An Uncued Brain-Computer Interface Using Reservoir Computing

Pieter-Jan Kindermans, Pieter Buteneers, David Verstraeten, Benjamin Schrauwen
Ghent University
Electronics and Information Systems Department
B-9000 Gent, Belgium
PieterJan.Kindermans@UGent.be

Abstract

Brain-Computer Interfaces are an important and promising avenue for possible next-generation assistive devices. In this article, we show how Reservoir Computing – a computationally efficient way of training recurrent neural networks – combined with a novel feature selection algorithm based on Common Spatial Patterns can be used to drastically improve performance in an uncued motor imagery based Brain-Computer Interface (BCI). The objective of this BCI is to label each sample of EEG data as either motor imagery class 1 (e.g. left hand), motor imagery class 2 (e.g. right hand) or a rest state (i.e., no motor imagery). When comparing the results of the proposed method with the results from the BCI Competition IV (where this dataset was introduced), it turns out that the proposed method outperforms the winner of the competition.

1 Introduction

A Brain-Computer Interface (BCI) or Brain-Machine Interface (BMI) is a direct communication method between man and machine without the need of any muscular activity. The computer analyzes brain signals and extracts relevant information w.r.t. a certain operation or task the subject is trying to accomplish, e.g. movement of an on-screen cursor. This way, a direct communication channel from the brain to the machine is created without the need for a mechanical device.

Obviously, the development of such BCIs would enable sufferers of many types of disabilities to communicate with their environment or to operate certain devices without direct mechanical interaction. This increases their potential freedom to operate and means of communication substantially, thus increasing their quality of life and decreasing the burden on health professionals and families. Several BCI have been investigated, such as the P300-speller which allows a user to spell words by focusing on consecutive letters placed on a grid, or the motor imagery BCI used for cursor control or spelling.

In this contribution we focus on an Electro-EncephaloGram (EEG) based uncued motor imagery BCI using Reservoir Computing (RC). We used a motor imagery dataset from BCI Competition IV [2]. Most motor imagery experiments are cued or synchronous experiments, which means that the classification algorithm has information of the start and end of each trial. Thus, these types of BCI reduce to a (possibly multi-class) classification of a series of sensor values limited in time. In contrast, for the application considered here the cues that indicate the start or end of a trial are omitted. This means that the BCI has to discriminate between two motor imagery classes and an idle state in an online manner instead of trial by trial. The addition of the rest state and the loss of cues makes the problem more complex but obviously more useable in practice.

An important aspect of a BCI is the discriminating neurophysiological phenomenon between the different possible actions, in this case Event Related Desynchronization (ERD) [10, 11]. When a

person imagines moving a limb, an attenuation in the EEG signal is noticeable at specific frequencies and in specific regions of the sensorimotor cortex. Left (right) hand motor imagery leads to desynchronization in the right (left) hand side of the sensorimotor cortex. The center of the sensorimotor cortex is activated during foot imagery. However, there can be quite some variation in the exact frequency bands and spatial locations of these ERDs from subject to subject, which is why a trainable preprocessing and classification method is desirable.

The features that were used in this work are based on the method of Common Spatial Patterns (CSP) [3, 7]. The CSP algorithm computes a set of spatial filters that are optimized to discriminate between two motor imagery classes. It has been shown that the CSP method can be improved by using a more complicated version or by combining it with subject-specific feature selection algorithms [1, 4, 9]. Ideas from different feature selection algorithms were combined in a novel feature selection algorithm that produced better spatial and frequency specific features.

The improved feature extraction method was used as a preprocessing step for a Reservoir Computing (RC)-based classifier [12]. Reservoir Computing is an umbrella-term for a set of methods for efficient training of recurrent neural networks. In this contribution we will use the Echo State Network (ESN) flavor of networks. These networks are recurrently connected with random, globally scaled weights.

In Section 2 we will discuss the methods used in this contribution. We will first briefly introduce the CSP method, our proposed feature selection algorithm and finally Reservoir Computing. In Section 3 we will present and discuss the different evaluation methods, experiments and results. We summarize and conclude in Section 4.

2 Methods

2.1 Common Spatial Patterns

The CSP algorithm [3, 7, 8] operates on EEG signals and searches for spatial filters that are optimized to discriminate between two motor imagery classes. The CSP-generated filters linearly combine the original signals to generate a new signal, which makes it a computationally attractive method once it is trained.

Each spatial filter maximizes the variance of the filtered signal during one condition (e.g. left hand movement) and minimizes the variance of the other condition (e.g. right hand movement). This minimization is related to the effects of ERD.

The construction of CSP filters is based on the simultaneous diagonalization of the covariance matrices of the two classes by generalized eigenvector decomposition:

$$\begin{aligned} V^{-1}\Sigma_1 V &= D, \\ V^{-1}\Sigma_2 V &= 1 - D, \\ V^{-1}(\Sigma_1 + \Sigma_2)V &= I, \end{aligned}$$

where Σ_1 and Σ_2 are the estimates of the covariance matrices of the input signals during class 1 and class 2 respectively, V is the matrix that contains a set of eigenvectors, D is a diagonal matrix that contains the corresponding eigenvalues and I is the identity matrix. Each spatial filter is associated with an eigenvector in V and an eigenvalue in D. If the eigenvalue for a filter is d, then the variance during class 1 equals d and the variance during class 2 equals 1-d. A filter with a very high or very low eigenvalue generates new signals with great differences in variance between the two classes.

The CSP computation starts with the computation of the covariance matrices of both classes

$$\begin{aligned} \Sigma_i &= \frac{E_i E_i^T}{\text{trace}(E_i E_i^T)}, \\ \Sigma &= \Sigma_1 + \Sigma_2, \end{aligned}$$

where E_i is the input signal for class i, Σ_i is the covariance matrix of class i and Σ is the total covariance matrix. The total covariance matrix can be factored in its eigenvectors

$$\Sigma = U^{-1} B U,$$

the matrix U contains the normalized eigenvectors and B the eigenvalues. The next step is the computation of the whitening transform

$$W = B^{-\frac{1}{2}}U.$$

The covariance matrix of the whitened input signals can be diagonalized as shown above:

$$S_1 = W\Sigma_1 W^T = VDV^{-1} \quad \text{and} \quad S_2 = W\Sigma_2 W^T = V(1-D)V^{-1}.$$

The CSP projection matrix is then given by:

$$P = W^T V.$$

A single CSP filter is just a column in this projection matrix. The CSP filtered version of the input is

$$E_{csp} = P^T E.$$

The CSP algorithm is commonly used in a more complete feature extraction system [8], consisting of several steps:

- spectral filtering of every EEG channel (typically bandpass 8-30 Hz),
- spatial filtering by using the CSP filters with the 3 highest and 3 lowest eigenvalues. These filters should maximize the variance differences between the classes,
- computation of the log-variance of a window of filtered EEG signals:

$$feature_i = log\left(\frac{\sigma_i^2}{\sum_{j=1}^{n}\sigma_j^2}\right),$$

  where σ_i stands for the variance of the signal generated by CSP filter i, n is the number of CSP filters.

2.2 Proposed Feature Selection Algorithm

The proposed algorithm is in part inspired by Filter Bank CSP (FBCSP) [1]. The goal is to find subject-specific passbands and spatial filters to generate better features. The algorithm works in three steps that are executed twice: once to find the best filter band and associated spatial filters within the frequency range 6-16 Hz and once to find the best one in 16-26 Hz. The algorithm is also illustrated in Figure 1.

1. In the first step we apply CSP to extract features. We then use the following scoring function to find the best spatial filters:

$$score_i = MI(m_1; m_2) + \max(MI(m_1; rest), MI(rest; m_2)),$$

   where i is the filter, $MI(a; b)$ denotes the mutual information between the input features generated by this filter and the class label when only class a and class b are considered, m_1 stands for motor imagery class 1, m_2 for motor imagery class 2 and rest for the rest class. The mutual information between the input X and the class labels Y is defined as

$$I(X;Y) = \int_X \int_Y p(x,y)\log\left(\frac{p(x,y)}{p_X(x)p_Y(y)}\right)dxdy.$$

2. Next we apply an overlapping filterbank (5 Hz bandwidth, 1 Hz overlap) before the best 10 CSP filters from the previous step are used and the log-variance is computed. This time we use the scoring function to determine the most informative frequency band.
3. The third step uses the best frequency band from the previous step to filter the EEG signal and recompute the CSP filters. These new CSP filters are then ranked again. The filter selection is dependent on the used frequency range.

In most experiments the differences between the filters with highest mutual information and the remaining filters were large. For the filters in the lower frequency range the best filters from each side of the eigenvalue spectrum are used. There was one subject (subject f) where the scores of 4

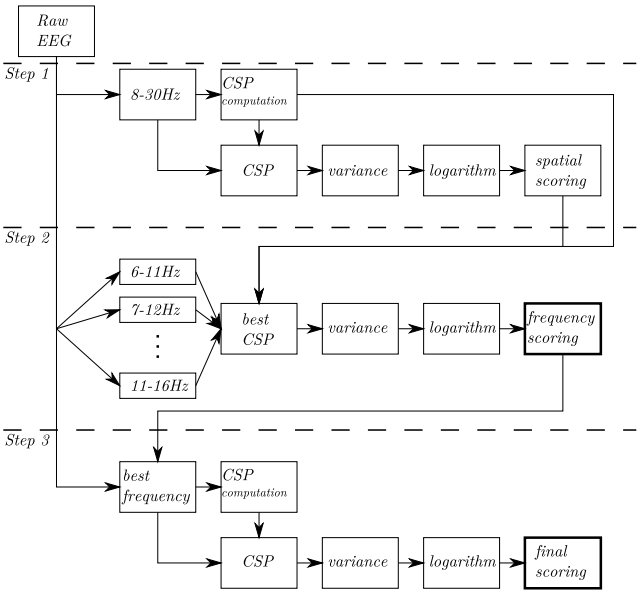

Figure 1: Schematic overview of the feature selection algorithm for the 6-16Hz range. The same process is repeated for the 16-26Hz range.

spatial filters (2 on each side of the eigenvalue spectrum) was very similar. There was no indication that one set would significantly outperform the other. In that case we choose to keep both sets. The process described above is repeated for the high frequency range (16-26Hz), but in this case a mutual information threshold of 0.3 was used. We added this threshold because most of the ERD effects are noticeable in the lower frequency range. Adding a feature from the higher frequency band is only useful if that feature is very informative. The threshold itself was empirically determined.

There are three main differences between this technique and the FBCSP method. The first difference is the iterative approach. The FBCSP method computes a total of $B \times N$ spatial filters, where B is the number of filter banks and N is the number of EEG channels. Our approach only computes $2 \times 2 \times N$ spatial filters (the leading factor 2 is because we run the algorithm twice). We use overlapping filter banks in contrast to non overlapping filter banks in FBCSP. When using an overlapping filter bank, it is impossible that the ERD is most noticeable at the boundary between two filters. We can always find a filter that captures the ERD and can compensate for shifts in ERD frequency. A final point of contrast is that we recompute the CSP filters based on the best scoring frequency.

2.3 Reservoir Computing

As a classification back-end, we used Reservoir Computing (RC) [12]. RC is a training method for recurrent neural networks and exists in different flavors. The method we use here is the Echo State Network (ESN) approach which is presented by H. Jaeger in 2001 [5], which means that a recurrent random network of tanh-neurons are used. RC greatly simplifies the training of recurrent neural networks.

The basic principle is as follows: a random recurrent neural network called the reservoir is created, and the input- and internal weights are globally rescaled to achieve certain desirable dynamic properties. A separate linear readout function is then trained on the response of this reservoir to the input signals. The reservoir is determined by only a few global parameters, which makes it easy to optimize. The most important parameters are the input scaling and the spectral radius. The input scaling determines the weights from the input to the reservoir, the spectral radius determines the global weight scaling inside the reservoir. Moreover, the linear readout function can be trained in a single step using a computationally efficient series of matrix operations. This training method avoid the long convergence times and vanishing gradient problems of traditional RNN learning rules.

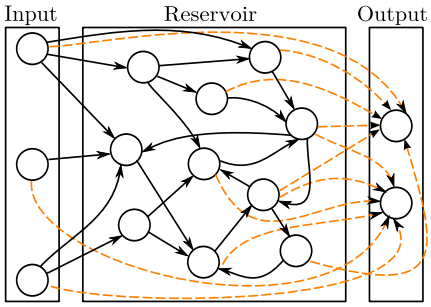

Figure 2: The Reservoir Computing concept. Only the dotted connections are trained. Some connections are not drawn to keep the picture clear.

In this contribution we use a reservoir consisting of so-called Leaky Integrator neurons [6]. These neurons can be described as standard neurons with a first-order low-pass filter added, which slows down the dynamics of the reservoir and extends its fading memory further in the past - which is a desirable property for the kind of long-term temporal processing needed in this task.

The operation of the reservoir can be described as follows: If we use $\mathbf{x}[k]$ to represent the current activation for each of the neurons in the reservoir we can calculate the next reservoir state $\mathbf{x}[k+1]$ using the following equation:

$$\mathbf{x}[k+1] = (1-\gamma)\mathbf{x}[k] + \gamma \tanh(W^{res}_{res}\mathbf{x}[k] + W^{res}_{inp}\mathbf{u}[k] + W^{res}_{bias}).$$

In this equation γ represents the leak rate and is used to set the cutoff frequency of the lowpass filter in the neurons, $u[k]$ is the input vector and W^{res}_{inp} represents the randomly generated weight vector from layer input to reservoir. The reservoir weight matrix is generated by a normal distribution with zero mean and unit variance. These weights are then rescaled so that the largest absolute eigenvalue of the weight matrix – called the spectral radius – equals a specific value. The input weights are randomly chosen out of $\{-0.1, 0.1\}$ and then multiplied by the input scaling. The output of the RC system, $\hat{y}[k]$, is determined by:

$$\hat{y}[k] = W^{out}_{inp}\mathbf{u}[k] + W^{out}_{res}\mathbf{x}[k] + W^{out}_{bias}.$$

In this study ridge regression was used to train $W^{out}_{inp}, W^{out}_{res}$ and W^{out}_{bias}. If

$$W = \begin{pmatrix} W^{out}_{inp}, & W^{out}_{res}, & W^{out}_{bias} \end{pmatrix}, X = \begin{pmatrix} \mathbf{u}[1], & \ldots, & \mathbf{u}[n] \\ \mathbf{x}[1], & \ldots, & \mathbf{x}[n] \\ 1, & \ldots, & 1 \end{pmatrix} \text{ and } Y = \begin{pmatrix} \mathbf{y}[1] \\ \mathbf{y}[2] \\ \vdots \\ \mathbf{y}[n] \end{pmatrix},$$

then it is possible to compute W as

$$W = \left((XX^T + \lambda I)^{-1} XY\right)^T,$$

where λ is the regularization parameter which has to be determined by cross validation.

We optimized the following parameters: the leak rate, bias scaling, connection fraction (the fraction of non-zero weights) of the internal reservoir weights, connection fraction from the input to the reservoir, input scaling and spectral radius. The optimization was done by performing cross validation on the training set. Because the reservoirs are randomly created, we repeated the main RC experiment 10 times and averaged the performance over reservoir instances to obtain statistically relevant results. This averaging was done both during parameter optimization and during the final evaluation.

3 Experiments And Discussion

3.1 Data and evaluation

We used data from the first task in the BCI Competition IV[1]. The task is to discriminate between two types of motor imagery and a resting state. This has to be done at every timestep and without

[1] http://www.bbci.de/competition/iv/

cues about the start or end of a trial. There are 7 different datasets available, four datasets containing natural EEG while the other three are artificially generated EEG. Only the first four (natural) datasets were used for evaluation purposes. Each subject has a dedicated training and a dedicated test set. The generated output for every timestep has to be a real number between -1 and 1. (class 1: -1, rest: 0, class 2: 1). It is important to point out that there is no response time limit. This means that the entire dataset can be processed before generating the output.

Three different evaluation methods are considered:

Discrimination between motor imagery classes This is done by using the single trial accuracy - so in this case the start and end cues were in fact used. For each motor imagery trial (or thought) the sign of the averaged output for that trial was used to label the trial. The fraction of correctly classified trials is given.

Rest state detection To evaluate the rest state detection we took the absolute value of the output: 1 is motor imagery, 0 is the idle state. The True Positive Rate (TPR) and False Positive Rate (FPR) is calculated for each threshold value between 0 and 1. The TPR is plotted versus the FPR to obtain the ROC curve, where a higher Area Under Curve (AUC) indicates better performance.

Mean square error (MSE) The MSE between the desired output and the actual output was used as performance measure in the BCI Competition. The MSE is computed according to the following equation:

$$MSE = \frac{\sum_{i=1}^{m}(\hat{y}_i - y_i)^2}{m},$$

where m is the total number of samples, $\hat{y}_i$ is the generated output at timestep i and y_i is the desired output at timestep i. We include MSE results in order to compare with the published results of the competition. All our results were obtained while optimizing the MSE, however the MSE gives little information about the discrimination between the different motor imagery classes and between motor imagery and the rest class. This is why the two other evaluation metrics were added to our experiments.

3.2 Experimental setups

Five different experimental setups were evaluated. The goal was to investigate how the different components influenced the performance. We start of with a default experiment and at each new experiment we increase the BCI's complexity. Postprocessing was applied before evaluating the MSE. This postprocessing consists of one single step: a lowpass filter (Butterworth filter with cutoff frequency 0.1 Hz) that was applied once in the forward and once in the reverse direction.

Linear regression with standard CSP In this experiment we applied bandpass filtering with a broad passband (8-30Hz). The CSP filters were then applied on these filtered signals. Then we computed the features as described in Section 2.1. We experimented with different windows for the variance calculation and the best performance was achieved by one starting 75 samples before the current timestep and stopping 75 timesteps after the current. The output was then generated by using linear regression on these features.

Linear regression with feature selection The only difference between this experiment and the previous one is the addition of the feature selection algorithm. The spectral filtering is done by using the subject specific passband and spatial filters. These passband and spatial filters are selected by the proposed algorithm. By comparing the results from this experiment with the previous one we can isolate the influence from the feature selection algorithm.

Linear regression with feature selection and delay lines We added delay lines before the linear regression so that information about past samples can be used. In this experiment we want to evaluate the influence on performance by using information from the past.

Reservoir Computing The delay lines with the linear regression is replaced by a Reservoir Computing system as described above. The features that are fed into the reservoir are the same as in the previous two experiments. We used a reservoir with 50 neurons, the leak rate was 0.025, the input scaling was 2, the spectral radius 0.8, the reservoir connection fraction 0.45, the input connection fraction 0.25 and the bias was 0.5. These parameter settings were obtained trough optimization using a grid search.

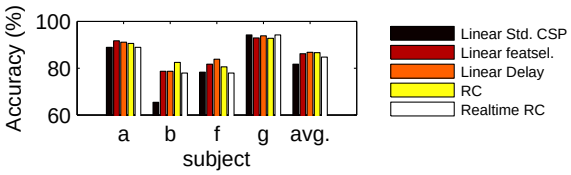

Figure 3: Discrimination between motor imagery states. Results are percentage correctly labelled trials.

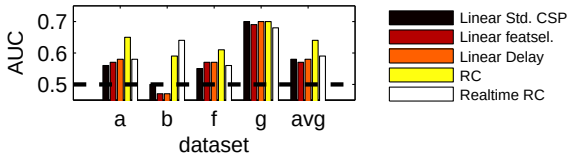

Figure 4: Results based on the rest state detection where the AUC measure was used. The dotted line represents chance level.

Real-time Reservoir Computing We made two changes in this experiment. The first change is that we omitted the variance calculation step, the input to the reservoir is just spectrally and spatially filtered EEG. This way the delay originating from the variance calculation is removed. The second change is that the postprocessing before MSE evaluation was not done because smoothing in realtime was not possible. This method works instantaneously, the output for the sample is generated without delay. This experiment is a proof of concept so only one reservoir was tested.

3.3 Discussion

If we take a look at the discrimination of the motor imagery classes, shown in Fig. 3, we find that the accuracy is very subject dependent: for instance, performance on subject g is much higher than on subject b. On average we see that without feature selection 81.7% of the samples are classified correctly. The addition of the feature selection algorithm increases this accuracy to 86.2%. The influence of the delay lines is minimal, an accuracy of 86.8% was achieved. The results achieved with Reservoir Computing are similar with an accuracy of 86.6%. In short: the feature selection algorithm has a large influence on the discrimination between motor imagery classes, Reservoir Computing does not.

The evaluation of the rest state detection – a feature which is crucial for real-world applicability of this BCI system because it eliminates the need for cues – shows that the addition of an idle state makes the BCI much more complex. (Fig. 4) The average AUC with the linear method is 0.58, on subject b the AUC is only just as good as chance level. Feature selection and delay lines do not improve the rest state detection, the average AUC becomes 0.57 with feature selection. The delay lines bring the AUC again up to 0.58. If we add Reservoir Computing the average AUC becomes 0.64. The performance on the easier subject g stays the same, on the other (harder) subjects there is more improvement. It is clear that Reservoir Computing enables us to improve the rest state detection significantly.

We now compare the results with the competition results based on MSE (Table 1). The linear method achieves good results on average, the MSE is 0.376 compared to the 0.382 of the winner. The best results per subject give an average MSE of 0.365 which leaves room for improvement. The addition of the feature selection algorithm brings the MSE down to 0.348 which is better than the best results from the competition. There is a slight decrease when the delay lines are added but this is much smaller than the decrease due to the feature selection. The addition of Reservoir Computing gives a second big improvement. The average MSE is now only 0.326. The realtime experiments give an MSE of 0.390. However it has to be said that the realtime constraints make this problem much more difficult than the task in the competition.

Table 1: Results based on the MSE. Comparison between methods and BCI Competition IV.

dataset	a	b	f	g	avg.
linear Std. CSP.	0.358	0.466	0.400	**0.280**	0.376
linear Featsel.	0.327	0.403	0.372	0.290	0.348
Linear Delay	0.324	0.403	0.369	0.286	0.345
RC	**0.296** (0.005)	**0.361** (0.041)	**0.361** (0.003)	0.284 (0.005)	**0.326**
Realtime RC	0.364	0.442	0.435	0.320	0.390
Competition winner	0.40	0.42	0.42	0.29	0.382
Competition best	0.35	0.42	0.40	0.29	0.365

4 Conclusions

In this paper we evaluated the performance of Reservoir Computing and a novel CSP algorithm in an uncued motor imagery BCI.We can conclude that the combination of Reservoir Computing and the proposed novel feature extraction method is a promising classification method for the difficult motor imagery BCI considered in this contribution. Its superior detection of the rest state compared to other methods increases the applicability of the method since start and stop cues are generally not available in real-world situations. Moreover, since the training of RC is entirely based on linear methods, a whole array of standard procedures for adaptability or online learning (such as Recursive Least-Squares) are available. This makes the system more robust against potential changes in the environment, which in turn again increases the usefulness of this setup in an actual assistive context.

References

[1] KK. Ang, ZY. Chin, H. Zhang, and C. Guan. Filter bank common spatial pattern (FBCSP) in brain-computer interface. In *Proceedings of IEEE International Joint Conference on Neural Networks*, pages 2390–2397. IEEE, 2008.

[2] B. Blankertz, G. Dornhege, M. Krauledat, K-R. Mueller, and G. Curio. The non-invasive Berlin Brain-Computer Interface: Fast acquisition of effective performance in untrained subjects. *Neuroimage*, 37(2):539–550, 2007.

[3] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K.-R. Muller. Optimizing Spatial filters for Robust EEG Single-Trial Analysis. *IEEE Signal Processing Magazine*, 25(1):41–56, 2008.

[4] G. Dornhege, B. Blankertz, M. Krauledat, F. Losch, G. Curio, and K.-R. Muller. Combined Optimization of Spatial and Temporal Filters for Improving Brain-Computer Interfacing. *IEEE Transactions on Biomedical Engineering*, 53(11):2274–2281, 2006.

[5] H Jaeger and H Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[6] H. Jaeger, M. Lukosevicius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007.

[7] Z. Koles, M. Lazar, and S. Zhou. Spatial patterns underlying population differences in the background EEG. *Brain Topography*, 2(4):275–284, 1990.

[8] J Muller-Gerking, G Pfurtscheller, and H Flyvbjerg. Designing optimal spatial filters for single-trial EEG classification in a movement task. *Clinical Neurophysiology*, 110(5):787–798, 1999.

[9] Q Novi, C. Guan, Tran H. Dat, and P. Xue. Sub-band common spatial pattern (SBCSP) for brain-computer interface. In *Proceedings of 3rd Int. IEEE/EMBS Conference on Neural Engineering, Vols 1 and 2*, pages 204–207. IEEE; EMBS, IEEE, 2007.

[10] G Pfurtscheller and FHL da Silva. Event-related EEG/MEG synchronization and desynchronization: basic principles. *Clinical Neurophysiology*, 110(11):1842–1857, 1999.

[11] G Pfurtscheller and C Neuper. Motor imagery activates primary sensorimotor area in humans. *Neuroscience Letters*, 239(2-3):65–68, 1997.

[12] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403, 2007.