Co-adaptivity in Unsupervised Adaptive Brain-Computer Interfacing: a Simulation Approach

Martin Spüler, Wolfgang Rosenstiel
Wilhelm-Schickard-Institute for Computer Sciencen
University of Tübingen
Tübingen, Germany
Email: {spueler,rosen}@informatik.uni-tuebingen.de

Martin Bogdan
Computer Engineering
University of Leipzig
Leipzig, Germany
Email: bogdan@informatik.uni-leipzig.de

***Abstract*—A Brain-Computer Interface (BCI) allows a user to control a computer by pure brain activity. Due to the non-stationarity of the recorded brain signals, the BCI performance tends to decrease over time. Recently, adaption of the BCI has been proposed as a means to counter non-stationarity and help to stabilize the BCI performance. Since most adaption methods for BCI are analysed in an offline setting, one important factor is not taken into account: that also the user is adapting. While online experiments take into account the adaptive user, a comparison of different classifiers with the same data is always biased towards the method that was used for feedback and thereby does not allow a proper evaluation of the classifier in a co-adaptive environment. To solve this problem, we propose a simulation approach that simulates an adapting BCI user and allows to test and compare different adaptive algorithms considering the co-adaptivity between BCI and user. With this approach we can also show, under which conditions an adaption of the BCI improves performance and when the adaptive BCI and the adaptive user hinder each other and lead to a decrease in BCI performance.**

Keywords-Brain-Computer Interface(BCI); unsupervised adaption; co-adaptivity.

I. INTRODUCTION

A Brain-Computer Interface (BCI) classifies the brain signals of a user, thereby giving him the possibility to communicate or control a computer by pure brain activity. One problem for current BCI systems is the high non-stationarity of the recorded brain signals, which causes the BCI performance to deteriorate over time. Adaption of the BCI classifier has been proposed as a means to counter these non-stationarities and to stabilize or even improve the BCI performance [1]. There exist numerous publications that show different adaption methods for BCI to increase performance in an offline analysis [2]–[5]. But an evaluation of adaptive algorithms in an offline setting is not advisable, since it does not take the user into account, who learns and adapts to the BCI. Thereby it is unclear if the adaptive BCI makes it harder for the user to learn BCI control or if the learning of the user, who adapts his control strategy, might hamper the learning of the adaptive BCI algorithm.

To overcome this problem and to take into account the learning user, online experiments have to be performed. This was already done with a supervised adaption of the classifier [6]–[8], which is not practical, since supervised adaption can not be used in practical BCI applications due to the missing class labels. So far, Vidaurre et al. are the only ones to show an unsupervised online adaption of a BCI classifier [9] with the result that an unsupervised adaption might not be feasible for all subjects. While these online experiments allow a suitable evaluation of a BCI adaption, they still do not allow a fair comparison between adaptive and non-adaptive methods to assess the benefit of BCI adaption, since the non-adaptive methods were evaluated offline on the data recorded from the online experiments. The results are thereby biased towards the adaptive method used during the online experiment.

To compare two different adaptive and non-adaptive methods for BCI, both have to be evaluated online. Due to external factors like inter- and intra-subject variability in BCI performance and learning, a large subject-population would be needed that goes beyond what is used in today's BCI research.

So far, there is no work that specifically addresses this aspect of co-adaptivity in a BCI.

As an approach to test different adaptive methods in an online setting with a learning user, we propose the use of a simulation method, that allows to simulate online BCI sessions with an adaptive user under different environments with different parameters in a fast and cost-efficient way.

In the following, we will explain, how the simulation works, what parameters are available to adjust the user's behaviour, and the environmental influences and we will demonstrate the influence of parameter changes. Based on an adaptive BCI classifier, we will show how the simulations can be used to evaluate the mutual influence between the user and the BCI, and under which conditions the user benefits from an adaptive BCI classifier or which conditions lead to decreasing BCI performance.

In addition, we will show, that this simulation approach can also be used to answer other questions like: can the user learn to control an adaptive BCI, if neither the user nor the BCI have any prior knowledge?

 ISBN: 978-1-61208-218-9

II. Methods

To simulate the interaction between an (adaptive) BCI and the user, a genetic algorithm was used to simulate the learning user and the BCI classifier was used as a fitness function for the genetic algorithm. An overview of the general concept is depicted in Figure 1. A motor imagery BCI, where the BCI was controlled by left hand motor imagery and right hand motor imagery was used as archetype for the simulated BCI.

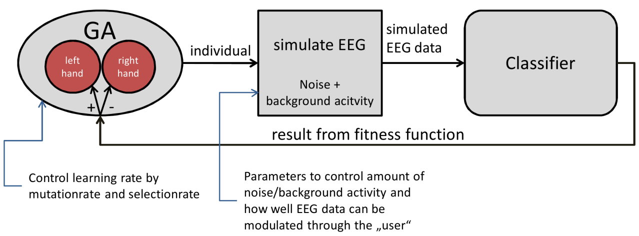

Figure 1. Overview of the simulation of two adaptive systems

The genetic algorithm consists of two populations: population P_L belonging to *left hand motor imagery* and population P_R belonging to *right hand motor imagery*. Both populations are filled with individuals, which can be seen as representations of the specific state of brain activity. Each individual $I_N = (i_N, f_N)$ consists of a vector $i_N \in \mathbb{R}^n$ and a fitness value $f_N \in \mathbb{R}$.

To control the BCI, a new individual I_N is created for each trial. Depending on the target class for this trial, the I_N is either created out of P_L if the target is the class associated with *left hand motor imagery* or vice versa. For the creation of a new individual, two parent individuals (I_1, I_2) are drawn randomly from the population, with individuals with higher fitness having a higher probability to be chosen. For the creation of the new vector i_N, it is chosen randomly, which parts are filled from i_1 and which from i_2. On average, i_1 and i_2 fill half of i_N. Depending on the parameter settings for the mutation rate, a noise vector is added to i_N to introduce mutations. The amplitude of the noise vector also depends on the mutation rate. Since I_N will be added to the population later, a random I_D is selected out of the population, from which the parents of I_N were drawn, and removed. Individuals with lower fitness have a higher probability to be drawn.

I_N is then used to generate an artificial EEG signal according to i_N. The EEG signal is then preprocessed and classified by the BCI classifier. The BCI classifier outputs $c_N \in \mathbb{R}$ with $c_N \geq 0$ when *left hand motor imagery* is classified and $c_N < 0$ when *right hand motor imagery* is classified. The BCI classifier then serves as a fitness function for the genetic algorithm with $f_N = c_N$ being the fitness value if the individual stems from P_L and $f_N = -c_N$ when the individual stems from P_R. At last the individual $I_N = (i_N, f_N)$ is added to its population.

A. Simulation of EEG data and preprocessing

For the generation of the EEG data, the vector $i_N \in \mathbb{R}^{20}$ is used as current state of brain activity. For the simulation of the EEG data a samplingrate of 100 Hz is used and the two electrodes C3 (located over the left motor cortex) and C4 (located over the right motor cortex) are simulated. The values $i_{Nx}, x = \{1, 3, 5, \ldots, n-1\}$ represent the brain activity in the left motor cortex, while the values $i_{Nx}, x = \{2, 4, 6, \ldots, n\}$ represent the brain activity in the right motor cortex.

The brain activity in the left motor cortex is used for generation of EEG data for C3 and the activity in the right motor cortex for the generation of EEG data for C4. The 10 values are used to modulate the frequency spectrum in the range from 3 to 30 Hz. To simulate background activity and noise as it is typically present in real EEG recordings, pink $(1/f)$ noise and white noise are added with amplitudes that can be predefined in the settings. Also the noise can change over time to simulate a covariate shift [10].

After generation of the EEG signal, the frequency spectrum for both electrodes is extracted by an autoregressive model with order 10. The power spectrum in the range from 1 to 50 Hz in bins with width of 1 Hz are used as features for the classification.

An example for the different steps of simulation of the EEG data and preprocessing is visualized in Figure 2.

B. BCI classifier

In the following, the two methods are introduced, that have been used for classification in the simulated BCI.

1) SimpleMu: SimpleMu is a very simple classifier that was used to test the parameters settings of the genetic algorithm with a simple non-adaptive classifier. It outputs the difference in the power spectrum between C3 and C4 in the Mu range (7 to 12 Hz). Assume $A_{i,b}$ is the power for channel i at frequency bin b, with $i = 1$ for C3 and $i = 2$ for C4, the output of the classifier is calculated as follows:

$$c_N = \sum_{b=7}^{12} A_{1,b} - A_{2,b} \tag{1}$$

2) K-means: K-means clustering [11] was chosen as an algorithm that allows an unsupervised adaption of the BCI, as well as an unsupervised calibration.

Given a $k \in \mathbb{N}\backslash\{0, 1\}$ and n datapoints $(x_1, x_2, \ldots, x_n)$ with $x_i \in \mathbb{R}^d$, k-means tries to partition the n datapoints into k $(k \leq n)$ cluster or sets $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$, while minimizing the within-cluster sum of squares:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2 \tag{2}$$

μ_i represent the mean of all datapoints in cluster S_i. For the initialisation of k-means an initial set of k means

 ISBN: 978-1-61208-218-9

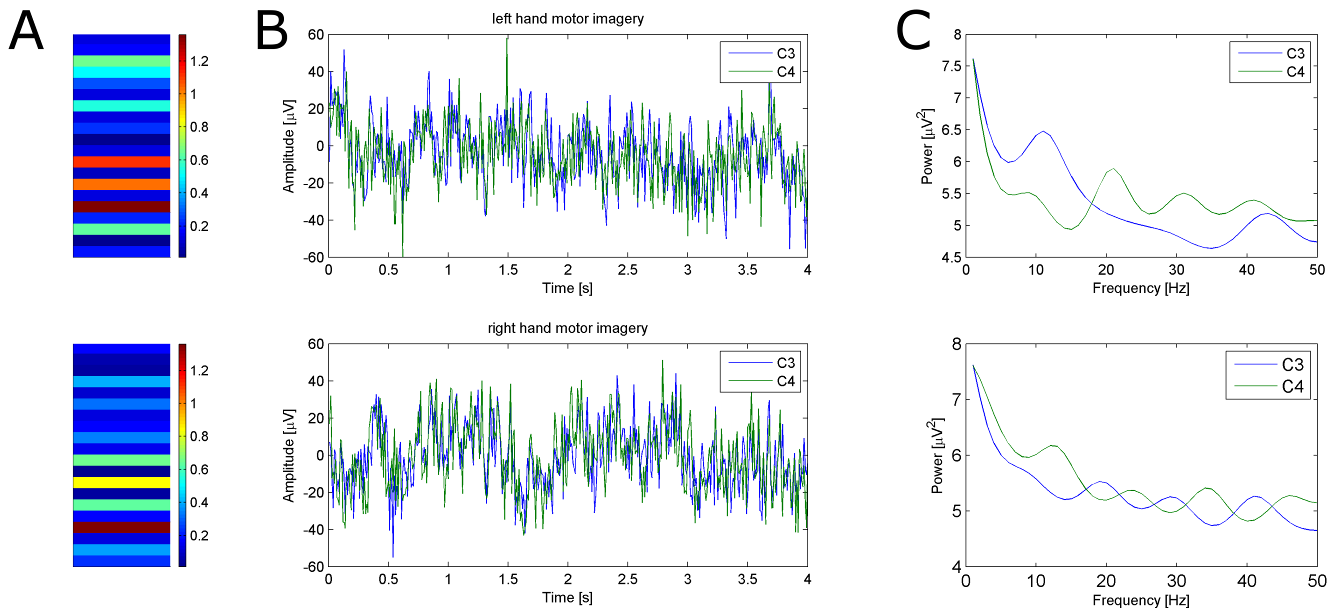

Figure 2. Simulation of EEG data: A) Vector for one individual of population for left hand imagery (top) and right hand imagery (bottom) B) Corresponding EEG-signal at electrodes C3 and C4 C) Corresponding frequency spectrum at electrodes C3 and C4

$(m_1^{(1)}, m_2^{(1)}, \ldots, m_k^{(1)})$ has to be given. When a new data-point x_{n+1} is added to adapt the classifier, it has to be ensured, that the mean of the existing clusters are adapted and no new clusters are created from scratch. Therefore the means $(\mu_1^{(n)}, \mu_2^{(n)}, \ldots, \mu_k^{(n)})$ from the previous result of k-means are used as initial set $(m_1^{(n+1)}, m_2^{(n+1)}, \ldots, m_k^{(n+1)})$.

In the following, $k = 2$ is used, since the BCI control is simulated with only 2 classes.

For the unsupervised calibration, the initial means are chosen randomly. Due to the fact, that k-means is a pure clustering approach without any knowledge of the true class labels, the clusters can represent the classes correctly, but the clusters can be associated with the wrong class label. In this case the BCI would always choose the wrong class and do exactly the opposite of what the user is intending to. Since a human user would recognize this fact and just correct the mistake of the BCI by switching the two imagery classes himself, such a behaviour had to be implemented into the genetic algorithm. Therefore, the individuals of the two populations P_L and P_R were completely switched if the accuracy fell below a threshold of 30 %, which should mimic the behaviour of the human user switching his mental imagery classes.

Although k-means does not offer a supervised calibration of the BCI by default, k-means can be performed on all individuals in P_L and P_R. The initial means $m_1^{(1)}$ and $m_2^{(1)}$ are set to the mean of all individuals of P_L and P_R, respectively, and the clusters can be associated with the correct class labels, to simulate a supervised calibration of the BCI.

A new trial x_t is then classified by calculating

$$c_N = \|\mathbf{x}_t - \boldsymbol{\mu}_L\| - \|\mathbf{x}_t - \boldsymbol{\mu}_R\| \tag{3}$$

where μ_L is the mean of the cluster for left hand motor imagery and μ_R the mean of the cluster for right hand motor imagery.

III. Parameter Changes and Their Effect

Different parameters can be used to test different settings and adjust the behaviour of the genetic algorithm. In the following, the important parameters are presented and their effect is demonstrated with the non-adaptive simpleMu classifier. It is assumed that the genetic algorithm has prior knowledge how to control the BCI, since it is also explained to BCI users, that they should control the BCI by motor imagery.

If not stated otherwise, a BCI session with 2000 trials was simulated, where both classes were evenly distributed. 10 sessions were simulated resulting in a vector for each session, where the outcome of each trial was marked with 1 if correct and 0 if wrong. The accuracy for one time point during this session was calculated by sliding a Hanning window with length 251 over the result-vector.

A. Learning rate

One parameter that strongly influences the behaviour of the genetic algorithm is the so called *learning rate*. It is one parameter, to be set ≥ 0, which is proportional to the mutation rate and to the weighting of the fitness function, when individuals with the highest or lowest fitness are randomly selected either for reproduction or elimination. While the algorithm will not learn at all with a learning rate of 0, it should learn faster and adapt faster to the BCI with a higher learning rate.

The effect of two exemplary learning rates is displayed in Figure 3.

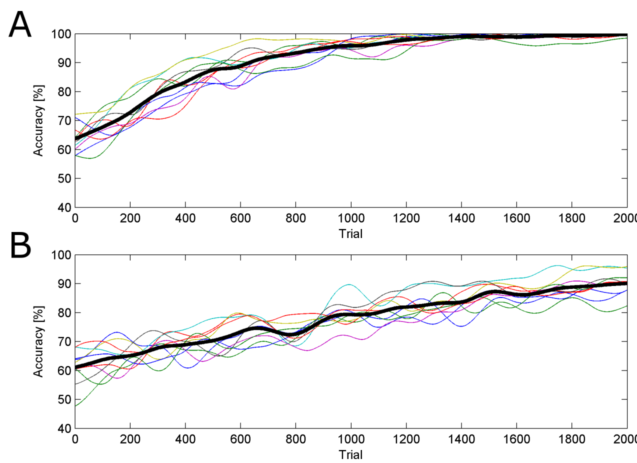

Figure 3. Result of a simulation with two different learning rates of 2 (A) and 0.5 (B). The thin colored lines represent different simulated sessions and the thick black line represents the average accuracy over all simulation runs.

A comparison of different learning rates with values ranging from 0.01 to 2 is shown in Figure 4. It is clear that the simulated user learns and adapts to the BCI and that it adapts faster to the BCI with a higher learning rate.

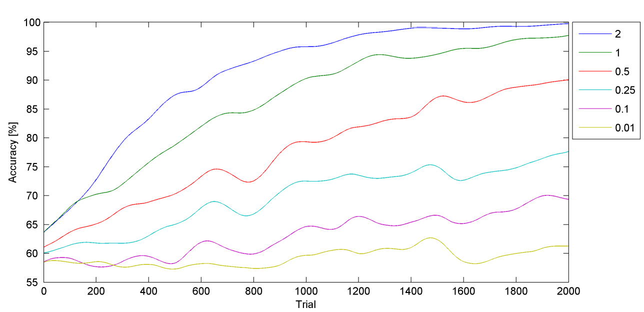

Figure 4. Average accuracy during simulations with different learning rates.

B. Initial performance

Another parameter that can be changed for the simulations is the initial performance of the BCI user. The initial performance is modeled by drawing the initial individuals for the populations P_L and P_R from two different distributions. Each distribution has 20 dimensions and for 18 dimensions the mean and standard deviation of these dimensions is equal across both distributions. For 2 dimensions, which roughly correspond to the alpha-band in the left or right motor cortex, the mean of the distributions differs by d_x times the standard deviation. The higher the d_x, the higher is the initial performance.

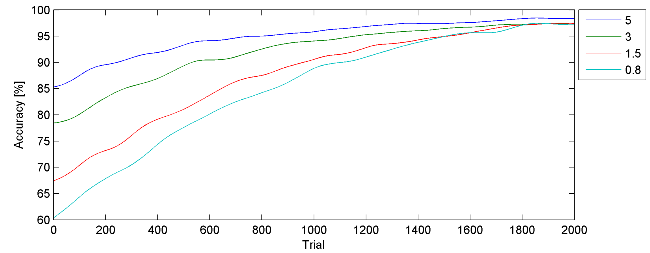

Figure 5. Average accuracy during simulations with different distance d_x between the two populations P_K and P_R, which results in different initial performance. Learning rate was set to 1.

Figure 5 shows results from simulations with learning rate 1 and different values for d_x. For each parameter combination 50 sessions were simulated and the average is displayed.

C. Signal-to-noise ratio

One problem when recording EEG for controlling a BCI, is the amount of noise and background activity, that is picked up by the EEG. Therefore EEG has a bad signal-to-noise ratio. To evaluate the adaption of the simulated user under different signal-to-noise ratios, signal-to-noise ratio was introduced as another parameter that affected how well the genetic algorithm was able to modulate the brain activity compared to the amplitude of the noise in the EEG signal. The result of simulations with different signal-to-noise ratios is shown in Figure 6. Due to these results, a value of 0.2 was chosen as a realistic value to be used for all following simulations.

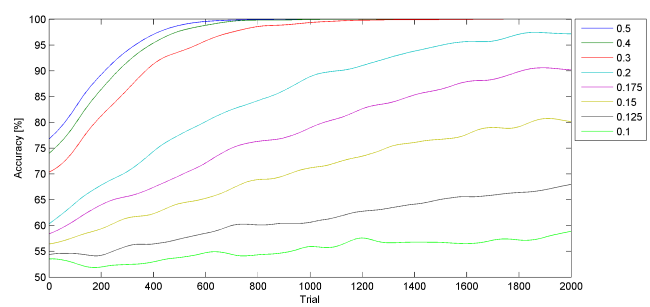

Figure 6. Results of simulations with different signal-to-noise ratios. For each value 50 sessions have been simulated and the average is displayed.

D. Covariate shift

To simulate the effect of non-stationarity in EEG signals, a covariate shift [10] of the data was introduced by increasing the amount of pink noise every trial by a specified value. To test how well the simulated user adjusts to the covariate shift, different extents of covariate shift were tested with 50 simulated sessions per value and a learning rate of 1. To simulate the effect of a covariate shift on BCI users with low initial performance and high initial performance, simulations were conducted with $d_x = 0.8$ and $d_x = 5$. The results for this simulations are shown in Figure 7.

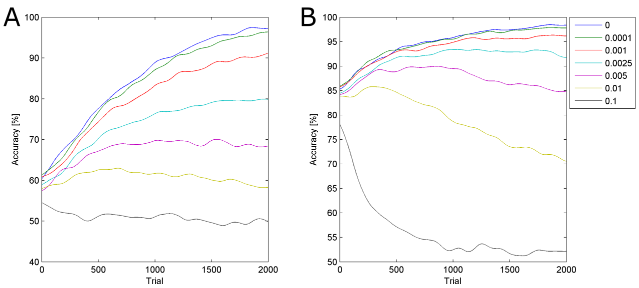

Figure 7. Results of simulations with different values for the amount of covariate shift and a learning rate of 1. A) for users with low initial performance B) for users with high initial performance

It can be seen that a higher covariate shift makes it harder for the BCI user to adapt to the changes and if the covariate-shift is too large, the user can't adapt and the BCI performance drops. This drop in BCI performance is especially visible for BCI users with high initial performance.

IV. Results

A. Adaptive vs. non-adaptive classifier

To simulate the effect of classifier adaption on the BCI performance, k-means was used as a classifier and a supervised calibration was simulated. For the adaptive case, the classifier was adapted in an unsupervised manner, while the classifier was static for the non-adaptive case.

To evaluate the benefit of adaptive classification, 500 sessions were simulated each with different values d_x and different learning rates whereas the mean accuracy over the whole session was taken as performance measure. The average performance difference between the adaptive classifier and the non-adaptive one, is shown in Table I. While the performance gets worse for low d_x with low learning rate and high d_x with high learning rate, the user benefits from the adaptive classifier when d_x is high with a low learning rate or with a high d_x and a low learning rate. To check if there is a significant difference between the results for the adaptive classifier and the results for the non-adaptive classifier, Wilcoxon's ranksum test was performed for each combination of d_x and learning rate. The parameter combinations with significant effects ($p < 0.01$) are marked in bold in Table I.

A 3-way ANOVA was applied to the values and shows, that the use of the adaptive classifier significantly ($p < 0.0001$) increases the total average performance by 1.41 %. The 3-way ANOVA also shows, that there are significant interactions between the factors learning rate and adaption ($p < 0.0001$), as well as a significant interaction between the factors d_x and adaption ($p < 0.0001$).

	learning rate						
d_x	0.1	0.25	0.5	1	2	4	8
0.8	**-0.020**	**-0.021**	**-0.021**	0.000	0.029	**0.085**	**0.192**
1.5	-0.001	**0.031**	**0.021**	**0.020**	**0.011**	**0.026**	**0.085**
3	0.014	**0.018**	**0.020**	-0.002	-0.014	-0.043	**0.006**
5	**0.026**	**0.020**	**0.016**	0.005	-0.025	**-0.074**	-0.005

Table I

Difference between adaptive and non-adaptive BCI classifier. Positive difference means a higher accuracy with the adaptive classifier. If there is a significant difference ($p < 0.01$) between the results for the adaptive classifier and the results with the non-adaptive classifier, the value has been marked bold.

B. Influence of covariate shift

To evaluate the effect of covariate shift on the benefit of adaptive classification, simulations were run with different amounts of covariate shift. For different combinations of d_x and learning rates 100 sessions were simulated each with a covariate shift ranging from 0 to 0.1. Since a 4-way ANOVA shows no significant ($p > 0.05$) interaction between learning rate and adaption, when considering different amounts of covariate shift, the results were averaged over the learning rates. Figure 8 shows the average performance improvement by adaptive classification with different amounts of covariate shift. It can be seen that it depends on the learning rate and the amount of covariate shift if BCI performance is increased by adaption of the classifier.

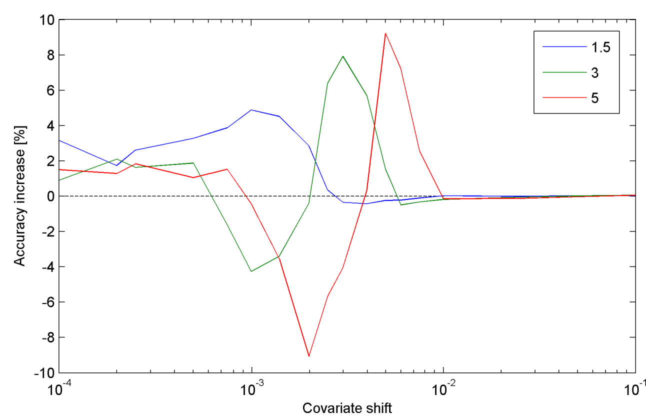

Figure 8. Results from simulations with different amounts of covariate shift with different initial performance.

C. Co-adaptivity without prior knowledge

There still is the question to be answered, if BCI control can be achieved if neither the user nor the BCI have any prior knowledge on how control might work or how the two classes can be differentiated. Therefore, simulations were run, in which the individuals for both populations P_L and P_R were drawn from the same distribution ($d_x = 0$) simulating that the user has no prior knowledge on how to control the BCI. Since the k-means classifier is initialized randomly,

 ISBN: 978-1-61208-218-9

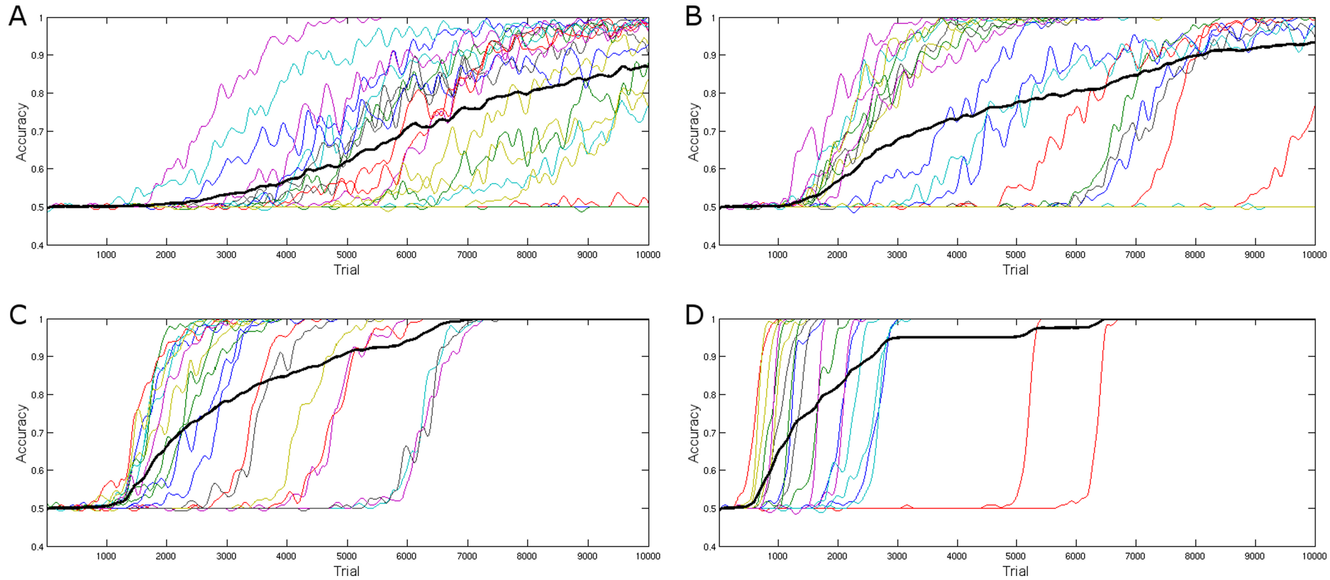

Figure 9. Co-adaptivity without prior knowledge. Performance during the first 10000 trials with a learning rate of A: 0.5 B: 1 C: 2 D: 4. Colored lines show the result of one session, while the black line shows the average performance.

the classifier has also no prior knowledge on what data to expect.

For different learning rates 20 sessions were simulated, with one session having 10000 trials. The results show, that regardless of the learning rate, performance can be achieved, but with a higher learning rate BCI control is achieved earlier. While with the same learning rate in some sessions BCI control was achieved very fast, it took longer in other sessions. At the end of the session significant BCI control was achieved in 45 % of the sessions with learning rate 0.25, 80 % with learning rate 0.5, 90 % with a learning rate of 1 and 100 % with learning rates of 2 and 4. Although BCI control was not achieved in all sessions with lower learning rate, the results show that the simulated user would still gain BCI control if more trials were performed i.e., for longer BCI sessions. The results from the simulations with different learning rates are displayed in Figure 9.

V. Discussion

At first it needs to be discussed, why we used the presented approach and think of it to be an appropriate model for a learning user. The approach by using a genetic algorithm does not try to resemble the biological processes involved in learning, but tries to model the behavioural processes and aspects involved in human learning. Learning to control a BCI is skill learning [12], which is learned by a human through reinforcement [13]. So, a human learns to control a BCI by trying different strategies and keeping those strategies that maximize the reward, which is a high BCI accuracy. Slight variations in the human's actions are introduced either voluntarily or involuntarily. If these variations lead to higher reward they are positively reinforced and thereby used more often, while variations which decrease the reward are negatively reinforced and thereby used less often. The genetic algorithm essentially does the same [14]; slight variations are introduced through mutation. Individuals with lower fitness are removed, which negatively reinforces behaviour that leads to poor BCI control and individuals with higher fitness are allowed to reproduce, whereby behaviour that leads to good BCI control is positively reinforced.

With this simulation approach we have presented a method that allows to simulate a BCI user that learns to control the BCI and adapts. By changing the parameters of the simulation it allows to adjust the simulations closer to real-life conditions. While a learning rate of 0.1 to 0.25 seems to be suitable to simulate learning of EEG-based BCI control [15], ECoG allows for faster learning of BCI control [16] and learning rates of 0.5 to 1 could be used to simulate ECoG-based BCIs.

The parameter d_x can be used to adjust the initial performance and $1.5 \leq d_x \leq 5.0$ can be used to set the initial BCI accuracy to a value that resembles the average performance reached by most users [17], [18].

Comparing simulation results for an adaptive and a non-adaptive classifier shows that most users will benefit from an adaptive classifier. But for user with low initial BCI performance ($d_x = 0.8$, respectively a BCI accuracy $< 65\%$) an unsupervised adaption does not increase or even decrease the BCI performance, which is in line with the findings by Vidaurre et al. [9].

Regarding the use of unsupervised adaption as a means to alleviate non-stationaries and thereby improve BCI accuracy under a covariate shift, the simulations show that adaption of the classifier can be used to alleviate non-stationarity and thereby improve BCI performance. But unsupervised adaption can also have a negative effect depending on the amount of covariate shift and the initial BCI performance of the user. The simulations also show that BCI control can be achieved if neither the BCI nor the user has any prior knowledge.

VI. CONCLUSION

In this paper, we have proposed a method that uses genetic algorithms to simulate the mutual interaction between a learning user and an adaptive BCI. We have shown the presented approach to be a viable method to study the interaction of 2 learning systems, namely the adaptive BCI and the learning user. It can be used to test and evaluate new adaptive classification methods in a co-adaptive environment and test if the use of a specific adaptive method is always beneficial or under which conditions a specific adaptive method should not be used. Due to the rather simple approach of generating artificial EEG data, it cannot replace an offline analysis on real human EEG data. But, simulations can be used in addition to an offline analysis, to investigate the behaviour of a specific adaptive algorithm in a co-adaptive environment and optimize adaptive algorithms to perform better in cooperation with a learning BCI user.

ACKNOWLEDGMENT

This study was funded by by the German Federal Ministry of Education and Research (BMBF, BFNT F*T, Grant UTü 01 GQ 0831).

REFERENCES

[1] J. d. R. Millán, “On the need for on-line learning in brain-computer interfaces,” in *Proceedings of the International Joint Conference on Neural Networks*, Budapest, Hungary, 7 2004, iDIAP-RR 03-30.

[2] J. Blumberg, J. Rickert, S. Waldert, A. Schulze-Bonhage, A. Aertsen, and C. Mehring, “Adaptive classification for brain computer interfaces,” in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, aug. 2007, pp. 2536 –2539.

[3] A. Satti, C. Guan, D. Coyle, and G. Prasad, “A covariate shift minimisation method to alleviate non-stationarity effects for an adaptive brain-computer interface,” in *Pattern Recognition (ICPR), 2010 20th International Conference on*, Aug. 2010, pp. 105 –108.

[4] C. Vidaurre, A. Schlögl, B. Blankertz, M. Kawanabe, and K. r. Müller, “Unsupervised adaptation of the lda classifier for brain-computer interfaces,” in *Proceedings of the 4th International Brain-Computer Interface Workshop and Training Course 2008*, 2008, pp. 122–127.

[5] G. Liu, D. Zhang, J. Meng, G. Huang, and X. Zhu, “Unsupervised adaptation of electroencephalogram signal processing based on fuzzy c-means algorithm,” *International Journal of Adaptive Control and Signal Processing*, 2011. [Online]. Available: http://dx.doi.org/10.1002/acs.1293

[6] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. Principe, and J. Sanchez, “Coadaptive brain-machine interface via reinforcement learning,” *Biomedical Engineering, IEEE Transactions on*, vol. 56, no. 1, pp. 54–64, 2009.

[7] G. Gage, K. Otto, K. Ludwig, and D. Kipke, “Co-adaptive kalman filtering in a naive rat cortical control task,” in *Engineering in Medicine and Biology Society, 2004. IEMBS ’04. 26th Annual International Conference of the IEEE*, vol. 2, sept. 2004, pp. 4367 –4370.

[8] C. Vidaurre, C. Sannelli, K.-R. Müller, and B. Blankertz, “Co-adaptive calibration to improve bci efficiency,” *Journal of Neural Engineering*, vol. 8, no. 2, p. 025009, 2011.

[9] C. Vidaurre, M. Kawanabe, P. von Bünau, B. Blankertz, and K. Müller, “Toward Unsupervised Adaptation of LDA for Brain-Computer Interfaces,” *Biomedical Engineering, IEEE Transactions on*, vol. 58, no. 3, pp. 587 –597, March 2011.

[10] M. Sugiyama, M. Krauledat, and K.-R. Müller, “Covariate shift adaptation by importance weighted cross validation,” *J. Mach. Learn. Res.*, vol. 8, pp. 985–1005, Dec. 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1314498.1390324

[11] J. A. Hartigan, *Clustering Algorithms*, 99th ed. New York, NY, USA: John Wiley & Sons, Inc., 1975.

[12] E. Curran, “Learning to control brain activity: A review of the production and control of EEG components for driving braincomputer interface (BCI) systems,” *Brain and Cognition*, vol. 51, no. 3, pp. 326–336, Apr. 2003.

[13] W. T. Fu and J. R. Anderson, “From recurrent choice to skill learning: a reinforcement-learning model.” *J Exp Psychol Gen*, vol. 135, no. 2, pp. 184–206, May 2006.

[14] J. J. Grefenstette, D. E. Moriarty, and A. C. Schultz, “Evolutionary algorithms for reinforcement learning,” *Journal Of Artificial Intelligence Research*, vol. 11, pp. 241–276, 1999.

[15] J. R. Wolpaw and D. J. Mcfarland, “Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans,” vol. 101, no. 51. National Academy of Sciences, Dec. 2004, pp. 17 849–17 854.

[16] G. Schalk, K. J. Miller, N. R. Anderson, J. A. Wilson, M. D. Smyth, J. G. Ojemann, D. W. Moran, J. R. Wolpaw, and E. C. Leuthardt, “Two-dimensional movement control using electrocorticographic signals in humans,” *Journal of Neural Engineering*, vol. 5, no. 1, p. 75, 2008.

[17] C. Vidaurre and B. Blankertz, “Towards a Cure for BCI Illiteracy,” *Brain Topography*, vol. 23, no. 2, pp. 194–198, 2010.

[18] B. Blankertz, C. Sannelli, S. Halder, E. M. Hammer, A. Kübler, K.-R. Müller, G. Curio, and T. Dickhaus, “Neurophysiological predictor of smr-based bci performance,” *NeuroImage*, vol. 51, no. 4, pp. 1303 – 1309, 2010.