35th Annual International Conference of the IEEE EMBS
Osaka, Japan, 3 - 7 July, 2013

Semi-Supervised Adaptation in SSVEP-Based Brain-Computer Interface Using Tri-training

Thomas Bender[1], Troels W. Kjaer[2], Carsten E. Thomsen[3], Helge B. D. Sorensen[1] and S. Puthusserypady[1]

***Abstract*— This paper presents a novel and computationally simple tri-training based semi-supervised steady-state visual evoked potential (SSVEP)-based brain-computer interface (BCI). It is implemented with autocorrelation-based features and a Naïve-Bayes classifier (NBC). The system uses nine characters presented on a 100 Hz CRT-monitor, three scalp electrodes for signal acquisition, a gUSB-amp for preamplification and two PCs for data-processing and stimulus control respectively. Preliminary test results of the system on nine healthy subjects, with and without tri-training, indicates that the accuracy improves as a result of tri-training.**

***Index Terms*— Brain-Computer Interface, Steady-State Visual Evoked Potentials, Tri-training, Autocorrelation, Naïve-Bayes Classifier**

I. INTRODUCTION

Brain computer interface (BCI) is an alternative/direct communication pathway between the brain and an external device, without having to go through the usual neuromuscular pathways [1]. Imagine being alert and aware of your environment, but unable to move, speak or express yourself due to e.g. amyotrophic lateral sclerosis (ALS), brain- or spinal cord injury, cerebral palsy, muscular dystrophies, multiple sclerosis, and numerous other degenerative diseases. This condition is called locked-in syndrome - the affected may lose all voluntary muscle controls, including eye movements and respiration. BCI research has opened up an exhilarating option for such disabled people to communicate with the outside world through their brain signals rather than the usual means. An often used method of extracting data is via electroencephalography (EEG), as it is cheap, simple, non-invasive, and with little or no risk and discomfort to the user.

Steady-state visual evoked potentials (SSVEP)-based BCIs function on the premise, that a visual stimuli (e.g. a flash of light) evokes a measurable response in the brain, primarily in the visual cortex [2]. This can be used to determine which of a series of flickering targets is being visually processed by the user. If each target is flickering at a unique frequency, this frequency will be replicated in the EEG when the user is gazing at it. By assigning different values (numbers, letters, command, etc.) to the targets, the user can transfer information to the system by shifting the gaze from target to target.

[1]Technical University of Denmark, Department of Electrical Engineering, Ørsteds Plads, 2800, Kgs. Lyngby, Denmark.
[2]Rigshospitalet, Department of Clinical Neurophysiology, Blegdamsvej 9, 2100 Copenhagen Ø, Denmark.
[3]University of Copenhagen, Department of Odontology, Nørre Allé 20, 2200 Copenhagen N, Denmark.

General challenges with BCI-systems are the lack of speed, robustness, and accuracy. A popular method to counter these is by combining the systems with classifiers [3], many times requiring slow and cumbersome training for each user; often even between or during sessions. Recently semi-supervised (or self-training) techniques have received a lot of attention from the machine learning community. They make use of both labelled- and unlabelled data for classifier design, and tries to adjust the classifier parameters to track variations in the data. Provided model assumptions are valid, they are proven to improve the performance in many classification problems, including BCI [4][5]. Methods such as co-training/self-training use agreement of multiple classifiers/confident predictions of the classifier in addition to the labelled data to adapt the classifier [4]. In our paper, a new approach at SSVEP-based BCI-systems, where a classifier is combined with a self-training algorithm known as tri-training is proposed and tested successfully.

II. METHODS

A. Experimental Setup

The overall system is illustrated in Fig. 1. A 3x3 pattern of targets is presented on a 100 Hz CRT monitor. Each has a checker-board-pattern and shifts between two states (checker-board and inverted checker-board) with a unique frequency between 4 and 17 Hz. The system uses 3 scalp electrodes (Fpz, Fz and Oz) connected to the subject with impedances $< 5\text{k}\Omega$. The sampling rate of the bioamplifier was set to 256 Hz. A computer processes the data from the subject, and send the results to another computer generating

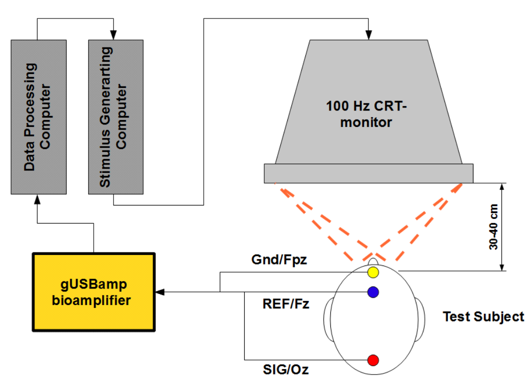

Fig. 1. Schematic representation of the proposed system

the visual output of the system. The subjects were seated within a distance of 30-40 cm of the stimulus screen.

B. Feature Extraction

The feature extraction method is presented in this subsection. An autocorrelation-based feature is chosen because of its simplicity. From theory, the normalized autocorrelation for sampled signal, $x(n)$, of infinite length, is defined as:

$$r_{xx}(l) = \frac{1}{\sum_{n=-\infty}^{\infty} x(n)^2} \sum_{n=-\infty}^{\infty} x(n)x(n+l), \;\; l \in \mathbb{Z}. \tag{1}$$

It can be seen that $r_{xx}(l)$ is symmetric around $l = 0$, and $r_{xx}(0) = 1$ for all real signals, which the recorded signals are assumed to be. Furthermore, in our case, all signals used are of finite length of $M = 256$ samples, and non-overlapping. As the feature-vector (from here on denoted by χ) should only consist of unique values, the following simplification is suggested:

$$\chi_x(l) = \frac{1}{\sum_{n=1}^{M} x(n)^2} \sum_{n=1}^{M-l} x(n)x(n+l), \;\; l = 1,2,...,M-1. \tag{2}$$

The relationship between the *true* SSVEP-signal ($x(n)$) and the recorded signal ($y(n)$) can be modelled as, $y(n) = x(n) + w(n)$ where $w(n)$ is the random measurement noise, which is assumed to be white. In this case, the following approximation can be made to the features of $y(n)$:

$$\chi_y(l) \approx \chi_x(l) \frac{10^{SNR/10}}{10^{SNR/10}+1}, \;\; l \neq 0, \tag{3}$$

where SNR is the signal-to-noise-ratio of $y(n)$. It can be seen that if no clean feature is present, i.e., $\chi_x(l) = 0$, then $\chi_y(l)$ should also be zero.

C. Classification

In this work, we have chosen the Naïve-Bayes classifier (NBC) because of its simplicity and robustness [6]. Furthermore, this classifier had already showed promising results in collaboration with the tri-training algorithm [7]. The NBC assumes normal distribution of the individual features within each class and independence between the features.

Given a labelled feature set, L^j of length N representing the training data for class C_j, it can be represented in a matrix form as in Eq. (4):

$$L^j = [X_y^{1,j}, X_y^{2,j}, \ldots, X_y^{N,j}] = \begin{bmatrix} \chi_y^{1,j}(1) & \chi_y^{2,j}(1) & \cdots & \chi_y^{N,j}(1) \\ \chi_y^{1,j}(2) & \ddots & \ddots & \chi_y^{N,j}(2) \\ \vdots & \ddots & \ddots & \vdots \\ \chi_y^{1,j}(M-1) & \chi_y^{2,j}(M-1) & \cdots & \chi_y^{N,j}(M-1) \end{bmatrix}, \tag{4}$$

where $X_y^{n,j}$ is a column vector representing the nth feature set of class C_j. This way the mean of each feature in the class C_j can be estimated by:

$$\mu_{lj} = \frac{1}{N} \sum_{n=1}^{N} \chi_y^{n,j}(l), \tag{5}$$

where μ_{lj} is the mean of the l-th feature in the j-th class. The variances is given by:

$$\sigma_{lj}^2 = \frac{1}{N-1} \sum_{n=1}^{N} (\chi_y^{n,j}(l) - \mu_{lj})^2, \tag{6}$$

thereby the probability of a given feature being in a specific class can be estimated by the normal probability density:

$$p(C_j|\chi_y(l)) = \frac{1}{\sqrt{2\pi}\sigma_{lj}} e^{-(\chi_y(l)-\mu_{lj})^2/2\sigma_{lj}^2}. \tag{7}$$

Finally the probability of a given set of features being from a specific class, C_j can be estimated by:

$$p(C_j|X_y) = P(C_j) \prod_{l=1}^{M-1} p(C_j|\chi_y(l)), \tag{8}$$

where $P(C_j)$ is the previous occurrence of the class C_j. The probability can be normalized with the sum of probabilities for all classes, but since it is only a matter of determining the most probable class this is not necessary.

D. Thresholding

To determine when the user intends to select a target, a threshold must be implemented. This should also help to ensure that the self-training algorithm is not corrupted by large amounts of noise. Since the feature vector is normalized with respect to the signal power, the values only depends on the quality of the signal, assuming a relatively constant SNR. Therefore the value,

$$\xi = \sum_{l=2}^{M-1} |\chi_y(l)| \tag{9}$$

can be interpreted as a measure of the signal quality. Different frequencies are expected to deliver varying signal quality [8], so the threshold requires either an individual value for each class. As this threshold is obviously directly related to the variance of the autocorrelation, certain factors must be taken into consideration:

- The shorter the signal, the greater the variance of the autocorrelation even if there is no signal. This allows shorter span in which to choose the threshold value.
- The autocorrelation is normalized to signal power. A signal with a sufficiently low SNR might deliver a "clean" signal after the autocorrelation, but it will be scaled down and not able to pass the threshold value, see Eq. (3). This means that signals that can potentially be classified are discarded.

E. Tri-training

Tri-training [7] is a new method for self-training of classifiers from unlabelled data when only little labelled data are available. Labelled data in this context are the signals recorded when the subject is gazing at a known target. To train the classifier accurately, labelled data for all targets are needed. The more data that can be collected, the better the classifier can be trained (as a rule-of-thumb). Unlabelled data

TABLE I

PSEUDOCODE REPRESENTING THE TRI-TRAINING ALGORITHM

```
function h_x = tri_train(L,U)
// h_x is the best classifier,
// L is the labeled dataset, U is the unlabeled dataset
  for i in {1,2,3}
    // random sample representing 75% of the labeled data:
    S_i = bootstrap(L, 0.5 × size_of(L))
    h_i = train_classifier_on(S_i) // train the classifier on the sample
    e'_i = 0.5 //starting error rate
    l'_i = 0 //size of last training set for the classifier
  while h_1 ≠ h_prev_1 or h_2 ≠ h_prev_2 or h_3 ≠ h_prev_3
      // as long as at least one of the classifiers has been updated:
    for i in {1,2,3}
      L_i = ∅
      UpdateClassifier_i = false
      e_i = measure_error_rate([h_j,h_k],L), [j,k] ≠ i
      if e_i < e'_i
      // if the new error rate is lower than the previous:
        for x in U
          if h_j(x) == h_k(x), [j,k] ≠ i
             // if the other classifiers agree on x:
            L_i = [x, L_i] // update the training set with x
        if l'_i == 0
         // if this classifier hasn't been trained before:
          l'_i = e_i/(e'_i − e_i) + 1
        if l'_i < size_of(L_i)
          if e_i × size_of(L_i) < e'_i l'_i // if eq. 10 is satisfied
            UpdateClassifier_i = true
          elseif l'_i > e_i/(e'_i − e_i)
           //if it is possible to satisfy eq. 10
            L_i = bootstrap(L_i, (e'_i l'_i / e_i − 1))
            UpateClassifier = true
    for i in {1,2,3}
      h_prev_i = h_i
      if UpdateClassifier_i == true
        h_i = train_classifier([L, L_i])
        e'_i = e_i
        l'_i = size_of(L_i)
   // estimate the best classifier
  x = min(measure_error_rate([h_1,h_2,h_3],L))
  // return the best classifier
  output(h_x)
```

are data that, with a fair certainty, are recorded with the subject gazing at one of the targets, but without knowing which, and are generated as the system is used. Tri-training attempts to label the unlabelled data democratically and use it for training. It has proven to be very effective [7] compared to other self-training algorithms, and enables the system to gradually improve while being used, and/or adapt to the current user. The system is best described by the pseudo-code shown in Table I.

L is the superset of all *labelled* data from all classes, so that $L^j \subsetneq L$. The set U is a set containing all *unlabelled* data ($U \cap L = \emptyset$) and $L_i \subset U$ is a set containing only self-labelled data. The function $size\_of(L_i)$ outputs the number of labelled feature sets in the set L_i, $bootstrap(L_i, n)$ outputs n random samples from L_i and $measure\_error\_rate([h_j, h_k], L)$ estimates the combined error rate of the jth and kth classifier, h_j and h_k, from the labelled dataset L, see Eq. (11). The key criterion in the tri-training algorithm is that the *absolute* number of falsely labelled feature sets from the unlabelled set must decrease for each training round [7]. This criterion can be summarized in the following equation:

$$0 \leq \frac{e_i}{e'_i} < \frac{l'_i}{size\_of(L_i)} < 1, \tag{10}$$

where e'_i, and l'_i are the error rate and training set size from the previous round respectively, and e_i, and L_i are the current error-rate and current training set. Eq. (10) also contains the criterion that the relative error rate must decrease, and that the number of labelled feature sets must increase. Because the error-rate cannot be determined, it is estimated from the labelled data. In this work, the error rate was estimated by:

$$e_i = 1 - \frac{|h_i(L)_{correct}|}{|L|}, \tag{11}$$

where $h_i(L)_{correct}$ is the correct classifications by h_i of the labelled set L.

III. RESULTS

The experiment was conducted on nine healthy subjects aged between 18 and 25; 4 females and 5 males. All except one were without prior experience with BCI-systems. The recorded data were bandpass-filtered from 5-60 Hz with a 6th order Butterworth filter. The system was trained 10 seconds on each target. After this, the system was tested by sampling 40 random targets. Each target was then highlighted until a correct hit was registered, and the next target from the set would be highlighted. All signals from the training session that surpassed the threshold value (both correct and incorrect hits) were saved as unlabelled training data. The classifiers were then tri-trained and, after a 30 s pause, the test procedure was repeated to measure the improvement. The accuracy ($P = \frac{|\text{correct hits}|}{|\text{correct hits}| + |\text{false hits}|}$) of the system and the information-transfer-rate (ITR) of the system were estimated for performance evaluation. The ITR was calculated using the following formula:

$$ITR = \frac{60}{\bar{t}} \left(log_2 K + P \cdot log_2 P + (1-P) \cdot log_2 \frac{1-P}{1-K} \right), \tag{12}$$

were K is the number of targets (9) and $\bar{t}$ is the average time the subject needed to correctly select a character. It

TABLE II

ITR RESULTS (BT: BEFORE TRI-TRAINING, AT: AFTER TRI-TRAINING)

Subject #	BT (bits/min)	AT (bits/min)
1	20.2 ± 10.8	38.9 ± 12.1
2	3.1 ± 3.7	3.4 ± 0.8
3	2.8 ± 2.6	4.6 ± 5.0
4	16.4 ± 0.1	16.8 ± 3.6
5	94.2 ± 3.9	106.9 ± 11.9
6	73.7 ± 1.4	78.3 ± 11.8
7	10.4 ± 5.8	9.8 ± 3.6
8	66.1 ± 8.5	45.5 ± 0.8
9	55.5 ± 6.0	43.9 ± 22.4

TABLE III

ACCURACY RESULTS

Subject #	BT (%)	AT (%)
1	58.1 ± 3.6	68.4 ± 0.8
2	29.2 ± 10.6	32.9 ± 3.1
3	39.2 ± 21.0	43.9 ± 26.3
4	55.9 ± 0.6	54.9 ± 6.9
5	94.1 ± 1.6	97.6 ± 3.4
6	86.3 ± 6.6	81.2 ± 13.8
7	50.0 ± 6.4	51.8 ± 9.9
8	94.2 ± 4.7	86.1 ± 3.9
9	83.4 ± 3.3	73.2 ± 0.0

TABLE IV

T-TEST RESULTS

Test	Null Hypothesis	p-value	Status
ITR	before $\geq$ after	0.479	not rejected
Accuracy	before $\geq$ after	0.683	not rejected
Selection Time	before $\leq$ after	0.203	not rejected

can be seen, in Table II, that the system is in many ways comparable to existing SSVEP systems in terms of ITR, and that 6 of the 9 subjects improves as a result of tri-training. 5 of the 9 subjects improves in terms of accuracy, as can be seen in Table III. The ITR vs. accuracy is however a weighting, and as the current system is build-up, it could be weighted differently. It could be argued that the subjects tend to fall in two groups performing either far above or far below average, agreeing to the findings of other researchers [9]. The t-test (Table IV) does not exploit any statistically significant changes before and after the tri-training. It should be noted, however, that subjects 7,8, and 9 were tested before a pause between the two testing phases was implemented, with their results indicating that fatigue has a negative influence on the performance. There can be several other factors responsible for the lack of improvement, e.g.:

- The unlabelled training set is relatively small ($\approx$45 epochs) compared to the epochs available under normal operating conditions. It is possible that the small amount of data gives broadly distributed results, whereof some can be worse than the initial accuracy.
- As the classifier assumes normal distribution, and as no signals below the threshold value are used in the training, it could be theorized that the chosen thresholding method distorts the classifier.
- The chosen feature has an almost binary quality; good or bad - this could lead to the classifiers saturating very fast.

Before the experiments were performed, Monte-Carlo simulations were carried out on artificial data. These mirrored most of the results shown - including that the classifiers seemed to saturate very fast, which could suggest that tri-training is not too effective with the used feature and/or data.

IV. Conclusion

The system presents a new approach to the SSVEP-based BCI that is computationally simple, requires little setup time and hardware, and delivers results in-line with existing systems [10]. In real-life applications, the tri-training algorithm could have access to a much greater amount of both labelled and unlabelled data. It is probable that the labelled data could be re-used from session to session, allowing the tri-training to compensate for minor differences as background noise, etc. Initial results indicate that tri-training could improve performance in some cases, but further optimization is most likely possible. The classifier output could in many aspects serve as an improved threshold value, and a classifier accounting for covariance between features could also likely improve performance.

V. Acknowledgment

The authors would like to thank all their test subjects for their support. We also appreciate master students Nelly Richard and Sabrina Lyngbye Wendt for their help with many practical issues.

References

[1] Jonathan R. Wolpaw et. al., “EEG-Based Communication: Improved,” *IEEE Transactions on Rehabilitation Engineering*, vol. 6, no. 3, pp. 326-333, 1998.

[2] Danhua Zhu et. al., “A Survey of Stimulation Methods Used in SSVEP-Based BCIs,” *Computational Intelligence and Neuroscience*, vol. 2010, Article ID 7023357, 12 pages, 2010

[3] Chi-Hsun Wu et. al., “Frequency recognition in an SSVEP-based brain computer interface using empirical mode decomposition and refined generalized zero-crossing,” *Journal of Neuroscience Methods*, vol. 196, pp. 170-181, 2011.

[4] Rajesh C. Panicker, Sadasivan Puthusserypady, and Ying Sun, “Adaptation in P300 Brain-Computer Interfaces: A Two-Classifier Cotraining Approach,” *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 12, pp. 2927-2935, 2010.

[5] 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 2004 25^{th}-29^{th} of July 2004, vol 4, pp. 2877-2882, 2004.

[6] Harry Zhang and Jiang Su, “Naive Bayes for optimal ranking,” *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 20, no. 2, pp. 79-93, 2008.

[7] Zhi-Hua Zhou, and Ming Li, “Tri-Training: Exploiting Unlabeled Data Using Three Classifiers,” *IEEE Transactions on Biomedical Engineering*, vol. 17, no. 11, 2005.

[8] Jian Ding, George Sperling, and Ramesh Srinivasan, “Attentional modulation of SSVEP power depends on the network tagged by the flicker frequency,” *Cereb Cortex. 2006 July*, vol. 16(7), pp. 1016-1029.

[9] Christoph Guger et. al. , “How Many People Could Use an SSVEP BCI?,” *Frontiers in neuroscience*, vol. 6, 2012.

[10] Guangyu Bin et. al., “VEP-Based Brain-Computer Interfaces: Time, Frequency, and Code Modulations,” *IEEE Computational Intelligence Magazine*, vol. 4, no. 4, pp. 22-26, 2009.