ESANN 2010 proceedings, European Symposium on Artificial Neural Networks - Computational Intelligence and Machine Learning. Bruges (Belgium), 28-30 April 2010, d-side publi., ISBN 2-930307-10-2.

Directional predictions for 4-class BCI data

Dieter Devlaminck[1], Willem Waegeman[3], Bruno Bauwens[1], Bart Wyns[1], Georges Otte[4], Luc Boullart[1], Patrick Santens[2]

1- Ghent University - Electrical Energy, Systems and Automation
Technologiepark 913, 9052 Zwijnaarde - Belgium

2- Ghent University Hospital - Dept of Neurology
De Pintelaan 185, 9000 Gent - Belgium

3- Ghent University - Dept of Applied Mathematics, Biometrics and Process Control, Coupure links 653, 9000 Gent - Belgium

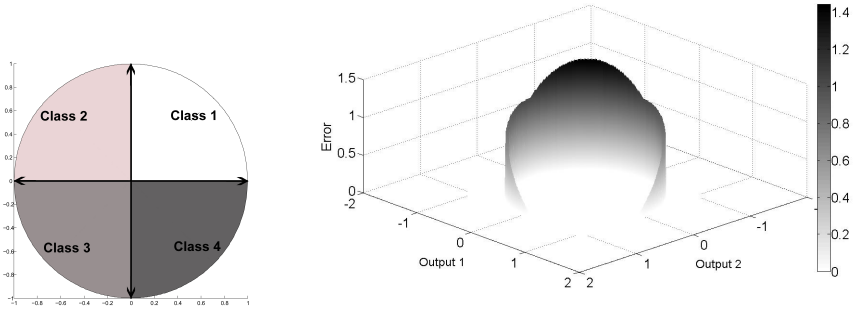
4- P.C. Dr. Guislain
Fr. Ferrerlaan 88A, 9000 Gent - Belgium

Abstract. Brain-computer interfaces (BCI) can allow disabled people to drive a wheel chair, just by imagining movement. Therefore, present day BCIs use training data, recorded during four different conditions, to calibrate a classifier. This, however, causes jerky behavior while abruptly switching between discrete states. We propose a cost-sensitive support vector approach for estimating two-dimensional directions based on four-class BCI data, that is recorded from three subjects.

We found that our method reduces the number of severe errors compared to classical support vector machines and results in a smoother trajectory estimate for application in a wheel chair.

1 Introduction

Brain-computer interfaces (BCI) [1, 2] are a hot topic within machine learning and still pose a lot of challenges, especially the systems that use motor imagination. These interfaces typically involve a calibration session, in which the user looks at a screen, displaying one of four possible cues, while his brain activity is recorded through a number of electrodes. The different cues typically inquire the subject to imagine left or right hand movement, foot movement or tongue movement for a certain period of time, which is denoted as the trial. Afterwards, one computes a number of features for each trial to train a classifier. This setup is very well suited for a lab environment, but when people want to drive a wheel chair through a BCI system, it is necessary to predict the intended direction. In other words, the output of the classifier should be two-dimensional and continuous. This is off course in contrast to the discrete target labels one gets from a classifier after a calibration session. On top of that, we also want to make our model cost-sensitive. To clarify this need, imagine an application where the user wants to control a computer cursor or a wheel chair by imagining one of the above mentioned movements. If a user wants to steer a wheel chair straight ahead, but slightly deviates to the right or left, then this results in a less severe error, compared to turning around and driving back in the opposite direction.



(a) (b)

Fig. 1: The left figure shows the different class regions in the two-dimensional prediction space Ω. The two constraints in (2) define the region for objects of class one. The right figure shows how the error ($= \xi_i + \xi_i^*$) is distributed for misclassified objects of class one. Objects classified in the upper right region do not get assigned an error, while objects classified in the opposite region get the largest penalty. Notice the discontinuities at the borders of the opposite region. Here, two slack variables are non-zero, leading to a larger penalty.

That is why we propose a machine learning method that makes continuous two-dimensional predictions, given training data with solely discrete, circular ordered labels. To this end, we use the framework of support vector machines (SVM) [3, 4]. We will also briefly explain the similarity and difference with the multi-class SVM algorithm proposed in [5], but due to the page limitation we will not go into detail.

2 Method

Let us consider the two-dimensional projection space Ω and divide it into four equal regions, as shown in Figure 1a. The goal is to project the given observations $\mathbf{x}$, belonging to one of the four classes, so that the projected points belong to their respective regions. More formally, we are looking for a function $f(\mathbf{x}) : \mathcal{X} \mapsto \Re^2$ which maps the objects in this two-dimensional space. The decision function is denoted by $h(\mathbf{x}) : \Re^2 \mapsto \mathcal{Y} = \{0, 1, 2, 3\}$, where 0, 1, 2 and 3 represent the labels of the four classes. The decision function classifies the object as class one if the two-dimensional projection lies in the upper right quadrant (see Figure 1a). The same reasoning can be applied to the other classes.

We study the use of a function $f(\mathbf{x}) = W^T \Phi(\mathbf{x}) + \mathbf{b}$ with $W \in \Re^{d \times 2}$,

$$W^T = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1d} \\ w_{21} & w_{22} & \dots & w_{2d} \end{pmatrix} = \begin{pmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \end{pmatrix} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}.$$

Next, we consider four vectors, as displayed in Figure 1a, normal to the boundaries, that divide the two-dimensional space Ω in four equal regions. The

four normal vectors

$$\begin{pmatrix}1\\0\end{pmatrix}, \quad \begin{pmatrix}0\\1\end{pmatrix}, \quad \begin{pmatrix}-1\\0\end{pmatrix}, \quad \begin{pmatrix}0\\-1\end{pmatrix}, \tag{1}$$

can now be used to define the four regions as follows,

$$(W^T\Phi(\mathbf{x}_i)+\mathbf{b})^T \begin{pmatrix}\cos(\frac{2\pi y_i}{4})\\ \sin(\frac{2\pi y_i}{4})\end{pmatrix} \geq 1 \quad \text{and} \quad (W^T\Phi(\mathbf{x}_i)+\mathbf{b})^T \begin{pmatrix}\cos(\frac{2\pi (y_i+1)}{4})\\ \sin(\frac{2\pi (y_i+1)}{4})\end{pmatrix} \geq 1, \tag{2}$$

$$\forall i : y_i \in \mathcal{Y}.$$

Similar to standard SVM, we can also introduce slack variables ξ_i and ξ_i^* to account for overlapping class distributions. Figure 1b illustrates the intuition behind the slack variables. These slack variables can be included as error terms in the minimization problem, leading to,

$$\min_{\mathbf{w}_1,\mathbf{w}_2,b_1,b_2,\xi} \frac{1}{2}(\mathbf{w}_1^T\mathbf{w}_1 + \mathbf{w}_2^T\mathbf{w}_2 + b_1^2 + b_2^2) + C\sum_{i=1}^{N} \xi_i + \xi_i^*$$

with constraints,

$$\begin{aligned}
\mathbf{w}_2^T\Phi(\mathbf{x}_i) + b_2 + \xi_i^* &\geq 1, && \forall y_i = 0 \vee y_i = 1 && (3)\\
\mathbf{w}_1^T\Phi(\mathbf{x}_i) + b_1 + \xi_i &\geq 1, && \forall y_i = 0 \vee y_i = 3 && (4)\\
-\mathbf{w}_1^T\Phi(\mathbf{x}_i) - b_1 + \xi_i &\geq 1, && \forall y_i = 1 \vee y_i = 2 &\\
-\mathbf{w}_2^T\Phi(\mathbf{x}_i) - b_2 + \xi_i^* &\geq 1, && \forall y_i = 2 \vee y_i = 3 &\\
\xi_i, \xi_i^* &\geq 0. &&&
\end{aligned}$$

Deriving the dual leads to the following maximization problem,

$$\max_{\alpha} \sum_{i=1}^{N}(\alpha_i^1 + \alpha_i^2 + \alpha_i^3 + \alpha_i^4) + \sum_{i,j}\Big[\Big(\alpha_i^1\alpha_j^4 + \alpha_i^2\alpha_j^3\Big) - \frac{1}{2}\Big(\alpha_i^1\alpha_j^1 + \alpha_i^2\alpha_j^2 + \alpha_i^3\alpha_j^3 + \alpha_i^4\alpha_j^4\Big)\Big](K_{ij}+1).$$

with constraints $C \geq \alpha_i^1, \alpha_i^2, \alpha_i^3, \alpha_i^4 \geq 0$. We use MOSEK [6], a software package for solving optimization problems, to find the maximum of this quadratic program. After computing α, one can make predictions for new test objects $\mathbf{x}$ as follows,

$$f(\mathbf{x}) = \sum_{i=1}^{N} \begin{pmatrix}\alpha_i^2 - \alpha_i^3\\ \alpha_i^1 - \alpha_i^4\end{pmatrix}(K(\mathbf{x}_i, \mathbf{x}) + 1).$$

The main difference with the multi-class SVM approach [5] is the use of only two separating hyperplanes instead of four in the four-class case. To see the

ESANN 2010 proceedings, European Symposium on Artificial Neural Networks - Computational Intelligence and Machine Learning. Bruges (Belgium), 28-30 April 2010, d-side publi., ISBN 2-930307-10-2.

similarity between the two, rotate the vectors in (1) by -45 degrees and stretch them so that they only contain ones. Write out the eight constraints as above. Now, substitute each positive $\mathbf{w}_1$ with $-\mathbf{w}_3$ and each positive $\mathbf{w}_2$ with $-\mathbf{w}_4$. When comparing these eight constraints with the 12 constraints in [5] for the four-class case, you will see that all eight find a match. The other four are not necessary for our purpose.

3 Experiments

We apply the proposed method to real-life data of the BCI competition, more specifically the data set IIIa [7]. It contains data of three subjects, who performed four different tasks. The four tasks represent the imagination of left hand, right hand, foot and tongue movement, recorded across 60 electrodes.

In a nutshell, we discuss the basic preprocessing method used. It is, however, not our intention to compete with current state-of-the-art preprocessing methods. We merely want to compare the use of a regular SVM classifier against the proposed method. Firstly, the signal is filtered between different frequency ranges (each filter bank having a range of 4hz) and a popular spatial filter is applied to each of them, namely common spatial patterns (CSP) [8]. For the multi-class CSP algorithm, we use the one-versus-all approach. In this way, each spatial filter corresponds to one of the classes and for the most discriminative spatial filters the most important filter banks are chosen based on the Fisher ratio. Subsequently, the temporal and spatial filtered EEG signal is divided in epochs (the middle two seconds of each trial). For each of these epochs we calculate the variance, which further serves as feature for the classifier. For evaluation purposes, a moving window of one second is employed to calculate the features on the test data. The window is then shifted with one sample, so that as many predictions as test instances are obtained.

The default Gaussian kernel is used to train the models. Thus, we need to determine the parameters C and γ through cross-validation. First, a coarse grained search is performed for $C = 2^{-17:4:23}$ and $\gamma^{-7:2:13}$ and afterwards a fine grained search is done by halving the step sizes. After cross-validation, we apply the learned model to the generated test set in order to get two-dimensional predictions.

In order to compare the methods we use different performance measures. The first and most common one is the accuracy. Secondly, we give the circular mean-squared error (cMSE) which is the evaluation of the following cost function,

$$\sum_{i=1}^{N} M_{y_i,h(\mathbf{x}_i)} \,,$$

where

$$M = \begin{pmatrix} 0 & 1 & 4 & 1 \\ 1 & 0 & 1 & 4 \\ 4 & 1 & 0 & 1 \\ 1 & 4 & 1 & 0 \end{pmatrix},$$

Table 1: Comparison between regular SVM and circular SVM on the BCI test data.

	Accuracy (%)		circular MSE		Severe errors (%)	
subject	SVM	cSVM	SVM	cSVM	SVM	cSVM
k3b	79	81	0.38	0.28	5	2
k6b	49	48	0.94	0.84	12	10
l1b	48	48	0.72	0.65	6	4

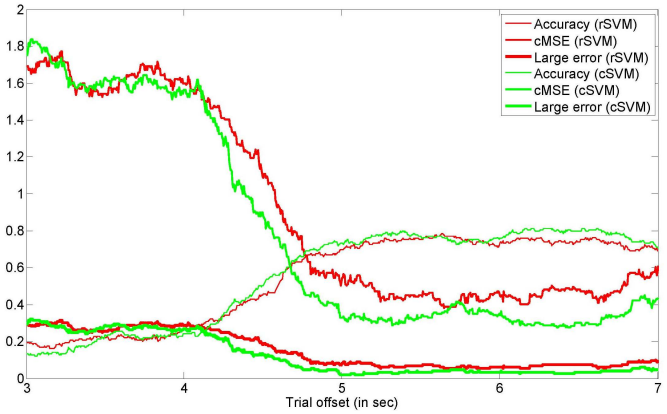


Fig. 2: Time course of three performance measures, accuracy, circular MSE and number of large errors, for trials of subject *k3b*.

y_i is the target label and $h(\mathbf{x}_i)$ the predicted label. Thirdly, we present the number of large errors, given as a percentage of the total number of test objects. These are the predictions that correspond to the largest entries of the above cost matrix and consequently get the largest penalty.

For computing these performance measures, we employ the method proposed in [9]. Here, the measure is calculated and averaged at each sample over all trials of the test set. In this way, we become a time course of the respective measure as shown in Figure 2. For the accuracy we then use the maximum occurring value as the final measure, while for the number of severe errors and the circular MSE we use the minimum. We can clearly see that for subject *k3b* the circular SVM performs best, during almost the whole trial when looking at the number of large errors and the cMSE, while it performs worse than the regular SVM classifier when looking at the accuracy. Table 1 shows that cSVM seems to perform better under the cMSE for all three subjects. On one subject *l1b* the accuracy is even better for circular SVM compared to the regular SVM approach.

Figure 3 displays for both methods a sample trajectory that simulates the estimated direction for driving a wheel chair. Comparing the two trajectories in Figure 3, one can see that without post-processing the regular SVM trajectory is not exactly smooth in contrast to the one of the proposed method. Moreover, in this example, regular SVM makes a severe error while our method still reaches

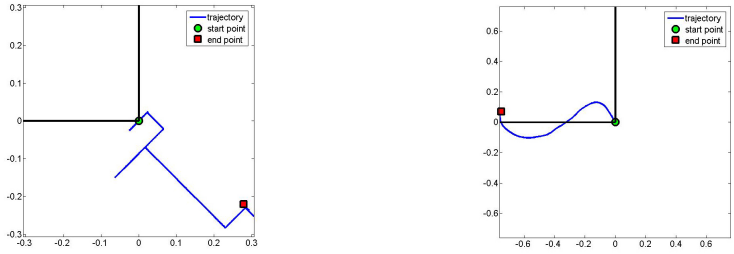


Fig. 3: Two trajectories are shown of the same trial by two different methods, regular SVM (left) and the proposed ordinal method (right). The black rectangle denotes the target region. One can see that the ordinal method results in a smoother trajectory. In the case of regular SVM only four different directions can be taken, so that sometimes the method takes a few steps back along the same trajectory before going into a different direction. Based on the raw output of regular SVM, this results in jerky movements.

the goal quadrant, defined by the black rectangle.

4 Conclusion

We proposed a four-class SVM model for circular labeled data. Offline analysis of the BCI data showed us a significant decrease in severe errors and much smoother trajectory estimates. Both aspects will be used in future work to get better direction estimates for driving a wheel chair or controlling a computer cursor through a BCI system.

References

[1] G. R. Müller-Putz, R. Scherer, G. Pfurtscheller, and R. Rupp. EEG-based neuroprosthesis control: a step towards clinical practice. *Neuroscience Letters*, 382:169–174, 2005.

[2] G. Pfurtscheller, G. R. Müller, J. Pfurtscheller, H. J. Gerner, and R. Rupp. 'Thought'-control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia. *Neuroscience Letters*, 351:33–36, 2003.

[3] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[4] C.W. Hsu and C.J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, Mar 2002.

[5] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks, Bruges, Belgium*, pages 219–224, 1999.

[6] The mosek optimization software. Available at `http://www.mosek.com/`.

[7] BCI Competition III. Technical report, IDA Fraunhofer first, 2005. Available at `http://www.bbci.de/competition/iii/`.

[8] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Transactions on Rehabilitation Engineering*, 8(4):441–446, 2000.

[9] A. Schlögl. Biosig - an open source software library for biomedical signal processing., 2003-2004.