window and 59 is the number of EEG channels. As, we are tuning the size of $\Delta 2$ to find the best sliding window size, the number of features will be variable. This will make tuning of joint parameters of BCI and RF difficult.

Bayesian optimization for tuning the joint parameters of RF and BCI system

As it was mentioned in the previous section, tuning the joint parameters of the BCI system and the classifier is an extremely difficult task. This optimization problem is a 9 dimensional problem (i.e. 6 BCI parameters for the BCI system and 3 RF parameters). One of parameters of the Random Forest which can affect its performance is the number of selected features to introduce the best split (we call this parameter $\Delta 7$). As in this research, the parameters of the BCI system and the RF parameters are optimized jointly the value of $\Delta 7$ is affected by the window size. That is to say, in the dataset used here, the maximum number of features that can be fed as a parameter to the random forest algorithm is $|\Delta 2| \times 59$. As a result, to train each Random Forest the number features to consider when looking for the best split in each node, should always be less than the maximum number of features (i.e. $\Delta 7 \leq |\Delta 2| \times 59$).

Typically Bayesian Optimization algorithm assumes that all dimensions of search space have bounds. In [10] the authors, has made an additional simplifying assumption. They assume that these bounds are axis-aligned, so the resulting search space is a hyper-rectangle. They then, generate a set of candidate point on this rectangle using the Sobol sequence generator. Sobol sequence is a smart way of generating candidates in the search space which is an example of low-discrepancy sequences. In this research, however, as the optimization has a constraint, generating the set of candidates on the hyper-rectangle is inappropriate.

[10] has two phases of generating candidate points. In the first phase it uses Sobol algorithm to generate a set of candidates on the unit hyper-rectangle. In the second phase, it first calculates the Expectation Improvement for the generated candidate points in the first phase. Then, it selects a set of optimal candidates (i.e. with respect to EI) and uses each of these selected candidates as an initial point for a gradient based optimization. Finally it takes the candidates generated in both phases and among those selects the candidate with best EI value. In this research, the candidate generation algorithm is changed to satisfy the constraint discussed above. In detail, in the first phase of generating the candidates, the candidates that do not satisfy the constraint (i.e. $\Delta 7 \leq |\Delta 2| \times 59$) are discarded. In the second phase, a constraint optimization algorithm is used to avoid the algorithm to optimize candidates back to the constraint violating space.

3 Results and Discussion

The task of classifying brain activity in self-paced BCI is really challenging. Due to the high rate of error in differentiating movement imagery task from each other and from the NC state, it is common in the BCI community to reduce the three class classification task to a movement detection problem (binary classification). In this case, the BCI system is designed to distinguish between the movement intervals and NC intervals. A binary BCI system is still a great advancement to assist paralyzed people to make binary decisions (e.g., yes or no). In the same way, we converted the three class classification in our dataset into a binary classification by considering all the movements (regardless of their type) as one class. Therefore, the goal here is to differentiate movements from the NC trials. As the evaluation data is recorded in a different session with a relatively different paradigm, the binary problem is still very challenging.

In this project, the performances of the proposed algorithm in section 2.4 and a simple grid search are compared. To perform the grid search, we simply set the parameters of the random forest as it was suggested by the inventor of Random Forest[1] , and performed the grid search only on the parameters of the BCI system (we refer to Bayesian Optimization for tuning the joint parameters of RF and BCI as BO-RF, and grid search for tuning parameters as GS-RF). For GS-RF evaluation, several combinations of $\Delta 3$, $\Delta 5 = \{50, 100, 150, 200\}$, and $\Delta 4$, $\Delta 6 = \{50, 100, 150\}$, and $\Delta 1 = \{75, 100, 125, 150\}$, and $\Delta 2 = \{20, 30, 40\}$, were examined. These combinations resulted in more than 100 evaluations for each subject.

The initial points to try for Bayesian Optimization are chosen at random. The algorithm then optimizes the objective function based on the evaluation of the initial points. As a result Bayesian optimization is sensitive to the random initialization. To demonstrate the required number of evaluations to find the best OOB score, BO-RF algorithm was repeated for three times with different initialization, and the mean and standard error is shown in Figure 3 (although, random initializations for three times is not enough to acquire reliable results, however, due to the limitations in computational infrastructure we were not able to get more

[1] For classification, the default value for number of features is $\left[\sqrt{p}\right]$ where p is the size of feature vector and the minimum node size is one.

results in the timeframe of this project). As Figure 3 shows, Bayesian optimization algorithm finds acceptable values for the objective function after very few iterations. Note that here the objective function is OOB score which we are maximizing.

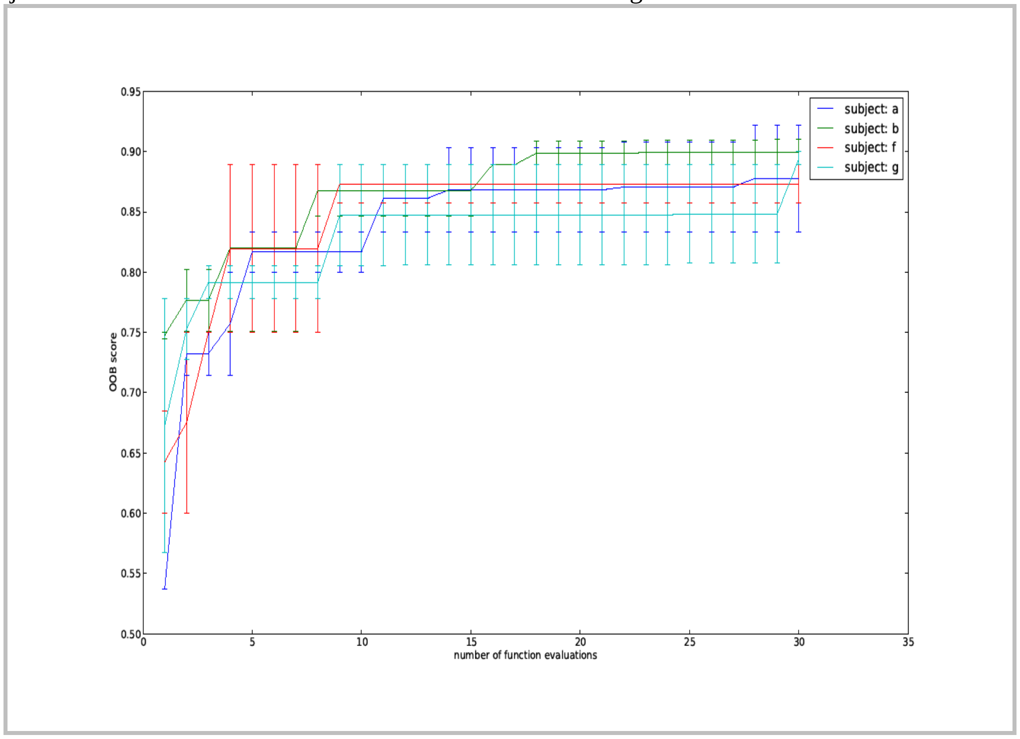

Figure 3: OOB score versus the number of function evaluations for the four subjects in the dataset. The results are the average of 3 different random initializations. In all experiments, J (number of parallel jobs) was set to 3.

The results of comparing BO-RF algorithm and the GS-RF algorithm based on the OOB score is shown Table 1. As Table 1 shows, BO-RF method outperformed GS-RF method with respect to the OOB score. The results of GS-RF are obtained after more than 100 evaluations of the objective function; however, the results of BO-RF are obtained after 45 evaluations.

Table 1: The optimal value of the objective function, which shows that our proposed approach (BO-RF) outperformed (GS-RF).

Subject	**A**	**B**	**G**	**F**
Best OOB Score BO-RF	0.922	0.910	0.888	0.900
Best OOB Score GS-RF	0.770	0.790	0.769	0.770

For each subject, a random forest is trained based on the optimal values of the BO-RF and GS-RF algorithms. The Accuracy of the optimal random forest for both algorithms on the evaluation data is shown in Table 2.

Table 2: The accuracy of BO-RF and GS-RF on the evaluation data

Subject	**A**	**B**	**G**	**F**
Accuracy BO-RF	%50	%58	%50	%52
Accuracy GS-RF	%50	%65	%50	%47

The results show that the performance of both methods significantly degrades on the evaluation dataset. As mentioned earlier (Section 1), we had applied several combinations of feature extraction and classification on this dataset; however, the performances of those systems on evaluation data were also poor. A considerable fact in EEG signals is that session-to-session transfer often causes considerable changes in the EEG signals. As in the dataset at hand, the recordings were done in two different sessions with two different protocols (i.e. visual stimuli for calibration data and acoustic stimuli for evaluation data), we were not optimistic about getting good results on the evaluation data. In [6] the authors also raised the same concern about the evaluation data of this dataset.

4 Conclusion and Future Work

In this research, Random Forest algorithm is employed, for the first time, to classify the raw brain signals. Unlike other works in the BCI literature, we eliminated the feature extraction phase in a BCI system. Applying Random Forests on raw brain signal can be regarded as a way of implicitly learning features from the data. We then used a Bayesian optimization framework to optimize the joint parameters of the Random Forest and BCI system. Results of our analysis showed that Bayesian Optimization outperformed the grid search algorithm. Bayesian Optimization was able to find better values of the objective function (i.e. OOB score) in fewer iterations.

In the future work, we are going to employ drop-out nets [11] to classify the raw brain signals. These learning algorithms are capable of performing multiple levels of feature learning/transformation on the data which might be useful in BCI systems.

The dataset used for this research was a self-paced dataset. The most important difficulty was that the evaluation data was recorded in different session with a different paradigm. As a result the performance of our methods was not promising on the evaluation data. Evaluation of the proposed BCI system on other dataset will also be the focus of our future work.

References

[1] A. Bashashati, M. Fatourechi, R. K. Ward, and G. E. Birch, “A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals,” *J. Neural Eng.*, vol. 4, no. 2, p. R32, Jun. 2007.

[2] A. Bashashati, R. K. Ward, and G. E. Birch, “Towards Development of a 3-State Self-Paced Brain-Computer Interface,” *Comput. Intell. Neurosci.*, vol. 2007, 2007.

[3] B. Z. Allison and J. A. Pineda, “ERPs evoked by different matrix sizes: implications for a brain computer interface (BCI) system,” *Ieee Trans. Neural Syst. Rehabil. Eng.*, vol. 11, no. 2, pp. 110–113, 2003.

[4] D. Yu and L. Deng, “Deep Learning and Its Applications to Signal and Information Processing [Exploratory DSP],” *Ieee Signal Process. Mag.*, vol. 28, no. 1, pp. 145–154, 2011.

[5] B. Blankertz, G. Dornhege, M. Krauledat, K.-R. Müller, and G. Curio, “The non-invasive Berlin Brain-Computer Interface: fast acquisition of effective performance in untrained subjects,” *Neuroimage*, vol. 37, no. 2, pp. 539–550, Aug. 2007.

[6] H. Zhang, C. Guan, K. K. Ang, and C. Wang, “BCI Competition IV - Data Set I: Learning Discriminative Patterns for Self-Paced EEG-Based Motor Imagery Detection,” *Front. Neurosci.*, vol. 6, Feb. 2012.

[7] A. Criminisi, J. Shotton, and E. Konukoglu, “Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning,” *Found Trends Comput Graph Vis*, vol. 7, no. 2–3, pp. 81–227, Feb. 2012.

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*, 2nd ed. 2009. Corr. 7th printing 2013. Springer, 2009.

[9] E. Brochu, V. M. Cora, and N. de Freitas, “A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning,” *arXiv:1012.2599*, Dec. 2010.

[10] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms,” *arXiv:1206.2944*, Jun. 2012.

[11] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv:1207.0580*, Jul. 2012.