31st Annual International Conference of the IEEE EMBS
Minneapolis, Minnesota, USA, September 2-6, 2009

Adaptive Schemes Applied to Online SVM for BCI Data Classification

Mohammadreza Asghari Oskoei, John Q. Gan, and Huosheng Hu

Abstract— **This paper evaluates supervised and unsupervised adaptive schemes applied to online support vector machine (SVM) that classifies BCI data. Online SVM processes fresh samples as they come and update existing support vectors without referring to pervious samples. It is shown that the performance of online SVM is similar to that of the standard SVM, and both supervised and unsupervised schemes improve the classification hit rate.**

I. Introduction

Electroencephalogram (EEG) is an electrical signal collected from scalp and represents brain activities. A pattern recognition based brain-computer interface (BCI) discriminates EEG patterns and produces pre-defined commands corresponding to the patterns, to accomplish individuals' intentions in communicating with a computer. Due to subject's brain conditions or environmental changes, EEG signals are non-stationary. This phenomenon necessitates adaptive schemes that modify BCI classification parameters during run-time [1].

In this regard, various methods of supervised and unsupervised adaptive schemes have been applied to BCI systems with LDA [2][3] or GMM [4] classifiers, including Kalman filter based methods for online adaptation [5][6].

A classifier is the core of a pattern recognition based BCI, and online training that involves modifying the classification criteria to cope with changes in signal patterns can be an option to build an adaptive BCI. For certain classifiers, such as support vector machine (SVM), the online training accommodates two crucial issues: updating online Training Data Set (TDS) with valid samples, and applying training during BCI operation. Updating TDS inserts fresh samples into TDS using supervised or unsupervised methods. This demands repeating training process during run-time. Using the whole TDS for run-time training is computationally expensive and cannot satisfy some real-time constraints. Hence, an online algorithm that uses merely fresh samples for the training process and in the meantime keeps the old trained patterns is required.

This paper presents supervised and unsupervised adaptive schemes with the core of online SVM for a BCI system and compares them with the non-adaptive scheme using both synthetic data and real BCI data. The rest of the paper is organized as follows. Section II introduces the online SVM. Adaptive schemes are presented in Section III. Section IV explains the experiments conducted to examine the performance of the adaptive schemes. Finally, Section V contains the conclusion.

M. Asghari Oskoei, PhD Candidate, School of CS and EE, University of Essex, UK, URL: http://privatewww.essex.ac.uk/~masgha/, Email: masgha@essex.ac.uk

J.Q. Gan, Reader, School of CS and EE, University of Essex, Colchester CO4 3SQ, UK, URL: http://dces.essex.ac.uk/staff/jqgan/, Email: jqgan@essex.ac.uk

H. Hu, Professor, School of CS and EE, University of Essex, Colchester CO4 3SQ, UK, URL: http://cswww.essex.ac.uk/staff/hhu/, Email: hhu@essex.ac.uk

II. Online SVM

SVM is a kernel-based approach with a strong theoretical background, which has become a popular tool for machine learning tasks involving classification and regression. It has been successfully applied to many applications, ranging from face identification and text categorization, to bioinformatics and database mining. SVM has been developed in three stages. At first, it was introduced to construct a linear optimal hyperplane with the widest margin between two classes. Then, it was extended to an optimal hyperplane in a feature space induced by a kernel function that covers nonlinear boundaries between classes. Finally, it was equipped to address noisy data by allowing some samples violating the margin between classes [7].

For a two-class data set $(x_1, y_1),\ldots,(x_n, y_n)$, $x_i \in R^d$ and $y_i \in \{\pm 1\}$, separating hyperplanes between two classes in a feature space mapped by $\varphi(x)$ are defined as:

$$w.\varphi(x)+b=0, \quad w\in R^d, b\in R \tag{1}$$

A unique hyperplane that yields the maximum margin of separation between two classes and tolerates misplaced samples with distance ($\xi_i \geq 0$) is constructed by solving the following quadratic programming (QP) problem:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \ , \quad \forall i \ \ y_i(w.x_i+b)\geq 1-\xi_i \tag{2}$$

The constant $C \in [0,\infty]$ is an upper bound for samples that lie on the wrong side of the hyperplane. It works as a controlling parameter to avoid overfitting problem in classification by creating a trade-off between the capacity of the classifier and error in TDS. Given kernel $K(x_i,x_j)=\varphi(x_i).\varphi(x_j)$ and weights $w=\sum\alpha_i\varphi(x_i)$, a way to solve (2) is via its Lagrangian dual that has been simplified to find the multipliers α_i:

$$\max \ J(\alpha)=\sum_i \alpha_i y_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j K(x_i,x_j) \tag{3}$$

$$\sum_i \alpha_i = 0 \ , \ A_i \leq \alpha_i \leq B_i \ , \ A_i=\min(0,Cy_i) \ , \ B_i=\max(0,Cy_i)$$

The objective function in (3) slightly deviates from the standard formulation because it makes the coefficients α_i positive when $y_i = +1$ and negative when $y_i = -1$. Solving (3) helps to construct optimal hyperplane (1) and build the following decision function:

$$f(x)=\sum_{i=1}^{n}\alpha_i K(x,x_i)+b \tag{4}$$

The above decision function addresses the feature space using support vectors (SVs), i.e., samples that make $\alpha_i \neq 0$. SVM has been very successful and widely used because it reliably delivers state-of-the-art classifier with minimal tweaking. Sequential minimal optimization (SMO) is one of the efficient numerical algorithms developed to solve (3). It works by making searches along the direction u starting from vector α, which yields a new vector $\alpha+\lambda_m u$, with

$$\lambda_m = \arg\ \max\ J(\alpha+\lambda u)\ ,\ 0\le\lambda\le\phi(\alpha,u) \tag{5}$$

where $\phi(\alpha,u)$ is an upper bound that ensures the $\alpha+\lambda_m u$ is feasible and u is a random search direction that $\sum_k u_k=0$.

It was observed that the direction search is much faster when its coefficients are mostly zero, hence, SMO uses search directions whose coefficients are all zero except for single +1 and single -1. Practical implementations of SMO, such as LIBSVM [9], rely on a small positive tolerance $\tau>0$, to select a suitable pair (i,j), called 'τ-violating pair', such that $\alpha_i<B_i$, $\alpha_j>A_j$, and g_i - $g_j>\tau$, where g_k is the gradient of $J(\alpha)$ and defined as

$$g_k=\frac{\partial J(\alpha)}{\partial \alpha_k}=y_k-\sum_i \alpha_i K(x_i,x_k) \tag{6}$$

Bordes et al. [8] have developed an iterative implementation of SVM, called LASVM, which suits online applications. LASVM reorganizes SMO direction searches; as such it converges to the solution to (3). It is built by alternating two kinds of direction searches named PROCESS and REPROCESS. PROCESS involves at least one sample that is not already an SV, and potentially can change its multiplier (α_i) so as to make it a new SV. This enables LASVM to update SVs with merely using fresh samples in TDS. REPROCESS involves two samples that already are SVs, and potentially can zero their multipliers to remove one or both of them from current SVs. LASVM, at first, initializes state variables, and then runs online iterations (i.e., PROCESS and REPROCESS) that sequentially visit all the randomly shuffled TDS samples (this may occur in epochs), and finally performs finishing step which is only useful when one limits the number of online iterations.

LASVM handles gracefully noisy data, converges to solutions of the known SVM methods (e.g., LIBSVM [9]), and brings the computational benefits and the flexibility of online learning algorithms. Experimental evidence indicates that LASVM matches the standard SVM in terms of accuracy after a single sequential pass over TDS [8]. LASVM can be used in the online setup where one is given a continuous stream of fresh random samples. The online iterations process fresh samples as they come and update existing SVs without referring to pervious samples. This is called incremental training, and it is a vital requirement to implement the online training with huge data sets, such as BCI data.

III. Adaptive Schemes

An adaptive scheme rebuilds the boundaries between classes during real-time operation. It updates TDS from fresh data (i.e., have not been used for training yet) and applies them for online training (i.e., LASVM online iterations) iteratively. An SVM can be initially trained (i.e., offline training) using pre-collected labeled data. However, supplying run-time TDS is a challenge. This can be conducted using supervised or unsupervised methods.

Although adopting supervised methods, in which labeled data are used for training a classifier, can be a protected option, in real-time applications, it often either too expensive or entirely impossible. For instance, providing true label for continuous stream of BCI data is nearly impossible. However, to have a comparative evaluation, by using pre-collected data, we employed two supervised methods to generate TDS for online training. In the former, called SP1, we applied all the fresh data to train the classifier. The latter method, named SP2, uses the most misclassified samples to generate TDS. In this method, we first classified the fresh data and then selected samples that located further from their true classes. This method neglects misclassified samples that are placed close to the current boundary, and liberates the impact of marginal data that are probably mislabeled. The furthest samples from the boundary between the classes are defined as

$$k^*=\arg\ y_k f(x_k)\le\delta \tag{7}$$

The threshold δ is chosen as the half of the maximum distance of the fresh samples from the boundary.

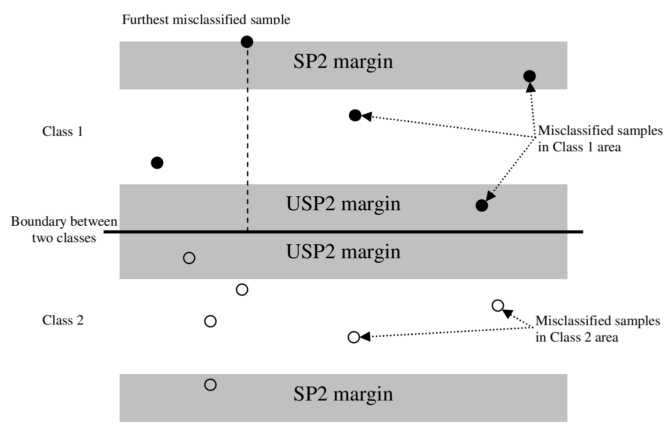

Figure 1 – Margins of SP2 and USP2 around a boundary between two classes

Unsupervised methods employ data samples without true labels. In these methods, the fresh data along with their predicted labels are adopted to update TDS for online training. We again applied two unsupervised methods, named USP1 and USP2. USP1 employs all the fresh data along with their predicted labels (by the current classifier), while USP2 conservatively chooses the samples that are closest to the current boundary. This prevents the classifier from sudden big changes during online training, because of lack of confidence to data labels. The closest samples to the boundary between classes are defined as

$$k^* = \arg \left| f(x_k) \right| \le \delta \tag{8}$$

The threshold δ is chosen as the half of maximum distance of the fresh samples from the boundary. Figure 1 illustrates the margins of SP2 and USP2 around a boundary between two classes.

IV. EXPERIMENTAL RESULTS

As mentioned earlier, providing labels for BCI data in real-time is nearly impossible, hence, we used three sets of pre-collected BCI data as well as forty sets of synthesized data to compare the performance of four proposed adaptive schemes (i.e., SP1, SP2, UPS1, and USP2) with non-adaptive scheme. In all schemes, online SVM with same parameters (i.e., C=1 and γ=0.5) is adopted as the classifier.

At each experiment, we divided a data set into m consecutive subsets, the first subset was used for offline training and the rest were used first for testing and then for online training. In non-adaptive scheme, the accuracy of classification over each subset ($Acc_i^{non\text{-}adaptive}$) was calculated using the SVM trained by the first subset, while in adaptive scheme, the accuracy of classification over each subset ($Acc_i^{adaptive}$) was calculated using the online SVM adapted (i.e., incrementally trained) by the pervious subset. Classification accuracy over each subset was computed by the rate of properly classified samples to all the samples in each subset. The impact of using adaptive scheme compared with non-adaptive scheme was quantified by

$$I = \frac{1}{m} \sum_{i=1}^{m} (Acc_i^{adaptive} - Acc_i^{non-adaptive}) \tag{9}$$

Its positive or negative value represents improvement or degradation of classification hit rate of the applied adaptive SVM, respectively.

A. Synthetic Data

The first experiment employed forty sets of synthetic data, which were designed in such a way that the boundary between the two classes changes smoothly with time. To evaluate the classification performance of the proposed adaptive schemes, each set was divided into eleven consecutive subsets, and the impact of adaptive SVM was individually calculated for each scheme using equation (9). Figures 2 and 3 depict the classification accuracy on two particular data sets, both showing remarkable improvements by the four adaptive schemes. Figure 4 illustrates the average impact of the four adaptive schemes over forty data sets. As can be seen, the adaptive schemes have improved classification accuracy by about 12%.

B. BCI Data

The second experiment employed the three sets of BCI data in “Data Set V” for the BCI Competition III. At first, we compared LASVM with LIBSVM in classification of the BCI data. Both used the same kernel and 2-fold cross validation method. The experimental results illustrated in Table 1 indicate that LASVM in two epochs performs comparably to LIBSVM in terms of classification accuracy.

Set	LIBSVM	LASVM (epochs=2)
#1	85.46 %	84.38 %
#2	77.96 %	77.14 %
#3	77.23 %	79.52 %

Table 1 – Cross-validation accuracy (%) of LIBSVM and LASVM applied on the three sets of BCI data

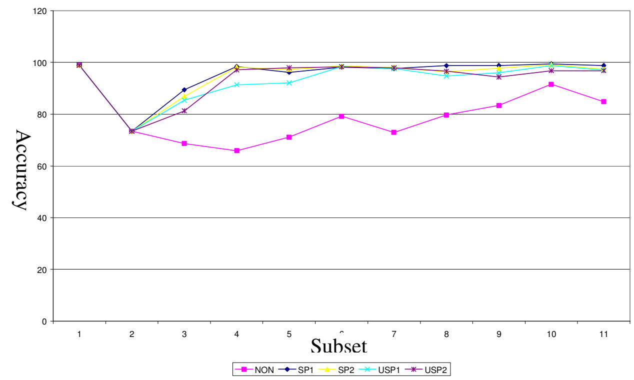

Figure 2 – Classification hit rate over eleven subsets of set three of synthetic data using the four adaptive plus the non-adaptive SVM

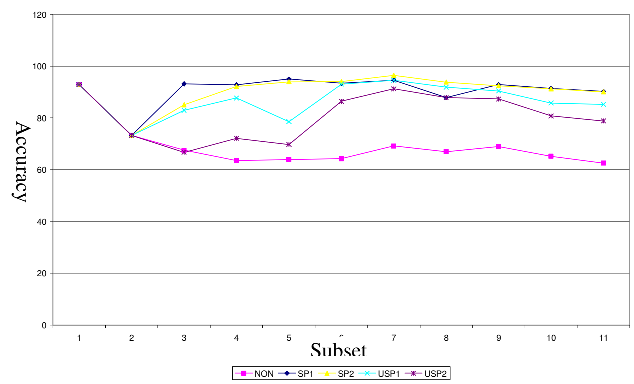

Figure 3 – Classification hit rate over eleven subsets of set nine of synthetic data using the four adaptive plus the non-adaptive SVM

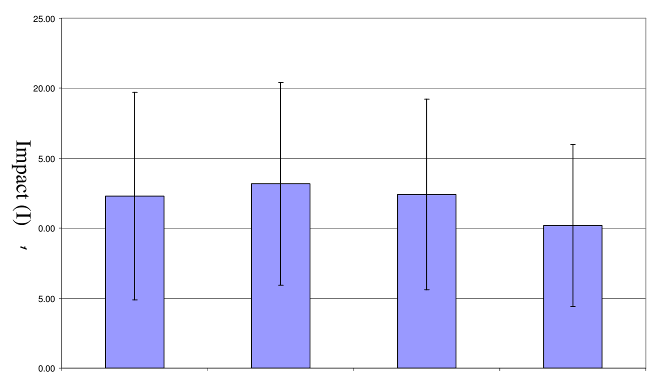

Figure 4 - Average of impacts (%) yielded by applying adaptive schemes on 40 sets of synthesized data

To evaluate the adaptive schemes, each data set was divided into ten consecutive subsets of equal size. The first subset was used for offline training, and the remaining subsets used for online training. Figures 5 and 6 depict the classification performance of the adaptive schemes as well as non-adaptive scheme over two data sets. As it is shown adaptive schemes improve the classification hit rate. Figure 7 shows the average of accuracy improvement, calculated using equation (9), after applying adaptive schemes. As can be seen, both the supervised (i.e., SP1 and SP2) and the first unsupervised schemes (i.e., USP1) improve the accuracy, but the second unsupervised method (i.e., USP2) degrades the classification accuracy.

V. Conclusion

Experimental results show that the online SVM (LASVM), which employs fresh samples only for online training without referring to pervious samples, significantly reduces the training time compared to the well-known SVM (LIBSVM) and produces similar classification accuracy. Moreover, the proposed adaptive schemes based on online SVM improve in general the classifcation hit rate on both the synthesized data and real BCI data.

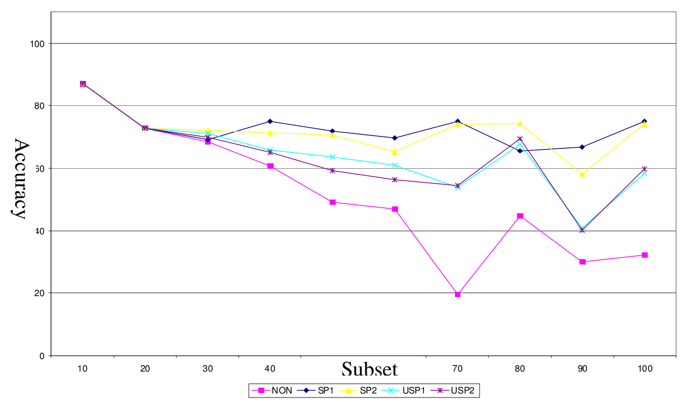

Figure 5 – Results of adaptive schemes and non-adaptive scheme applied on the BCI data set 3

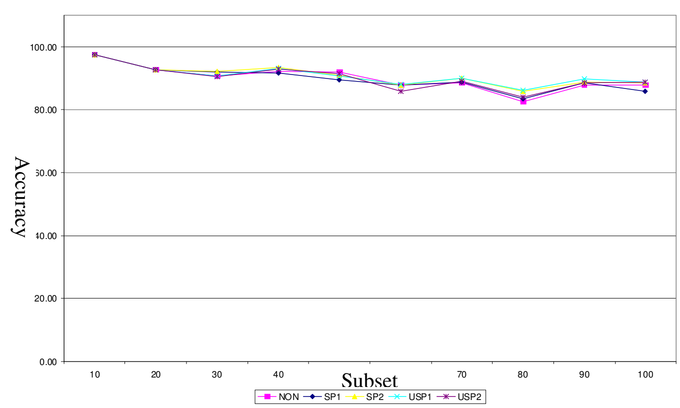

Figure 6 – Results of adaptive schemes and non-adaptive scheme applied on the BCI data set 1

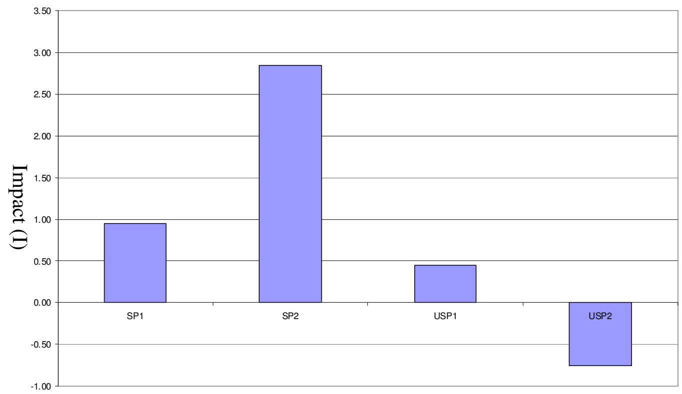

Figure 7 – Average of impact (%) yielded by applying adaptive schemes on 3 sets of BCI data

Acknowledgment

The authors would like to thank Bashar Awwad Shiekh Hasan for helping with the experiments. This work is partly supported by the UK EPSRC under grant EP-D030552-1

References

[1] J del R. Millan, "On the need for on-line learning in brain computer interfaces," *Proc. of IJCNN'2004*, Budapest, Hungary, 2004, pp. 2877-2882.

[2] C.S.L. Tsui and J.Q. Gan, "Asynchronous BCI control of a robot simulator with supervised online training," *Proc. Of International Conference on Intelligent Data Engineering and Automated Learning (IDEAL2007)*, Birmingham, UK, 2007, pp. 125-134.

[3] C.S.L. Tsui and J.Q. Gan, "Comparison of three methods for adapting LDA classifiers with BCI applications," *The 4th International Workshop on Brain-Computer Interfaces*, Graz, Austria, 2008, pp. 116-121.

[4] B. Awwad Shiekh Hasan and J.Q. Gan, "Unsupervised adaptive GMM for BCI", *Proc. of the 4th International IEEE EMBS Conference on Neural Engineering*, Antalya, Turkey, 2009.

[5] J.W. Yoon, S.J. Roberts, M. Dyson, and J.Q. Gan, "Sequential Bayesian estimation for adaptive classification," *Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems (MFI2008)*, Seoul, Korea, 2008, pp. 601-605.

[6] C.S.L. Tsui, J.Q. Gan, and S.J. Roberts, "A self-paced brain-computer interface for controlling a robot simulator: An online event labelling paradigm and an extended Kalman filter based algorithm for online training," *Medical & Biological Engineering & Computing,* vol. 47, no. 3, pp. 257-265, 2009.

[7] M. Asghari Oskoei and H. Hu, "Support vector machine-based classification scheme for myoelectric control applied to upper limb," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 8, pp. 1956-1965, 2008.

[8] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *Journal of Machine Learning Research*, vol. 6, pp. 1579-1619, September 2005.

[9] C-C. Chang and C-J Lin, "LIBSVM: a library for support vector machines," at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.