16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, August 25-29, 2008, copyright by EURASIP

BAYESIAN FEATURE SELECTION APPLIED IN A P300 BRAIN-COMPUTER INTERFACE

Ulrich Hoffmann[†‡]*, Ashkan Yazdani*[‡]*, Jean-Marc Vesin*[⋆]*, Touradj Ebrahimi*[‡]

Fatronik-Tecnalia[†], Biorobotics Department, San Sebastian, Spain
Multimedia Signal Processing Group[‡], Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
Signal Processing Laboratory[⋆], Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
email: uhoffmann@fatronik.com, web: http://bci.epfl.ch

ABSTRACT

Feature selection is a machine learning technique that has many interesting applications in the area of brain-computer interfaces (BCIs). Here we show how automatic relevance determination (ARD), which is a Bayesian feature selection technique, can be applied in a BCI system. We present an computationally efficient algorithm that uses ARD to compute sparse linear discriminants. The algorithm is tested with data recorded in a P300 BCI and with P300 data from the BCI competition 2004. The achieved classification accuracy is competitive with the accuracy achievable with a support vector machine (SVM). At the same time the computational complexity of the presented algorithm is much lower than that of the SVM. Moreover, it is shown how visualization of the computed discriminant vectors allows to gain insights about the neurophysiological mechanisms underlying the P300 paradigm.

1. INTRODUCTION

Brain-computer interfaces (BCIs) are systems that enable communication with other persons or control of devices, only through cerebral activity, without using muscles. Almost all BCI systems rely on machine learning algorithms in order to translate measurements of cerebral activity into commands for a computer. An interesting subclass of machine learning algorithms for BCIs uses the strategy of feature selection. This means that during the training phase, the learning algorithm attempts to determine a small subset of features which is relevant to the classification task at hand (see [8] for a general introduction to feature selection methods).

One of the main motivations for employing feature selection is that classification accuracy can potentially be improved by excluding noisy and irrelevant features from the classification rule. Furthermore, using a small subset of features implies that the number of operations needed for classifying new examples is drastically reduced and classification becomes faster. Last but not least, feature selection often allows to better understand classification rules and the data that is classified. This is of great importance when analyzing new BCI paradigms or while finding new features for already existing paradigms.

To find new features for a BCI paradigm, an approach that might be used is to first build a large dictionary of features by using many different feature extraction methods. Then, a feature selection algorithm can be used to find the most relevant features. Another possible application in the BCI context is to test for the presence of artifacts in the data. For example in electroencephalogram (EEG) based BCIs, features selected mainly from frontal electrodes might raise the suspicion that subjects are using eye-movement or eye-blinks to control a BCI. Still another important application of feature selection is electrode selection. By applying feature selection to groups of features corresponding to different electrodes, the number of necessary electrodes for classifying cerebral activity can be reduced. Hence, the setup time and complexity of BCI systems can be reduced with the help of feature selection.

Given the various advantages of applying feature selection in BCIs, it is no wonder that many feature selection algorithms have already been described in the BCI literature. The simplest approaches to perform feature selection are so-called filter methods [8], in which the discriminative power is computed for each feature individually. Then, features are ranked by their discriminative power and a subset containing only the most discriminative features is used for training a standard machine learning algorithm [3, 11]. Filter methods are a simple and efficient tool for feature selection, however their performance is limited by the fact that each feature is analyzed individually. This can lead to the inclusion of many correlated and redundant features in the selected subset. Furthermore, features that are relevant only in combination with other features might not be selected by filter methods [8]. An additional problem is that the optimal number of selected features typically has to be decided by cross-validation, which is computationally expensive.

An approach to feature selection that does not suffer from the problems related to the analysis of individual features is to use regularization. In [1], Fisher's discriminant in conjunction with a regularization term that penalizes weight vectors with a large l_1 norm is used to compute sparse linear discriminants. It should be noted that the method described in [1] critically depends on a regularization constant which determines the degree of sparseness. The regularization constant has to be estimated through a cumbersome and computationally expensive cross-validation procedure.

Still another method for feature selection and channel selection in BCIs is to use SVMs along with recursive feature elimination (RFE) [11]. In SVM-RFE, first a linear discriminant is computed with the SVM learning algorithm. Then, the features corresponding to the weights with smallest absolute value are removed, and the procedure is repeated recursively until only a preset number of features remains. While the classification performance of SVM-RFE is very good, the method suffers from an extremely high computational complexity. This is because training a SVM is complex and because the optimal number of features as well as regularization constants have to be estimated via cross-validation.

In this paper, we explore the use of a feature selection method that is known as ARD in the neural networks literature or as the relevance vector machine (RVM) in the area of kernel methods [12, 14]. More specifically, we use ARD to perform classification via sparse linear regression to the class labels. We have termed the resulting algorithm Sparse Bayesian linear Discriminant Analysis (SBDA). The method we describe does not suffer from the limitations of filter methods, is free from regularization constants that have to be estimated by cross-validation, and has low computational complexity. We have previously used a similar method for electrode selection [9]. However, to the best of our knowledge, the ARD approach has previously not been used for feature selection in a BCI.

The outline of the rest of the paper is as follows. In Section 2, the datasets that were used to analyze the behavior of SBDA are described. In Section 3, feature extraction and ARD are described[1]. In Section 4, evaluation methods are explained. In Section 5 results are presented. A conclusion follows in Section 6.

2. DATASETS

To evaluate feature selection with SBDA, data recorded in a P300 environment control paradigm were used. The data were recorded while users were facing a laptop screen on which six images were displayed (see Fig. 1). The images were selected according to an application scenario in which users can control electrical appliances via a BCI system. The images on the screen were flashed in random sequences, one image at a time. Each flash of an image lasted for 100 ms and during the following 300 ms none of the images was flashed, i.e. the interstimulus interval (ISI) was 400 ms.

The idea underlying the P300 environment control paradigm is that subjects can select one specific target-image by concentrating on it. Since images are flashed in a random sequence and since the subject is concentrating on only one out of six images, it is expected that the target image will evoke a P300. Using the same principle as in the well known P300 speller paradigm [7], the target image can then be inferred from the EEG by looking for flashes of images which evoked a P300-like waveform [10].

During the recording of the data the task of the subjects was to silently count how often a prescribed image was flashed. For example, the operator would ask the subject, "Now please count how often the TV is flashed". Then, a random sequence of flashes was displayed, and after the end of the sequence subjects were asked to announce their count.

The data analyzed here were recorded with a 32-channel Biosemi amplifier from four male able-bodied subjects with an age of 30 ± 2.3 years. Typically the data for a given subject consists of about 540 target trials, and 2700 nontarget trials. The datasets are available for free download at `http://bci.epfl.ch/efficientp300bci.html`[2]. More details about the P300 environment control paradigm and about the datasets can be found in [10, 9].

In addition to the datasets described above, P300 datasets from the BCI competition 2004 were used for benchmarking. A description of these datasets can be found in [2].

[1] The description of ARD is given only for the sake of completeness. Alternative accounts of the same method can be found in [12, 14].

[2] At the same URL datasets from four disabled subjects can also be found. An analysis of these datasets is however beyond the scope of this paper.

Figure 1: Display used in the environment control system. Images were flashed, one at a time, by changing the overall brightness of images.

3. ALGORITHMS

3.1 Feature Extraction

The following steps were used to build feature vectors from the EEG data recorded with the environment control paradigm:

1. *Referencing*: The average signal from electrodes T7 and T8 was used for referencing.
2. *Filtering*: A 6th order forward-backward Butterworth bandpass filter with cutoff frequencies of 1 Hz and 12 Hz was used to filter the data.
3. *Downsampling*: The EEG was downsampled from 2048 Hz to 32 Hz by selecting each 64th sample.
4. *Single trial extraction*: Single trials of length 1000 ms, starting at stimulus onset, i.e. at the beginning of the intensification of an image, were extracted from the data.
5. *Windsorizing*: To reduce the effects of large amplitude outliers, the data from each electrode were windsorized. The 10th percentile and the 90th percentile were computed for the samples from each electrode. Values below the 10th percentile or above the 90th percentile were replaced by the respective percentiles.
6. *Feature vector construction*: The samples from all 32 electrodes were concatenated into feature vectors. The dimensionality of the feature vectors was $32 \times 32 = 1024$.
7. *Normalization*: The mean and standard deviation were computed for each of the 1024 features. Then, features were normalized by subtracting the mean and by dividing by the standard deviation.

To extract features from the BCI competition data, the method of the competition winners as described in [13] was used.

3.2 Sparse Bayesian Discriminant Analysis

3.2.1 Likelihood, Prior, and Posterior

To describe SBDA, it is useful to first look at the model which links D-dimensional feature vectors $\mathbf{x} \in \mathbb{R}^D$ to class labels $t \in \{-1, 1\}$:

$$t = \mathbf{w}^\mathsf{T}\mathbf{x} + n.$$

Here $\mathbf{w} \in \mathbb{R}^D$ contains the weights assigned to different features and $n \in \mathbb{R}$ is drawn from a white Gaussian noise process. The class labels are thus modeled to be linear combinations of the features with additive Gaussian noise.

Using the above model, we can derive an expression for the likelihood of different weight vectors $\mathbf{w}$, given the training data. To this end, we introduce some further notation. We denote by $\mathbf{X} \in \mathbb{R}^{D\times N}$ the matrix resulting from the horizontal stacking of N feature vectors, by $\mathbf{t} \in \{-1,1\}^N$ the vector containing all class labels, by $\mathbf{D}$ the pair $(\mathbf{X},\mathbf{t})$, and by β the inverse variance of the Gaussian noise process. The likelihood for $\mathbf{w}$ then is:

$$p(\mathbf{D}|\beta,\mathbf{w}) = \left(\frac{\beta}{2\pi}\right)^{\frac{N}{2}} \exp(-\frac{\beta}{2}\|\mathbf{X}^\mathsf{T}\mathbf{w}-\mathbf{t}\|^2).$$

To compute the posterior distribution of $\mathbf{w}$ in the Bayesian paradigm, a prior distribution has to be specified. This distribution allows to express prior knowledge about weight vectors and can potentially have a strong influence on the posterior distribution. The prior used in SBDA is a multivariate Gaussian density with a zero mean vector and a diagonal covariance matrix. Denoting by α_i the inverse variance of the prior distribution for weight w_i, the prior can be expressed as

$$p(\mathbf{w}|\alpha) = \prod_{i=1}^{D}\left(\frac{\alpha_i}{2\pi}\right)^{\frac{1}{2}} \exp\left(-\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{I}'(\alpha)\mathbf{w}\right),$$

where $\mathbf{I}'$ is a matrix of size $D\times D$, with elements $\alpha_1 \ldots \alpha_D$ on the diagonal and all other elements zero. As we will see in Section 3.2.2, the hyperparameters α_i are used to estimate the relevance of each weight w_i, i.e. to perform feature selection. In fact, the hyperparameters corresponding to irrelevant features go to infinity and hence the corresponding weights can be set to zero.

Note that other useful learning algorithms can be built by using variations of the above prior. Using the same value in all diagonal entries of $\mathbf{I}'$, the prior distribution becomes isotropic and can be used to build a classification algorithm in which weight vectors are shrunk to zero but in which no feature selection is performed. Experiments with an algorithm using such a prior are described in [9, 10]. Still another possibility is to use one α_i for each electrode. Then, electrode selection can be performed [9].

Given likelihood and prior, the posterior distribution of $\mathbf{w}$ can be computed using Bayes rule:

$$p(\mathbf{w}|\beta,\alpha,\mathbf{D}) = \frac{p(\mathbf{D}|\beta,\mathbf{w})p(\mathbf{w}|\alpha)}{\int p(\mathbf{D}|\beta,\mathbf{w})p(\mathbf{w}|\alpha)\,d\mathbf{w}}. \qquad (1)$$

Since both prior and likelihood are Gaussian, the posterior is also Gaussian and its parameters can be derived from likelihood and prior by completing the square. The mean $\mathbf{m}$ and covariance $\mathbf{C}$ of the posterior satisfy the following equations.

$$\mathbf{C} = (\beta\mathbf{X}\mathbf{X}^\mathsf{T} + \mathbf{I}'(\alpha))^{-1} \quad \mathbf{m} = \beta\mathbf{C}\mathbf{X}\mathbf{t}$$

The posterior distribution can be used to compute a probability distribution of regression targets $\hat{t}$ for a previously unseen feature vector $\hat{\mathbf{x}}$. This so-called predictive distribution is obtained by integrating over $\mathbf{w}$:

$$p(\hat{t}|\beta,\alpha,\hat{\mathbf{x}},\mathbf{D}) = \int p(\hat{t}|\beta,\hat{\mathbf{x}},\mathbf{w})p(\mathbf{w}|\beta,\alpha,\mathbf{D})\,d\mathbf{w}.$$

The predictive distribution is again Gaussian and can be characterized by its mean μ and its variance σ^2.

$$\mu = \mathbf{m}^\mathsf{T}\hat{\mathbf{x}}, \qquad \sigma^2 = \frac{1}{\beta} + \hat{\mathbf{x}}^\mathsf{T}\mathbf{C}\hat{\mathbf{x}}$$

While in principle the predictive distribution can be used to compute class probabilities [9], here we have only used the mean of the predictive distribution for classification.

3.2.2 *Maximization of the Marginal Likelihood*

To compute β,α we write down the likelihood for the hyperparameters. The likelihood $p(\mathbf{D}|\beta,\alpha)$ is the normalizing integral from equation 1.

$$p(\mathbf{D}|\beta,\alpha) = \int p(\mathbf{D}|\beta,\mathbf{w})p(\mathbf{w}|\alpha)\,d\mathbf{w} \qquad (2)$$

The quantity $p(\mathbf{D}|\beta,\alpha)$ is also known as the evidence, or the marginal likelihood, and corresponds to the probability of the data given the hyperparameters β and α. The integral in equation 2 can be solved by considering that everything is Gaussian and using standard expressions for Gaussian integrals. After computing the integral, it is convenient to use the logarithm of the likelihood function for further analysis.

$$\begin{aligned}\log(p(\mathbf{D}|\beta,\alpha)) =& \frac{1}{2}\sum_{i=1}^{D}\log(\alpha_i) + \frac{N}{2}\log(\beta) \\ &- \frac{N}{2}\log(2\pi) + \frac{1}{2}\log(\det(\mathbf{C})) \\ &- \frac{\beta}{2}\|\mathbf{X}^\mathsf{T}\mathbf{m}-\mathbf{t}\|^2 - \frac{1}{2}\mathbf{m}^\mathsf{T}\mathbf{I}'(\alpha)\mathbf{m}\end{aligned}$$

To maximize this log-likelihood, partial derivatives with respect to the α_i and β are taken and equated to zero. For this purpose, the following identity is useful:

$$\frac{\partial}{\partial x}\log\det\mathbf{A} = \mathrm{tr}\left(\mathbf{A}^{-1}\frac{\partial\mathbf{A}}{\partial x}\right).$$

Using this identity we obtain

$$\frac{\partial\log(p(\mathbf{D}|\beta,\alpha))}{\partial\alpha_i} = \frac{1}{2\alpha_i} - \frac{1}{2}c_{ii} - \frac{1}{2}m_i^2,$$

where the c_{ii} are the values on the diagonal of $\mathbf{C}$ and the m_i are the elements of $\mathbf{m}$. Taking the derivative with respect to β yields

$$\frac{\partial\log(p(\mathbf{D}|\beta,\alpha))}{\partial\beta} = \frac{N}{2\beta} - \frac{1}{2}\mathrm{tr}(\mathbf{X}\mathbf{X}^\mathsf{T}\mathbf{C}) - \frac{1}{2}\|\mathbf{X}^\mathsf{T}\mathbf{m}-\mathbf{t}\|^2.$$

Setting the derivatives to zero and solving for α_i and β we obtain the update equations:

$$\alpha_i = \frac{1}{c_{ii}+m_i^2} \qquad (3)$$

$$\beta = \frac{N}{\mathrm{tr}(\mathbf{X}\mathbf{X}^\mathsf{T}\mathbf{C}) + \|\mathbf{X}^\mathsf{T}\mathbf{m}-\mathbf{t}\|^2}. \qquad (4)$$

The partial derivatives for the α_i and β depend on the posterior mean $\mathbf{m}$ which itself depends on the α_i and β. Equations 3 and 4 thus represent implicit solutions for the hyperparameters. Thus, to maximize the log-likelihood an iterative scheme is used in which first $\mathbf{C}$ and $\mathbf{m}$ are computed for a given setting of the hyperparameters and then the hyperparameters are updated according to equations 3 and 4. Features for which the α_i become very large can be removed from the training data during the optimization process. Typically this quickly reduces the dimensions of $\mathbf{C}$ and $\mathbf{m}$ and helps to speed up the training process.

4. EVALUATION METHODS

4.1 Environment Control Datasets

4.1.1 Cross-Validation

Ten-fold cross-validation was used to analyze the features selected by SBDA and to get an idea of the achievable classification performance. More specifically, the data for each of the four subjects was partitioned into ten subsets of equal size. Then, SBDA was trained on nine of the subsets, the resulting weight vector $\mathbf{w}$ was stored in a file, and the classification performance was tested on the data in the tenth subset. This procedure was repeated ten times, each time using a different subset of data for testing. The ten-fold cross-validation was repeated five times for each subject, each time using a different partition of the data. The result of the cross-validation procedure were thus fifty weight vectors and fifty performance estimates for each subject.

To estimate the classification performance achievable with an SVM, the procedure described above was augmented with a five-fold inner cross-validation. More specifically, on each of the fifty randomly drawn training sets a five-fold cross-validation was performed in order to estimate the regularization parameter of the SVM.

4.1.2 Relevance Maps

Given the weight vectors computed with cross-validation, we used the following steps to compute and visualize the relevance of features:

1. The l_1 norm of each weight vector was set to 1 and the absolute value of each weight was computed. Denoting by w_{ij} the j-th weight in the i-th weight vector, this can be expressed as $r_{ij} = |w_{ij}| / \sum_{k=1}^{D} |w_{ik}|$ The result of this step was a set of relevance vectors $\mathbf{r}_i$ with entries r_{ij}.
2. The average relevance $\bar{r}_j$ was computed from all relevance vectors: $\bar{r}_j = \frac{1}{R} \sum_{i=1}^{R} r_{ij}$
3. The average relevance values were multiplied by 100 for easy interpretation and plotted in the form of scalp maps. Scalp maps were computed for eight temporal segments of length 125 ms each. Scalp maps for individual temporal segments were computed by summing for each electrode the corresponding relevance values $\bar{r}_j$.
4. Scalp maps were visualized with EEGLAB [6].

4.1.3 Classification Accuracy

To compare the classification accuracy of SBDA and SVM, a performance measure called per block accuracy (PBA) was used (see also [9]). PBA measures the percentage of blocks of feature vectors in which the target image can be correctly identified from the classifier output. For the data analyzed in this paper, blocks containing six feature vectors were used. Each of the six feature vectors in each block corresponded to a flash of one of the images in the P300 environment control (cf. Fig. 1). Blocks were counted as correctly classified if the largest classifier output was obtained for the feature vector corresponding to the flash of the target image.

4.2 BCI Competition Datasets

For the BCI competition datasets only classification accuracy was evaluated. Classifiers were trained on the competition training sets and tested on the competition test sets.

5. RESULTS

5.1 Selected Features

On average 229 out of the 1024 features were selected and hence about 80% of the features were discarded by SBDA. The number of selected features differed only little between subjects (cf. Table 1). Analysis of individual weight vectors showed that the relevance of selected features varied between 0% and 3%, with many features having a relevance around 0.5% and only some features having a relevance bigger than 1%.

The distribution of relevance on electrodes and time intervals is shown in Fig. 2. Looking at the temporal distribution, one can see that the interval 250ms - 375ms is most relevant. In particular, in this interval, electrode Pz is strongly relevant. It is known that the P300 evoked by visual stimuli appears approximately 300ms - 500ms after stimulus onset and has maximal amplitude at parietal sites. The large weights at electrode Pz in the interval 250ms - 375 ms thus seemingly serve to "pick up" the P300.

The second most relevant interval is the interval 125ms - 250ms. Since mainly occipital and parietal electrodes are relevant in this interval, it seems possible that visual evoked potentials (VEPs) also play an important role for classification. This might be explained by assuming that most of the subjects focus on the target image. Hence, target images appear in the foveal visual field, whereas nontarget images appear in the peripheral visual field. It is known that in the foveal visual field the number of neurons representing objects of a given size is much larger than the number of neurons representing objects of the same size in the peripheral visual field [5]. VEPs corresponding to flashes of the target image can thus be expected to have a significantly larger amplitude than VEPs corresponding to flashes of nontarget images.

5.2 Classification Accuracy and Complexity

A comparison of the classification accuracy of SBDA with the classification accuracy of a linear ν-SVM is shown in Table 1 and Table 2. To perform the comparison we used LIBSVM, which is a state-of-the-art implementation of the SVM [4]. As can be seen, there is no significant difference in accuracy between SVM and SBDA (cf. Table 1, Table 2).

Both SBDA and the SVM were run on a PC with a 3.4 Ghz processor and 1GB of RAM. SBDA was implemented in MATLAB, while LIBSVM is implemented in C. Performing one cross-validation fold, consisting of classifier learning and testing, took on average one minute with SBDA and twenty minutes with the SVM. An advantage of SBDA is

	S6	S7	S8	S9	Average
# Features	209	231	259	217	229±22
PBA SBDA	68	72	87	59	72±12
PBA SVM	68	71	84	61	71±10

Table 1: Number of features and classification accuracy for the four able-bodied subjects S6, S7, S8, and S9 (subject names were chosen to be consistent with other publications [9, 10]). Classification accuracy is expressed as PBA (cf. Section 4.1.3) and shown for SBDA and SVM. Classification by chance would have resulted in a PBA of 16.66%.

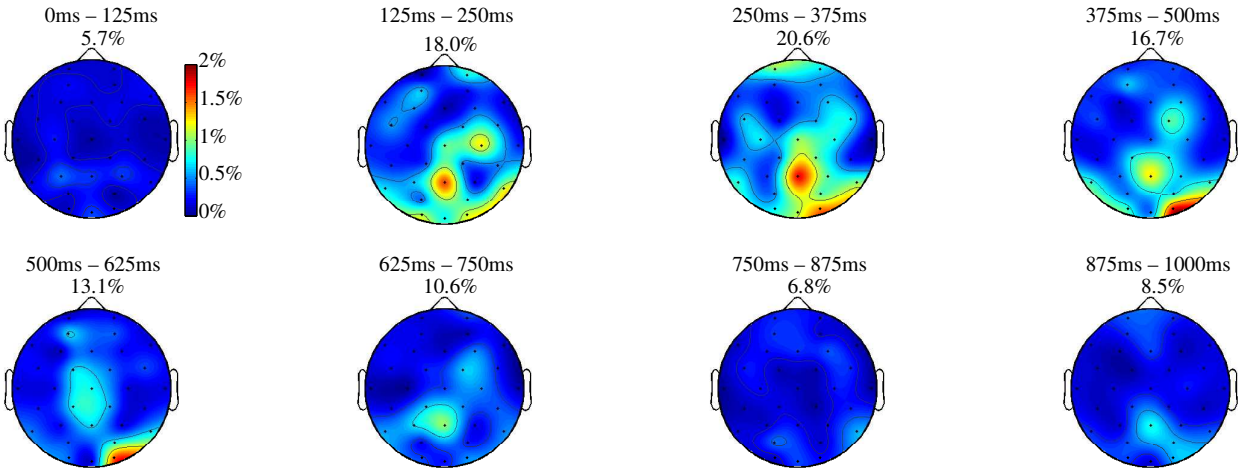

Figure 2: Average relevance maps obtained from data of four able-bodied subjects. Maps were computed from SBDA weight vectors. Colors indicate the contribution in % to the total l_1 norm of weight vectors, Numbers above maps show the summed contribution of all electrodes in temporal intervals.

	Subject A		Subject B		Average	
	5	15	5	15	5	15
SBDA	68	98	82	97	75	97.5
SVM	79	96	59	98	69	97

Table 2: Classification accuracy for BCI competition data. Shown is the percentage of correctly predicted symbols in the test set after 5 and 15 repetitions.

thus that its computational complexity is much lower than that of the SVM. This is important in many situations, for example when the classification accuracy for different feature extraction methods has to be compared, or when classifiers have to be set up quickly from freshly acquired training data.

6. CONCLUSION

We have shown how ARD can be used for feature selection in BCI machine learning tasks. We have described SBDA, which is an algorithm that uses ARD for feature selection. Experiments showed that the classification accuracy of SBDA is similar to that achievable with an SVM. At the same time SBDA has much smaller computational complexity than the SVM and offers the advantages of feature selection mentioned in the introduction of this paper. Moreover, we demonstrated how the sparse discriminant vectors computed by SBDA can be visualized in the form of relevance maps. The relevance maps allowed for a better understanding of the neurophysiological mechanisms underlying the P300 paradigm analyzed in this paper.

REFERENCES

[1] B. Blankertz, G. Curio, and K. R. Müller. Classifying single trial EEG: Towards brain-computer interfacing. In *Adv. Neur. Inf. Proc. Sys. (NIPS)*, volume 14, 2002.

[2] B. Blankertz, K. Müller, D. Krusienski, G. Schalk, J. Wolpaw, A. Schlögl, G. Pfurtscheller, J. Millan, M. Schröder, and N. Birbaumer. The BCI Competition III: Validating alternative approaches to actual BCI problems. *IEEE Trans. Neur. Syst. Rehab. Eng.*, 14(2):153–159, 2006.

[3] V. Bostanov. Feature extraction from event-related brain potentials with the continuous wavelet transform and the t-value scalogram. *IEEE Trans. Biomed. Eng.*, 51(6):1057–1061, 2004.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[5] P. Daniel and D. Whitteridge. The representation of the visual field on the cerebral cortex in monkeys. *J. Physiol.*, 159:203 – 221, 1961.

[6] A. Delorme and S. Makeig. EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. *J. Neurosci. Meth.*, 134(1):9–21, 2004.

[7] L. A. Farwell and E. Donchin. Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalogr. Clin. Neurophysiol.*, 70:510–523, 1988.

[8] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[9] U. Hoffmann. *Bayesian Machine Learning Applied in a Brain-Computer Interface for Disabled Subjects*. PhD thesis, École Polytechnique Fédérale de Lausanne, Switzerland, 2007.

[10] U. Hoffmann, J.-M. Vesin, T. Ebrahimi, and K. Diserens. An efficient P300-based brain-computer interface for disabled subjects. *J. Neurosci Meth.*, 167:115–125, 2008.

[11] T. Lal, M. Schröder, T. Hinterberger, J. Weston, M. Bogdan, N. Birbaumer, and B. Schölkopf. Support vector channel selection in BCI. *IEEE Trans. Biomed. Eng.*, 51(6):1003–1010, 2004.

[12] D. J. C. MacKay. Probable networks and plausible predictions – a review of practical bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6:469–505, 1995.

[13] A. Rakotomamonjy, V. Guigue, G. Mallet, and V. Alvarado. Ensemble of SVMs for improving brain-computer interface P300 speller performances. In *Int. Conf. Neur.l Netw. (ICANN)*, 2005.

[14] M. Tipping. Sparse bayesian learning and the relevance vector machine. *J. Mach. Learn. Res.*, 1(3):211–244, 2001.