Comparison of Algorithms for Detecting Hand Movement from EEG signals

Kristóf Várszegi
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary

Abstract—**In an ideal future limb function can be restored after suffering any kind of damage from an accident or a disease. The dream of fully functional prosthetic limbs depicts a great opportunity towards this vision. One of the main challenges is to create a control algorithm which processes physiological signals of the human body and moves the prosthetic limb according to the user's intentions. Brain-Computer Interfaces (BCIs) are popular subjects of research worldwide in the view of prosthesis control. Building mostly on EEG (electroencephalogram) signals, such systems must perform a challenging task – extracting control information from a signal with extremely low signal-to-noise ratio. Various digital signal processing and machine learning algorithms have been published in the past few years, but there is still no clear candidate for the title of the best one for e.g. detecting hand movement intention. In this paper various algorithms are compared on the example of detecting hand movement from EEG signals in real-time. The first goal of this paper is to present a couple of promising methods from other researchers in the past years, along with demonstrating the performance of other algorithms on a certain grasp-and-lift task. Another goal of the paper is to outline a future direction for the research of movement intent detection algorithms.**

Index Terms*—*BCI; prosthesis; EEG; signal processing; movement detection; movement intention; motion intent

I. The Basic EEG processing Pipeline in a real-time BCI

A real-time prosthetic limb would analyze the incoming EEG signals over the last time window having for example 1 second length. The processing algorithm would then indicate if there was movement intention in the window it just examined or there wasn't any (which makes this task a classification problem technically). In the first case the prosthetic limb would initiate a movement and continue based on the evaluation of the next time windows of the signal. This is different from the trial-based approach where a certain number of trials are averaged and a decision is made afterwards, like in the case of P300 spellers [1].

Various signal processing pipelines are utilized in experiments found in the literature, but the main stages are roughly the same [2, 3, 4, 5, 6]. The acquisition of the signal is not in the scope of this paper (see [7] for details on this topic), but the other steps of the pipeline are briefly summarized in this section. The transformations applied to the raw digital EEG signal can be divided into the following steps:

1. Artifact rejection
2. Time-domain filtering
3. Spatial filtering
4. Class feature generation
5. Classification

A. *Artifact rejection*

Raw EEG recordings usually contain short-term components, artifacts which do not originate in neural activity, but rather in some other source which is irrelevant to what we are looking for. Such components are usually higher in amplitude and different in shape than a clean signal. Artifacts may arise from eye movements or blinking for example. These should be removed from the signal as much as possible. In some environments the artifact removal is done by experts, sweeping plots of the signal manually. However, this is uncomfortably time-consuming, and not feasible for a real-time processing pipeline. Reducing the presence of artifacts can be done by automatic thresholding, meaning that outside certain previously defined values the signal is considered to be invalid, hence set to zero for those time intervals. Independent Component Analysis (ICA, [8]) is also often used to find artifacts in the signal, sometimes in combination with thresholding.

B. *Time-Domain Filtering*

Filtering in the time domain serves two purposes mainly. The first is to eliminate DC and high frequency noise and 50/60 Hz power line noise as much as possible. The second is to extract the frequency band of interest. Such extraction is not limited to only one band. The signals can be separated into different frequency bands (e.g. mu and beta), and within each one some features can be calculated later on. Such multi-band filters are called filter banks [5].

C. *Spatial Filtering*

978-1-5090-1897-0/16/$31.00 ©2016 IEEE

The EEG is recorded from multiple electrodes placed on the scalp. As we are looking for neural activity over various regions over the scalp, it is important what components do the signals from different electrodes share. From another point of view the different signal channels will contain redundancy and crosstalk as well, which needs to be reduced. The main purpose of spatial filtering is to extract the most relevant signal components from the array of channels. The most common spatial filtering techniques are the Independent Component Analysis [8], the Principal Component Analysis (PCA) [9], and the Common Spatial Pattern (CSP) method [4]. The CSP method performs better than PCA or ICA, but it requires much higher computational performance.

D. Class Feature Generation

Feature generation serves as a pre-processing stage before the classification step. Its purpose is to combine all the signal data over the temporal and spatial window of interest to a few values which effectively indicate the class of the given data instance. These feature values should be the ones by which the class of the data instance can be best determined.

A common feature is the amplitude of the oscillation within a given frequency band, e.g. the mu or the beta band in case of movement-related neural activity. The power of the signal within a selected frequency band averaged over a time period (e.g. 1 second) can also be a good feature. The covariance matrix of the multi-channel values over a time window was also tried with promising results [10], with the hypothetic advantage that it contains both per-channel and inter-channel information.

However, class features might not need to be generated in case of classification algorithms which have high number of parameters, such as Artificial Neural Networks (ANNs). ANNs may find the best suitable transformation for the input data [11] due to their highly adaptive nature.

E. Classification

The identification of the class to which the data instance belongs is the classification step. This typically happens on the basis of training with a set of data instances whose class is already known. The most common classifiers include the Linear Discriminant Analysis (LDA) [12], Logistic Regression (LR) [13], Support Vector Machine (SVM) [14], and the Artificial Neural Network [15] along with its architectural variants. The SVM and the ANN are more computationally intensive than the LDA and LR, but they have the potential to produce much better results.

II. Related Works

Feasible methods have been studied in [2], [3], and [4], which all present promising movement or movement intention detection methods. Two other methods, also targeting the same detection task, are summarized below as they contain rather complex signal processing pipelines. Some of the pipelines might have been applied to offline signal data in the experiments, but they are basically feasible for real-time processing.

Ang et al. developed a rather complex method for motor imagery classification [5]. The subjects of their experiment were two of the BCI Competition IV datasets. These datasets contain 4 classes of motor imagery EEG trials: left hand, right hand, foot, and tongue. The authors named their algorithm Filter Bank Common Spatial Pattern (FBCSP), which consists of four progressive stages of signal processing and machine learning. The first stage of the pipeline is a time-domain filter bank with Chebyshev Type II band-pass filters. The second is a CSP spatial filter. The third is a feature selection step with the MIBIF (Mutual Information-based Best Individual Feature) and MIRSR (Mutual Information-based Rough Set Reduction) algorithms. Last but not least the selected CSP features are classified using the Naive Bayesian Parzen Window method. The processing pipeline is depicted in Fig. 1. The measure of the classification performance in this experiment was the kappa value [16], for which the best results yielded 0.572 and 0.599, making this method promising for motor imagery classification.

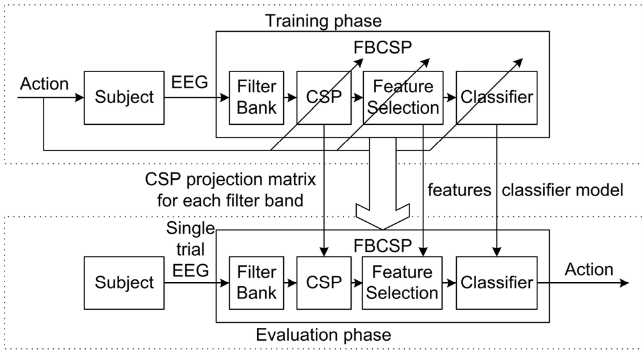

Fig. 1. The architecture of the filter bank common spatial pattern (FBCSP) algorithm for the training and evaluation phases [5].

Lei et al. proposed an improvement to the CSP method in [6]. They selected different CSP subspaces and their outputs were combined by majority voting, as depicted in Fig. 2. The underlying principle is that a classifier ensemble, a combination of similar classifiers is very likely to outperform a single classifier on its own. The discrimination of two different movements was studied from three healthy, right-handed participants, who performed motor imagery tasks during the EEG recording. The classification features are obtained by projecting the signal using spatial filters, and calculating the difference of log-power values coming from

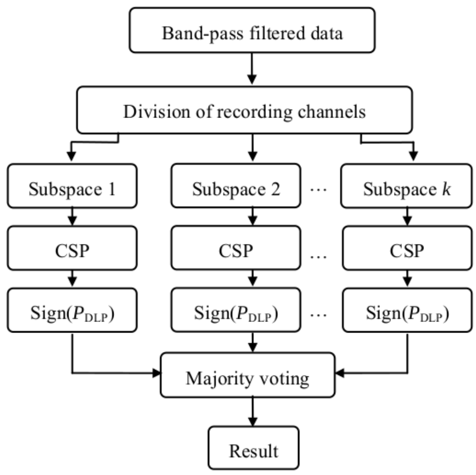

Fig. 2. The processing pipeline described in [6].

two tasks. In case of a single classifier, the sign of the resulting feature is interpreted as the predicted class. 10 feature values were computed with 10 different spatial filters and summed to represent the majority voting method (being aware that in the case of 10 features, the equality of votes may occur). The basis of decision between two classes was the sign of the final sum. According to the study, this CSP ensemble method outperformed LDA classifiers and SVMs, making it a promising method for BCI applications targeting movement restoration.

III. Comparison of Three Methods on a Grasp-and-Lift Task

In a previous study my goal was to design digital signal processing algorithms capable of detecting hand movement [17]. I designed three algorithms with a specific principle behind each. The algorithms were developed for a grasp-and lift detection task in an offline dataset. This was derived from the WAY-EEG-GAL dataset [18], and was pre-processed by Kaggle [19].

The dataset contains labeled EEG recordings from twelve healthy participants who performed grasp-and-lift series. The EEG recordings were carried out with 32 channels, and after decimation the resulting rate was 500 Hz. In total there were 12 subjects, 10 series of trials for each subject, and approximately 30 trials within each series. In each of the trials, the participants were cued to reach for the object, grasp it with the thumb and index finger, lift it and hold it for a couple of seconds, put it back on the support surface, release it, and, lastly, to return the hand to a designated rest position. These six events were labeled "HandStart" meaning the start of reaching for the object, "FirstDigitTouch" the moment of contact between the finger and the object, "BothStartLoadPhase" the appearance of the lifting force, "LiftOff" the object leaving the support surface, "Replace" the object being replaced on the platform, and "BothReleased" meaning the releasing of the object with all fingers.

The goal was to detect each event within ±150 milliseconds around the exact occurrence in time. A predicted probability of one for the entire window was expected, and zero outside it. The measure of the detection performance was the area under the receiver operating characteristic curve (AUROC). This was calculated for each event over all of the subjects, then the event-wise AUROC values were averaged over all 6 events.

A difficulty in achieving high detection results is that the classification decisions are made between classes with different population size. The events of interest only occupy a small percentage of the total time of the recording. Because of this, a seemingly low False Positive Rate (FPR), even 1-2% is still practically unacceptable, as it means a large number of false detections compared to the true events. The algorithms presented here create models for each subject individually. Creating a global model which instantly fits any subject is a much more difficult task. The three algorithms designed in my study are presented below.

A. Band-Pass Filter Bank Common Spatial Pattern with Logistic Regression

The principle behind this method is to exploit the CSP method's ability to enhance the variance of the signal's channels at the motion events of interest, and then to use this for effective feature generation. The processing pipeline of this algorithm is shown in Fig. 3.

The artifacts are rejected using an experimental threshold, and setting the signal value to zero when it exceeds the limits. The time-domain filter bank consists of two FIR band-pass filters. The spatial filtering is done using the CSP method, where the spatial filters are calculated separately for each event. The class features are normalized signal variances within a time window. Finally, Logistic Regression ensembles combining the estimated per-passband probabilities by arithmetic mean are used per event. This algorithm yielded an AUROC of 0.722 averaged over all of the subjects, with a standard deviation of 0.0533 over the subjects. Fig. 4. shows the ROC curves for the detection of each event.

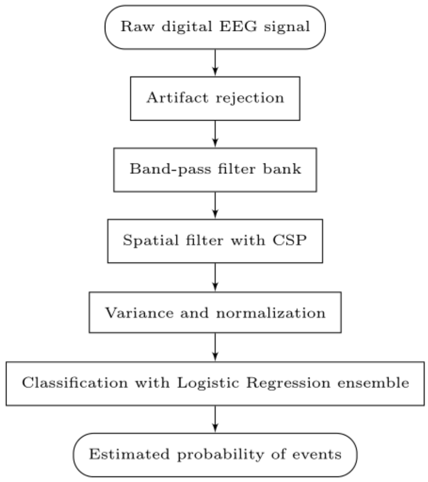

Fig. 3. The processing pipeline in the method Band-Pass Filter Bank Common Spatial Pattern with Logistic Regression.

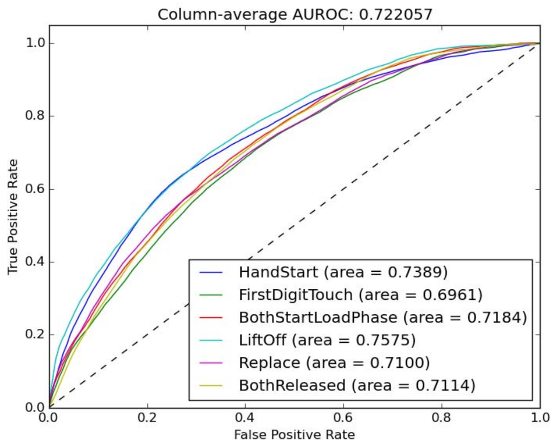

Fig. 4. The ROC graph of the results with the method Band-Pass Filter Bank Common Spatial Pattern with Logistic Regression.

B. Low-Pass Filter Bank with Logistic Regression

The principle behind this method is to build on the differences between various frequency ranges of the EEG signals. The low-pass filter bank utilized in this method concentrates on the low frequencies, below the beta range, with fewer filters in and above that. The processing pipeline of this method is shown in Fig. 5.

The artifact rejection is done by zeroing out the value of the signal when it exceeds an experimental threshold, similarly to the previous algorithm. The time-domain filtering is performed by a filter bank with fourteen IIR filters. The cutoff frequencies are densely defined in the filter bank below the beta band (11 filters), but there are filters with cutoff frequencies in the beta and gamma bands as well. There is no explicit spatial filtering performed in this method. The next step is a normalization using mean and standard deviation estimates calculated from the training data. The feature vector comprises the normalized raw and low-pass filtered signal values at the time point of interest. The classification is done with Logistic Regression per event.

This algorithm reached an averaged AUROC of 0.796 with standard deviation of 0.048 over the subjects. Fig. 6. shows the ROC curves for the detection of each event.

C. Normalization and a Convolutional Neural Network

The principle behind this method is to use a model with a high number of learned parameters which is – without significant pre-processing or a priori information – capable of finding the relevant features in the signal by itself. The processing pipeline of this algorithm is shown in Fig. 7.

The artifact rejection is done with a simple thresholding, similarly to the previous two methods. No explicit time-domain or spatial-domain filtering is performed. Each channel is normalized using the mean and standard deviation estimated from the training data. The normalized signal values over a 2-second time window, preceding the moment of interest, are taken as features. The signal values of every channel in this time window is fed to a Convolutional Neural Network (CNN). Only one CNN is used for the classification of all six motion events, per subject.

The CNN consists of altogether 6 layers: input, convolution, maxpool, dense, dense, and output. This CNN architecture is rather common, and is derived from [10] and [11]. Training the CNN took around 1600 seconds per subject. This algorithm reached an averaged AUROC of 0.829 with standard deviation of 0.0475 over the subjects. Fig. 8. shows the ROC curves for the detection of each event.

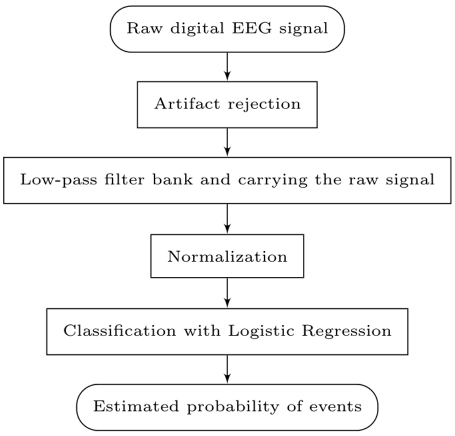

Fig. 5. The processing pipeline in the method Low-Pass Filter Bank with Logistic Regression.

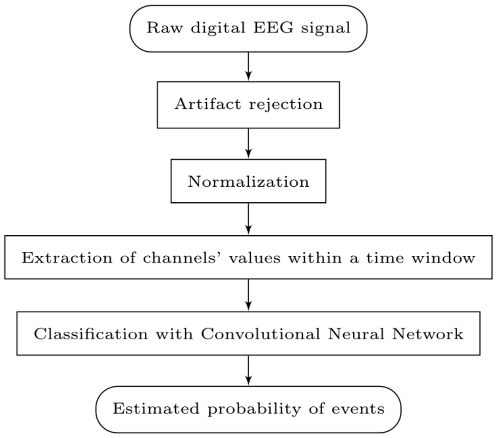

Fig. 7. The processing pipeline in the method Low-Pass Filter Bank with Logistic Regression.

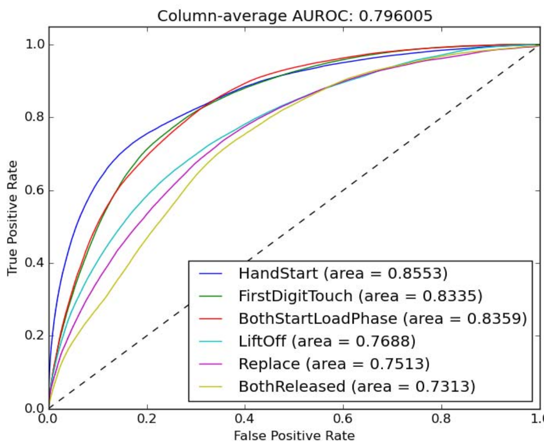

Fig. 6. The ROC graph of the results with the method Low-Pass Filter Bank with Logistic Regression.

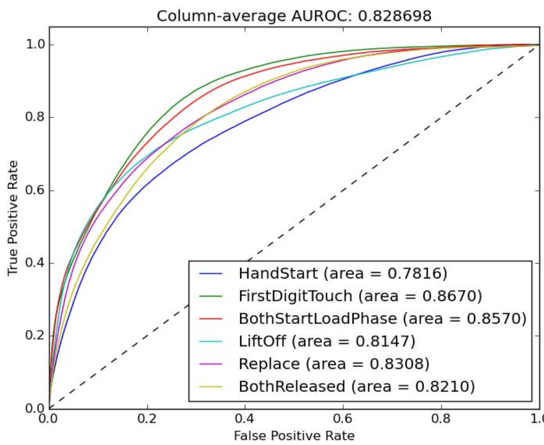

Fig. 8. The ROC graph of the results with the method Normalization and a Convolutional Neural Network.

IV. Comparison and Conclusions

The three algorithms in the previous chapter were presented because they all build on different principles and assumptions regarding the EEG signals. They provide information about the feasibility of various choices for the elements of the signal processing pipeline. All three algorithms were compared in terms of both detection performance and real-time feasibility. Table I. summarizes the performance indicators of the algorithms. The real-time applicability of an algorithm means that it is able to output the estimated probabilities for each event until the next time sample arrives into the pipeline. This condition stands for all three algorithms. The highest latency occurred at the algorithm which used a CNN, but even this one is able to keep up with the incoming stream. (The length of the test recordings is 104.8 minutes and even the slowest algorithm could process it in 60 minutes, which means that even if it got behind, it could catch up.)

TABLE I. Comparison of Detection Performance

Algorithm	AUROC mean	AUROC st. dev.	Processing time[a]
III.A	0.722	0.053	25 mins
III.B	0.796	0.048	4 mins
III.C	0.829	0.0475	60 mins

[a] Time took to process the 104.8 minutes long test sequence.

There are a couple of ways to improve the computational efficiency of the algorithms. The latency issue could be somewhat resolved using a parallel computational architecture, in which another processing pipeline can be initiated while the previous one hasn't finished yet. Furthermore, it might not be necessary to evaluate the last time window every time a new sample arrives. It could be satisfactory to run the pipeline only ten or twenty times per second, as these rates are probably still high enough to appear fluent to the user.

Looking at the results of the algorithms it is visible that the analytical pre-processing steps such as filter banks or CSP increase the detection accuracy. However, it is also visible that an algorithm with higher number of learned parameters is able to perform better. This brings us to the main conclusion of this paper, which is that a major emphasis should be put on the research of ANN architectures for solving the movement detection and later the movement intent detection task. (Along with this, it would be needed to research the parameter selection for pre-processing steps such as time-domain filter banks, to find how it would improve an ANN's performance to a significant extent.) Apart from the basic Multi-Layer Perceptron (MLP), many kinds of Neural Network variants exist which have the potential to perform well in detecting movement of movement intention from EEG signals, such as CNNs, Recurrent Neural Networks (RNNs), and Long Short-Term Memory networks (LSTMs) [20], therefore these should be studied in our context. The best results on the WAY-EEG-GAL dataset were achieved by CNNs [10, 11], approaching a magnificent AUROC value of 0.98, which also makes the Neural Network an extremely promising choice for a classifier.

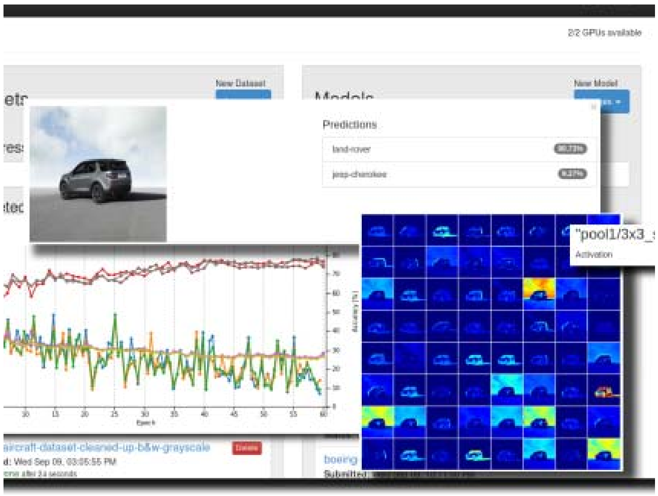

Fig. 9. An example of visualizing the activations inside a CNN in an image recognition task. [21]

The main issue with ANN-s is that the actual transformations they apply to the data are unknown. This limits our ability to extrapolate and estimate if a given ANN architecture which proved successful in one situation would perform well in another tasks as well. However, looking inside an ANN is not impossible. In image processing, where ANN-s are popular, it is a common practice to examine the outputs of the middle layers of the network and study what kind of signal features do they amplify. It is a difficulty that in an EEG signal processing task the inter-layer activations are not images, hence a visual evaluation is not enough. Time- or frequency-domain analysis of these activations could help to overcome this. The purpose of such analysis would be to gain knowledge about the effectiveness of various elements of an ANN architecture, providing clearer directions for designing an optimal one in a given movement detection or movement intent detection task.

ANN-s with more layers usually provide better results, and the current trend in data science where ANN-s are applied is that more and more complex network structures are utilized. However, large networks require high computational capacity. Many of the best processing pipelines for the WAY-EEG-GAL dataset ran on high performance General Purpose Graphical Processing Unit (GPGPU) clusters. To support this trend in real-time EEG signal processing prosthetic devices need to be fitted with high performance parallel processors, which currently seems unfeasible because of the size of such devices. Another option would be to move the processing pipeline computation to a separate computer (placed in a server room for example). The prosthesis could direct the signal stream wirelessly to the computer. The computer could process the newly arrived data samples, and then it could stream the output of processing pipeline back to the prosthesis. This way the user could move freely as far as the wireless connection allows him to, and the size of the processing computer would become a less important concern.

V. SUMMARY

This paper presents some of the most relevant processing methods for detecting hand movement from EEG signals and indicates a direction for future research. Two processing pipelines were described which are still among of the best performers, and three algorithms developed in a previous study were presented and evaluated further. Methods incorporating classifiers with a high number of learnable parameters seem to outperform strictly analytically designed ones. However, traditional pre-processing steps such as time-domain filtering and spatial filtering still might improve the detection accuracy of processing pipelines which employ e.g. ANNs. The final conclusion is that the power of ANNs is not something we can throw away because they seem like black boxes. We rather need to look inside them, develop methods to analyze their internal activations, figure out the behavior of their architectural elements, and create a knowledge basis for conscious ANN design to handle EEG signal processing tasks in BCI-s targeting movement restoration.

REFERENCES

[1] Dean J Krusienski, E.W. Sellers, F. Cabestaing, S. Bayoudh, D.J. Mcfarland et al., “A comparison of classification techniques for the P300 speller,” Journal of Neural Engineering, vol. 3, pp. 299-305, 2006.

[2] G. Pfurtscheller et al., “EEG-Based Asynchronous BCI Controls Functional Electrical Stimulation in a Tetraplegic Patient,” EURASIP Journal on Applied Signal Processing, vol. 19, pp. 3152–3155, 2005.

[3] J.R. Wolpaw, D.J. McFarland, “Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans,” Proceedings of the National Academy of Sciences of the United States of America vol. 101, pp. 17849–17854, 2004.

[4] J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg, “Designing optimal spatial filters for single-trial EEG classification in a movement task,” Clinical Neurophysiology vol. 110, pp. 787–798, 1999.

[5] K.K. Ang, Z.Y. Chin, C. Wang, C. Guan, and H.A. Zhang, “Filter bank common spatial pattern algorithm on BCI competition IV Datasets 2a and 2b,” Frontiers in Neuroscience, 6.39., 2012.

[6] X. Lei, P. Yang, P. Xu, T.-J. Liu, and D.-Z. Yao, “Common Spatial Pattern Ensemble Classifier and Its Application in Brain-Computer Interface,” Journal of Electronic Science and Technology of China, 7.1., 2009.

[7] M. Teplan, “Fundamentals of EEG Measurement,” Measurement Science Review, vol. 2, section 2, pp. 1-11, 2002.

[8] A. Hyvärinen and E. Oja, “Independent Component Analysis: Algorithms and Applications,” Neural Networks vol. 13, pp. 411–430, 2000.

[9] K. Pearson, “On Lines and Planes of Closest Fit to Systems of Points in Space,” Philosophical Magazine vol. 2, pp. 559–572, 1901.

[10] A. Barachant and R. Cycon, ”Code and documentation for the winning solution in the Grasp-and-Lift EEG Detection challenge,” 2015. https://github.com/alexandrebarachant/Grasp-and-lift-EEG-challenge. [Accessed: 2 January 2016]

[11] blog.kaggle.com, “Grasp-and-Lift EEG Detection Winners' Interview: 3rd place, Team HEDJ,” 2015. http://blog.kaggle.com/2015/10/05/grasp-and-lift-eeg-detection-winners-interview-3rd-place-team-hedj. [Accessed: 2 January 2016]

[12] K. Teknomo, “Linear Discriminant Analysis Tutorial,” 2015. http://people.revoledu.com/kardi/tutorial/LDA/LDA.html. [Accessed: 2 January 2016]

[13] C.R. Shalizi, “Advanced Data Analysis from an Elementary Point of View,” Carnegie Mellon University, 2012.

[14] C.J.C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” Data Mining and Knowledge Discovery vol. 2, pp. 121–167, 1998.

[15] M.A. Nielsen, “Neural Networks and Deep Learning,” Determination Press, 2015.

[16] J. Cohen, “A coefficient of agreement for nominal scales,” Educational and Psychological Measurement vol. 20, pp. 37–46, 1960.

[17] K. Várszegi, “Detecting Hand Motions from EEG Recordings,” B.U.T.E. Scientific Student Conference, 2015.

[18] M.D. Luciw, E. Jarocka, and B.B. Edin. “Multi-channel EEG recordings during 3,936 grasp and lift trials with varying weight and friction,” Scientific Data, vol. 1, p. 140047, 2014.

[19] Kaggle Inc., “Grasp-and-Lift EEG Detection,” 2015. https://www.kaggle.com/c/grasp-and-lift-eeg-detection/data [Accessed: 21 October 2015]

[20] S. Hochreiter, J. Schmidhuber, “Long short-term memory,” Neural Computation vol. 9, pp. 1735–1780, 1997.

[21] NVIDIA Corporation, “NVIDIA DIGITS Interactive Deep Learning GPU Training System” 2016. https://developer.nvidia.com/digits [Accessed: 15 April 2016]