J. Neural Eng. **7** (2010) 026003 (10pp) doi:10.1088/1741-2560/7/2/026003

A generative model approach for decoding in the visual event-related potential-based brain–computer interface speller

S M M Martens and J M Leiva

Empirical Inference Department, Max Planck Institute for Biological Cybernetics, Tübingen, Germany

E-mail: smm.martens@gmail.com

Received 12 November 2009
Accepted for publication 25 January 2010
Published 18 February 2010
Online at stacks.iop.org/JNE/7/026003

Abstract

There is a strong tendency towards discriminative approaches in brain–computer interface (BCI) research. We argue that generative model-based approaches are worth pursuing and propose a simple generative model for the visual ERP-based BCI speller which incorporates prior knowledge about the brain signals. We show that the proposed generative method needs less training data to reach a given letter prediction performance than the state of the art discriminative approaches.

(Some figures in this article are in colour only in the electronic version)

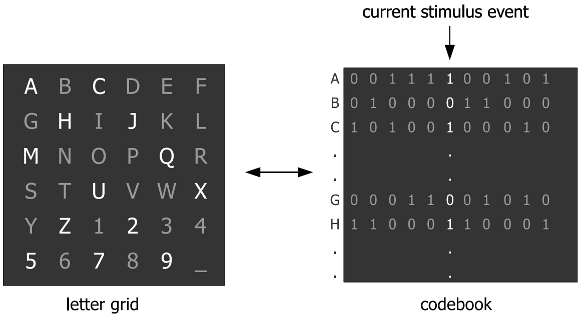
1. Introduction

In brain–computer interface (BCI) research, we aim at inferring some unknown mental state of the subject from an observation of his/her brain activity. We try to decode the brain activity as if the brain were a communication channel which encodes an input message as brain activity. The decoding may be performed in a probabilistic setting to express how certain we are about what we inferred. In that case, we need to learn distributions or conditional probabilities of the unknown and observed variables and do statistical inference. There are two main approaches in the field of statistical inference. They will be discussed in a BCI setting in which we denote the mental state by y and the brain activity by x.

In a *generative modelling* approach we learn the distribution of the brain signals given the mental state $p(\mathrm{x}|\mathrm{y})$ for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, or the joint probability of the brain signals and the mental state $p(x, y)$. That means that we obtain a model of how to generate the brain signals for all possible mental states $\mathcal{Y}$. Then, using Bayes' theorem, we can turn $p(x|y)$ or $p(x, y)$ into $p(y|x)$ and perform a maximum a posteriori (MAP) decoding to infer the most probable state given the observed brain signals. Any prior knowledge about the brain signals may be incorporated into the generative model. Moreover, the decoding performance gives an indication of how realistically we modelled the part of the brain signal involved in encoding the mental state.

The alternative to generative modelling is the *discriminative approach* where we estimate the conditional probability $p(y|x)$ directly without caring about modelling the brain signals. This approach may be easier if the distribution $p(x|y)$ is complex [1].

In this paper, we focus on doing statistical inference in one type of BCI system called the *visual ERP-based speller* [2]. This system enables users to spell words by focusing their attention on letters in a letter grid displayed on a computer screen. While a sequence of controlled stimulus events over time takes place on the letters, the electroencephalogram (EEG) of the user is recorded. If we represent the stimulus events for a given letter in the letter grid as a bitstring [3], we may think of this bitstring as a codeword in a noisy communication channel. The codeword entries with value 1 correspond to stimulus events in which the letter participated; all other entries have value 0. In this way, we may represent all the letters in the letter grid by codewords. The collection of all codewords is the codebook. Each column of the codebook represents a stimulus event over time and the 1-entries in the column indicate which letters take part in that stimulus event. For example, have a look at the sixth column of the codebook in figure 1 (the sixth column is indicated by an arrow above the codebook). This column shows that this stimulus event takes



A B C D E F
G H I J K L
M N O P Q R
S T U V W X
Y Z 1 2 3 4
5 6 7 8 9 _

A 0 0 1 1 1 1 0 0 1 0 1
B 0 1 0 0 0 0 1 1 0 0 0
C 1 0 1 0 0 1 0 0 0 1 0
. .
. .
G 0 0 0 1 1 0 0 1 0 1 0
H 1 1 0 0 0 1 1 0 0 0 1
. .
. .

Figure 1. A sequence of stimulus events over time on a letter in the letter grid is represented by a codeword which is a row in the codebook.

place on $\{A, C, H\}$ (and on $\{J, M, Q, U, X, Z, 2, 5, 7, 9\}$ as can be seen in the letter grid; however, their codewords are not depicted in the figure).

The user should transmit the information about the codeword of the letter he/she wants to communicate, the target codeword. An entry with value 1 in this target codeword is referred to as a target event, a 0 as a non-target event. By producing different brain responses for the target and non-target events on a letter, the user implicitly conveys the information of its codeword. A common strategy for the subject is to count the target events and to ignore the non-target events. The resulting epochs following a target event will contain attention-modulated components such as the P300 event-related potential (ERP) at larger amplitudes than the epochs following a non-target event.

Now let us look at the statistical inference problem in the speller system. Our task is to infer the letter of interest by translating the information about the codeword from the user's EEG. We denote the codeword by c and the observed multi-channel brain signals by b. A MAP decoding consists of finding the most probable codeword from the set of possible codewords given the brain signals

$$\hat{c} = \underset{c \in \mathcal{C}}{\operatorname{argmax}}\, p(c|b), \tag{1}$$

with $\mathcal{C}$ the codebook. Since $p(c|b) = p(c, b)/p(b)$ and $p(b)$ is independent of the codewords, (1) is equivalent to

$$\hat{c} = \underset{c \in \mathcal{C}}{\operatorname{argmax}}\, p(c, b). \tag{2}$$

In practice, learning either the discriminative model $p(c|b)$ in (1) or the generative model $p(c, b)$ in (2) is a hard task. The variable b is continuously valued and has high dimension, and c has as many dimensions as the length of the codebook. Therefore, to learn $p(b|c)$ or $p(c|b)$ we would need an infeasibly large training set.

Fortunately, independence assumptions in the brain signals generation process can simplify the learning of the joint $p(c, b)$ in (2) significantly. For example, let us assume that the brain signals collected at a particular stimulus event b_j are only dependent on codeword entry c_j and not on other stimulus events nor on brain signals at other stimulus events. In that case, the joint $p(c, b)$ can be factorized into products of conditional probabilities $p(b_j|c_j)$ and marginal probabilities $p(c_j)$:

$$p(c, b) = \prod_j p(b_j|c_j)p(c_j). \tag{3}$$

Now the dimensionality of the learning problem has decreased because both b_j and c_j have fewer dimensions than b and c, respectively. Interestingly, we can again choose between a generative approach that learns $p(b_j|c_j)$ and a discriminative approach that learns $p(c_j|b_j)$. The latter approach turns the problem into a per-bit classification problem. A MAP decoding is then obtained by expressing the learned $p(c_j|b_j)$ as $p(b_j|c_j)$ with Bayes' theorem, estimating the joint $p(c, b)$ by (3) and doing the decoding as in (2).

Some generative approaches were proposed in the original visual speller paper by Farwell and Donchin [2], albeit without a probabilistic framework. They involved area, peak-picking and covariance measures in one EEG channel. The area and peak-picking method assume that the response to certain stimulus events is reflected by an increased EEG amplitude. The covariance method estimates a template for the brain response to stimulus events and uses the covariance of the template and the observed brain signals as a similarity measure. A similar method was proposed by Sutter for a different type of speller system based on visual evoked potentials (VEP) [4]. His method estimated the brain response to a sequence of stimulus events. Also this method considered only one EEG channel, whereas other EEG channels also contain relevant information.

Lately, good results have been achieved with discriminative approaches which treat the decoding as a per-bit classification problem [5–9]. Remarkably, the frequently used classifiers such as the support vector machine (SVM) and stepwise discriminative analysis (SWDA) are not designed to give proper probabilistic measures. Therefore, it is not evident how to combine the classifier outputs for the different stimulus events over time. Nevertheless, taking the inner product of the codewords and the classifier outputs and selecting the codeword with the largest outcome seem to give satisfying results. Comparisons between the performance of discriminative versus generative approaches are scarce but seem to indicate that discriminative approaches outperform the generative approaches [10].

Despite the current popularity of discriminative approaches, we claim that generative approaches are worth pursuing for the visual speller and more generally for BCI research. Firstly, there is evidence that generative methods need less training data to approach their asymptotic error than discriminative methods [11, 12]. One explanation is that in the generative model we restrict the space of possible models and avoid overfitting by incorporating prior knowledge about the data, whereas in a discriminative approach this regularization of the model space is generally obtained by a cumbersome cross-validation. In BCI a fast convergence of the classification method is desirable since the acquisition of training data is time consuming and boring. If the BCI is to be used in patients, then a fast classification learning curve is even more stringent due to the reduced attention

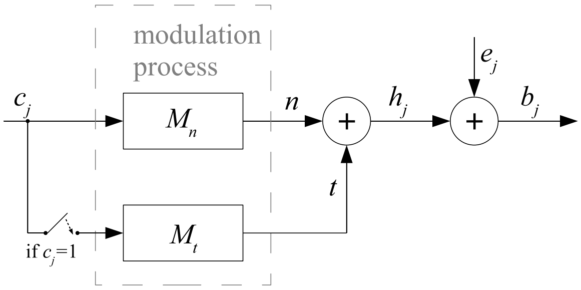


Figure 2. Blockscheme of modulation process with codeword entry c_j as input of unknown systems M_n and M_t and hidden brain response h_j as output. A noisy version b_j of h_j is measured.

span [13, 14]. Secondly, since the decoding performance indicates how well the generative model has represented the reality, a comparison of different generative models results in a better understanding of how the brain signals are generated. Possibly, this information helps the BCI researcher to improve his BCI paradigm and get brain signals with higher signal-to-noise ratios.

The contribution of this paper is threefold. We present a generative model that generates brain signals as a function of a given stimulus sequence in the visual speller system. We derive a simple maximum a posteriori (MAP) solution for predicting the target letter given a multi-channel EEG recording and the proposed generative model. Finally, we compare its decoding performance with state-of-the-art discriminative approaches.

2. Methods

2.1. Generative model of brain signals

Inspired by the psychophysiology literature which postulates that ERPs have strong deterministic characteristics [15–17], we model the response of the brain to a stimulus event as a deterministic system. The system consists of operators M_n and M_t which map a codeword entry c_j to a multi-dimensional brain response h_j (figure 2). Each c_j can take a value from the set $\{0, 1\}$, whereas h_j is a continuous-valued variable with dimensions $[N_s \times N_{ch}]$ with N_s being the number of time samples and N_{ch} the number of channels.

We assume that every stimulus event c_j evokes a sensory response n with a duration of SOA seconds, independent of the value of c_j. SOA stands for stimulus onset asynchrony and is defined as the time interval between the start of one stimulus event and the start of the next event. The sensory response n has a dimension $[N_n \times N_{ch}]$ with $N_n = \lfloor \mathrm{SOA} \cdot f_s \rceil$ samples, $\lfloor \cdot \rceil$ being the nearest integer function. In addition, if the stimulus event is a target event, i.e. $c_j = 1$, a target-like modulation t is evoked which lasts 0.6 s. This t has a dimension $[N_t \times N_{ch}]$ with $N_t = \lfloor 0.6 \cdot f_s \rceil$ samples. We write this as

$$h_j = \begin{cases} n & \text{if } c_j = 0, \\ t + n & \text{if } c_j = 1, \end{cases} \tag{4}$$

where we can make the dimensions correct by filling up n with zeros.

We cannot observe h_j directly. Instead, we may define an epoch b_j as an EEG segment in which we expect to capture the hidden brain response h_j and some non-task-related noise e_j. Here, the noise represents the background EEG, artefacts and measurement noise. From now on we will refer to h_j as the hidden brain response and to b_j as the observed brain response.

If the modulation process behaves as a linear system in the sense that the principle of superposition holds, the hidden brain response h to a sequence of stimulus events c can be constructed by simply adding up the responses h_j for the separate stimulus events c_j at the time points where stimulus event j takes place. This operation may be expressed as a convolution or as a matrix multiplication using Toeplitz matrices. In the latter case we write

$$h = \mathbf{S_t}^T t + \mathbf{S_n}^T n. \tag{5}$$

The matrices $\mathbf{S_t}$ and $\mathbf{S_n}$ are constructed as follows (see also figure 3). Given a length N codeword c with entry 1 specifying a target event and entry 0 specifying a non-target event, we construct a $[1 \times N_s]$-padded codeword with sampling frequency f_s by adding zeros in between the codeword entries. An $[N_t \times N_s]$ Toeplitz matrix $\mathbf{S_t}$ is built with the padded codeword on the first row. Empty entries are set to 0. In a similar way, we construct an $[N_n \times N_s]$ Toeplitz matrix $\mathbf{S_n}$ from a $[1 \times N]$ vector of ones. Note that the overlap of long-latency brain signals at short SOA [18] is explicitly modelled in (5).

Given an ERP plus additive noise, it follows from the central limit theorem [19] that an averaging procedure improves the signal-to-noise ratio proportionally to the number of epochs used for averaging. Therefore, if e is uncorrelated with h, then the average of the observed EEG epochs b_j for $c_j = 1$ and for $c_j = 0$ in a training set is estimates for t+n and n, respectively. Alternatively, we may estimate t and n by the least-squares (LS) solution which minimizes $\varepsilon = E[b - h]^2$ in a training set with h given by (5). The LS solution for t and n is obtained by solving the following system of linear equations:

$$\begin{pmatrix} \mathbf{S_t S_t}^T & \mathbf{S_t S_n}^T \\ \mathbf{S_n S_t}^T & \mathbf{S_n S_n}^T \end{pmatrix} \begin{pmatrix} t \\ n \end{pmatrix} = \begin{pmatrix} \mathbf{S_t} b \\ \mathbf{S_n} b \end{pmatrix}. \tag{6}$$

A misalignment between the constructed hidden and observed brain signals may have a significant impact on the decoding. It is important to have the starting time of each stimulus event at a large precision and to do the construction of the hidden brain signals at a sufficiently large sampling frequency. We found that a sampling frequency of 250 Hz resulted in sufficiently small rounding errors.

2.2. MAP estimation

The MAP estimation decides what is the most likely codeword $\hat{c}$ given the observations b. A unique hidden brain response exists for each of the different codewords in the codebook. In that case, we can equivalently look for the most likely hidden

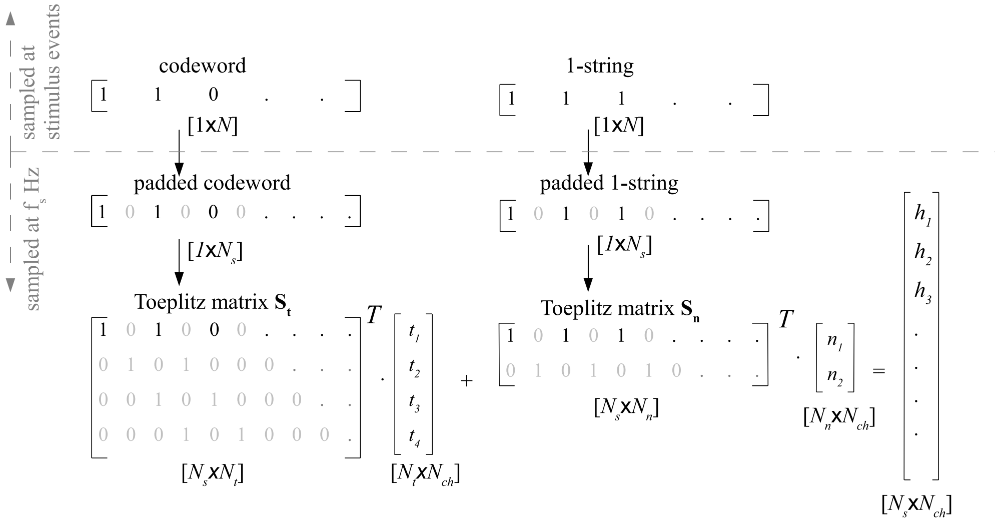


Figure 3. Construction of hidden brain signal.

brain response $\hat{h}$ given the observed brain signals b:

$$\hat{h} = \underset{h}{\operatorname{argmax}}\, p(h|b). \tag{7}$$

Using Bayes' theory we state $p(h|b) = p(b|h)p(h)/p(b)$. For the MAP estimation we may neglect $p(b)$, since the term is independent of h:

$$\hat{h} = \underset{h}{\operatorname{argmax}}\, [p(b|h)p(h)]. \tag{8}$$

We assumed that the observed brain signals are noisy versions of the hidden brain responses according to $b = h + e$. In that case, the distribution $p(b|h)$ is just the distribution of the noise $p(e)$. Let us consider h as a vector in a signal vector space of dimension L. Under the reasonable assumption that the noise is white in the frequency band of t and n and Gaussian with zero mean, the noise in each channel has a spherical distribution across the different components of that vector space (see appendix A for details). Then, for channel k the distribution of the noise $p(e_k) = p(b_k|h_k)$ is

$$p(b_k|h_k) = \frac{1}{(2\pi\sigma_k^2)^{L/2}} \exp\left(-\frac{\|b_k - h_k\|^2}{2\sigma_k^2}\right). \tag{9}$$

Let us assume that we decorrelate the noise across channels by a linear transformation $\mathbf{W}$. The resulting decorrelated noise e^W is in fact independent across channels with variance σ_W^2 (see appendix B), and we may write the distribution $p(e^W) = p(b^W|h^W)$ as a product:

$$p(b^W|h^W) = \prod_k \frac{1}{\left(2\pi\sigma_W^2\right)^{L/2}} \exp\left(\frac{-\left\|b_k^W - h_k^W\right\|^2}{2\sigma_W^2}\right). \tag{10}$$

It turns out that the MAP criterion of (8) is equivalent to the following MAP criterion (see appendix B):

$$\hat{h} = \underset{h}{\operatorname{argmax}}\, p(b^W|h^W)p(h). \tag{11}$$

Because the logarithm is monotonic, we can perform the MAP decision on the log-likelihood $\hat{h} = \operatorname{argmax}_h \left[\ln\left[p(b^W|h^W)p(h)\right]\right]$:

$$\hat{h} = \underset{h}{\operatorname{argmax}} \sum_k \left[\ln q - \frac{\left\|b_k^W - h_k^W\right\|^2}{2\sigma_W^2}\right] + \ln p(h), \tag{12}$$

with $q = \left(2\pi\sigma_W^2\right)^{-L/2}$.

We may set σ_W^2 to 1 and neglect q since this term is independent of h^W. The resulting MAP decision is simply

$$\hat{h} = \underset{h}{\operatorname{argmin}} \left[\sum_k \left\|b_k^W - h_k^W\right\|^2 - 2\ln p(h)\right]. \tag{13}$$

If the letters are randomly drawn from the set of letters in the letter grid, the letter prior is flat and the MAP estimation becomes equivalent to a maximum likelihood (ML) estimation. In that case

$$\hat{h} = \underset{h}{\operatorname{argmax}}\, p(b|h) = \underset{h}{\operatorname{argmin}} \sum_k \left\|b_k^W - h_k^W\right\|^2. \tag{14}$$

The decoding rule in (14) says that we select the brain response for which the sum of the squared values of the difference between the whitened observed brain signals and the transformed hidden brain signals over all components k is minimal. This gives a MAP estimation under a number of assumptions about the noise and under a flat letter prior.

2.3. Summary of the method

```
GENERATIVE MODEL
INPUT: Set of training observations $b_{\mathrm{train}}$ and selected codewords
Band-pass filter $b_{\mathrm{train}}$, store as $b$
IF averaging:
  Cut up $b$ into epochs $b_j$
  Average all $b_j$ for which $c_j = 0$, store as $n$
  Average all $b_j$ for which $c_j = 1$, subtract $n$, store as $t$
ELSEIF least-squares:
  Construct matrices $\mathbf{S_t}$ and $\mathbf{S_n}$
  Build target and non-target templates $t$ and $n$ by (6)
END
DECODING
INPUT: Test observation $b_{\mathrm{test}}$
Band-pass filter $b_{\mathrm{test}}$, store as $b$
Whiten $b$ by the whitening matrix $\mathbf{W}$, save as $b^W$
FOR each codeword $c$ in the codebook
  Construct matrices $\mathbf{S_t}$ and $\mathbf{S_n}$
  Build hidden response $h$ using $t$ and $n$ by (5)
  Whiten $h$ by $\mathbf{W}$, save as $h^W$
  FOR each channel $k$
    Compute distance $\|b_k^W - h_k^W\|^2$
  END
END
Apply decoding rule (13)
Predict the letter corresponding to $\hat{h}$
```

3. Evaluation on real data

We used the visual speller data from six subjects (see [3]). Each subject used two different codebooks, the RC and d10 codebook and two different stimulus types, the FLASH and FLIP. The RC codebook represents the standard row–column type of stimulus events. The codewords have length 72 corresponding to 6 stimulus rounds and minimum hamming distance 12. The d10 codebook is a codebook for which per stimulus events more letters participate. It consists of length-72 codewords with a minimum hamming distance of 30. Given an SOA of 167 ms, one trial (in which one letter is communicated) takes about 12 s independent of the codebook used. The FLASH is the standard stimulus type which consists of intensification of letters. The FLIP is an alternative stimulus type which consists of rotation of blocks around the letters. During the experiment, a 58-channel EEG was recorded at 250 Hz. Per stimulus type and per codebook, the subjects spelled either 64 (RC) or 32 (d10) letters in the copy-spelling mode. The letters were randomly drawn from the set of letters in the letter grid.

The signal analysis was performed offline in Matlab (The MathWorks, Inc.). The EEG channels had a common average reference (CAR). The EEG was bandpass filtered between 0.5 and 10 Hz using FIR Bartlett–Hanning filters with order 1000.

We divided the data into several training and test sets. The sizes of these sets are specified further on in the paragraph *Learning speed*. On each training set, we trained the generative models GEN1 and GEN2. We also implemented the winning algorithms for the P300 speller data in BCI Competition II[1] and III[2] by Kaper *et al* [5] and Rakotomamonjy and Guigue [8]. Both are discriminative approaches and apply a soft-margin support vector machine (SVM) [20, 21]. We will refer to these algorithms as DIS1 and DIS2.

GEN1. We obtained a target and non-target template from the training set by averaging epochs. With these templates and the codebooks from the test set, we constructed the brain response h from (5) for each hypothetical target letter. Then, we downsampled the observed and constructed brain signals to 25 Hz and transformed these signals by a whitening filter. The EEG data before the start of the stimuli were too short to reliably estimate the whitening filter (note that we need to estimate the N_{ch}^2 entries in the covariance matrix). Therefore, we chose to estimate the whitening filter on the (non-downsampled) training data and the pre- and post-stimulus data from the test set. Prior to whitening, the last eigenvector with zero eigenvalue due to the CAR operation was removed from the whitening matrix. Finally, since the letter prior was flat, we performed the decoding as in (14).

GEN2. We obtained a target and non-target template from the training set by performing a least-squares estimation as in (6). The remaining procedure was the same as in GEN1.

DIS1. Only channels Fz, Cz, Pz, Oz, C3, C4, P3, P4, PO7 and PO8 were considered in the analysis. Although Kaper *et al* applied a [0.5–30] Hz bandpassfilter, we used [0.5–10] Hz filtered data for a fair comparison between the discriminative and generative methods. The filtered signals were cut up in 0.600 s EEG epochs synchronized by the stimulus cues. These epochs were downsampled to 25 Hz and normalized to the $[-1, 1]$ range. An SVM with a Gaussian radial basis function (RBF) kernel [22] was trained on balanced training data. The balancing was done by throwing away randomly selected excessive non-target epochs. The SVM penalty parameter C and the kernel width that gave the highest binary classification scores on a tenfold cross-validation on each training set was selected. Then, the trained SVM was applied to the data in the test set. The inner products of the codewords and the classifier outputs were calculated. The letter whose codeword showed the largest inner product was selected. Since we expect that balancing by throwing away training data affects the learning speed, we also trained an SVM on the intact training data. Furthermore, we investigated if the performance was affected if no channel selection was done.

DIS2. Although Rakotomamonjy and Guigue applied a [0.1–60] Hz bandpassfilter and an epoch length of 667 ms, we used [0.5–10] Hz filtered data and 0.600 s EEG epochs, again for a fair comparison between the discriminative and generative methods. The epochs were downsampled to 25 Hz and normalized to have zero mean and unit variance. If the number of training letters was larger than or equal to 10, SVMs

[1] P300 speller paradigm, Wadsworth Center, NYS Department of Health. BCI Competition II, http://bbcide/competition/ii/.

[2] P300 speller paradigm, Wadsworth Center, NYS Department of Health. BCI Competition III, http://bbcide/competition/iii/.

with linear kernels were trained on each partition consisting of five training letters. For each partition the penalty parameter C that gave the largest score $C_{cs} = tp/(tp + fp + fn)$, with tp being the number of true positives, fp the number of false positives and fn the number of false negatives on the remaining training partitions, was selected. We applied the channel selection procedure as in [8]. If the number of training letters was smaller than 10, one SVM was trained on the complete training set and C that gave the largest score C_{cs} in a tenfold cross-validation on the training set was selected. Also here, the channel selection procedure was applied. Then, the trained SVMs were applied to the data in the test set. The inner products of the codewords and the (summed up) classifier outputs were calculated. The letter whose codeword showed the largest inner product was selected. We also trained a single SVM without a channel selection procedure to evaluate the effect of applying multiple classifiers and the channel selection procedure.

We calculated the letter prediction performance as the percentage of correctly classified test letters. For each method, we evaluated the learning speed and the steady state performance.

Learning speed. A learning curve shows the decoding performance as a function of the number of training examples. In our data, we constructed learning curves using the first 1, 2, 5, 10, 20 or 30 letters as training data for the RC data and the first 1, 2, 5, 10 or 20 letters for the d10 data.

Steady state performance. The steady state performance is the asymptotic letter prediction performance at an infinitely large training set. We estimated the steady state performance by considering the letter prediction performance using a training set of 30 (RC) or 20 (d10) letters.

4. Results

In the following subsections, we first test which generative and discriminative method performed best on the RC FLASH data. In the last subsection, we compare the selected generative and discriminative methods on the RC and d10 FLASH and FLIP data.

4.1. GEN1 versus GEN2

The learning curves of the generative methods showed an increase in letter prediction performance as we increased the number of letters in the training set from 1 to 30 (see figure 4). Both methods reached a letter prediction performance of (close to) 100% with an training set size of 20 or 30 letters. The average performance was slightly higher for the averaging-based generative model GEN1.

4.2. DIS1 versus DIS2

All the discriminative methods reached a letter prediction performance of (close to) 100% with an training set size of 20 or 30 letters. Both the balancing of training data and the channel selection in DIS1 harmed the learning speed (see figure 5). Most likely, for large enough training set sizes the balancing does not hurt the SVM performance. The channel selection procedure in DIS2 decreased the learning speed.

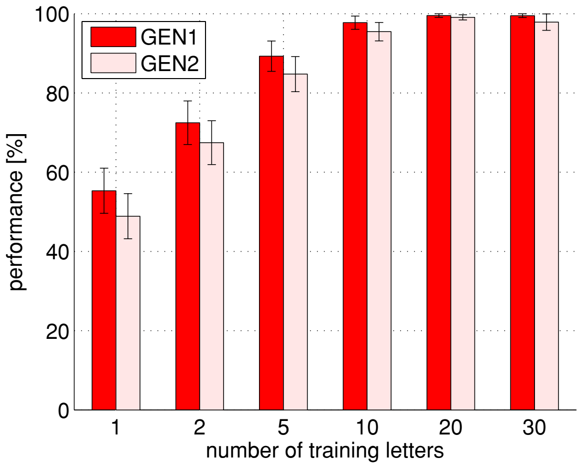


Figure 4. Letter prediction performance as a function of training set size for the generative models GEN1 which uses averaging (dark bars) and GEN2 which uses least-squares (light bars) to obtain the templates. The data are from six subjects who used the RC codebook with the FLASH stimulus type. The bars represent averages over subjects, and the error bars denote the average standard deviation over subjects.

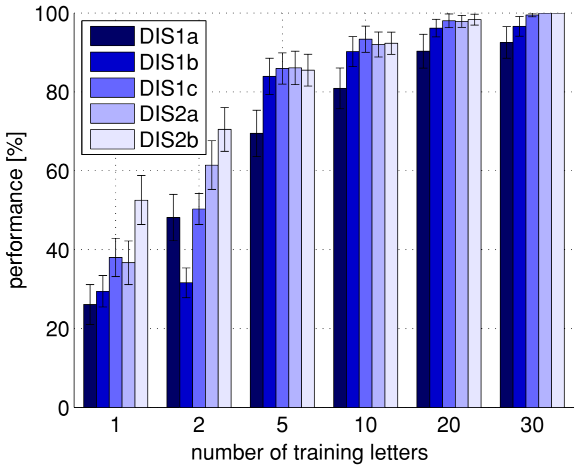


Figure 5. Letter prediction performance as a function of training set size for the SVM-based decoding for DIS1 with balanced training data and channel selection (DIS1a), not balanced training data and channel selection (DIS1b), not balanced training data without channel selection (DIS1c), and for DIS2 which uses a linear instead of a Gaussian RBF kernel with channel selection procedure and multiple classifiers (DIS2a) and without channel selection and single classifier (DIS2b). The data are from six subjects who used the RC codebook with the FLASH stimulus type. The bars represent averages over subjects, and the error bars denote the average standard deviation over subjects.

For large training set sizes the selected RBF width in DIS1 was always large. For small training set sizes, occasionally a small kernel width was selected in the cross-validation. We found that the larger kernel widths resulted in better letter prediction performance. This indicates that a linear kernel

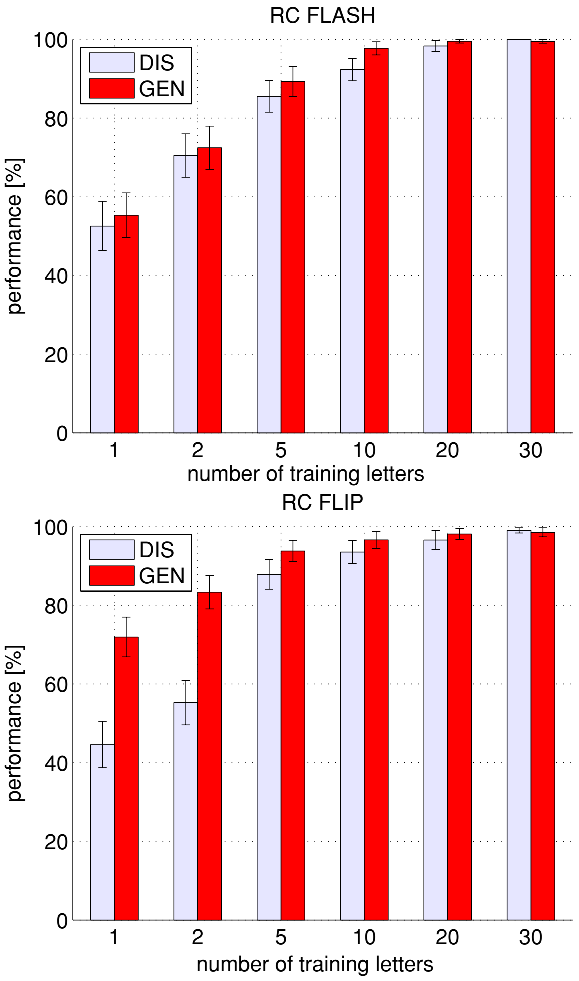


Figure 6. Letter prediction performance as a function of training set size for DIS (light bars) and GEN (dark bars) for different stimulus types in the RC codebook. The bars represent averages over subjects, and the error bars denote the average standard deviation over subjects.

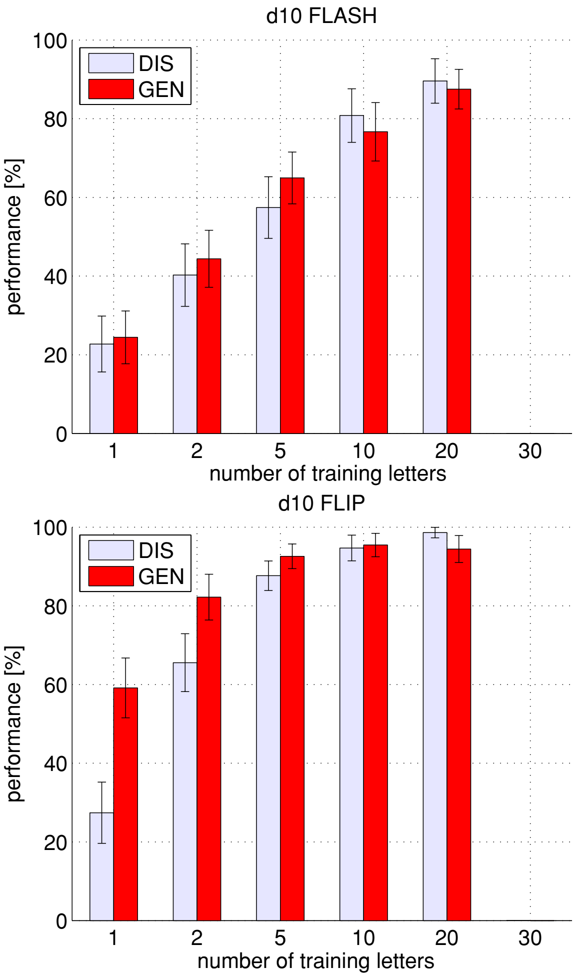


Figure 7. Letter prediction performance as a function of training set size for DIS (light bars) and GEN (dark bars) for different stimulus types in the d10 codebook. The bars represent averages over subjects, and the error bars denote the average standard deviation over subjects.

may perform just as well as a Gaussian kernel on this type of data. Since the Gaussian kernel is more prone to overfitting, a linear kernel should give better results on small training set sizes. In agreement, DIS2 (without a channel selection procedure) performs better than DIS1 on small training set sizes (see figure 5). For training sets consisting of five or more training letters, DIS1 (with imbalanced training data and without channel selection) and DIS2 perform equally well.

4.3. Discriminative versus generative

In this section we compare the performance of the best generative method GEN1 and the best discriminative method DIS2b, which will be from now on referred to as GEN and DIS, respectively. The steady state performance given a large amount of training data was the same for both methods. They reached close to 100% letter prediction performance on the RC FLASH, RC FLIP and d10 FLIP data. On the d10 FLASH data both methods reached a letter prediction accuracy of about 90%. Possibly, at 20 training letters the steady state performance was not yet reached.

Despite the similar steady state performances, the learning speed was clearly different (see figures 6 and 7). On the RC FLASH data the generative method reached 90% letter prediction accuracy with a small training set size consisting of five training letters, whereas the discriminative method needed two times more training data, respectively, to produce this level of accuracy. On the FLIP RC and d10 data, the faster learning speed of the generative method was even more pronounced.

Even though both the generative and the discriminative approach had a similar steady state performance, the generative model gives more insight into where and when relevant information is present in the EEG than the discriminative method (see figure 8). The classifier weights in the discriminative method merely show which combination of channels and time points maximizes the classification score. The weights reflect both the discriminative information and the cancellation of noise signals. In contrast, the target template in the generative method is the additive component in the

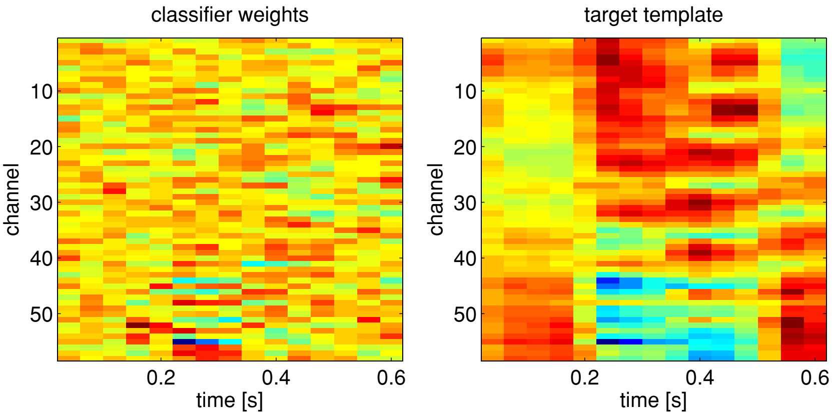


Figure 8. Classifier weights in DIS2 (left plot) versus target template in GEN1 (right plot) for the RC FLASH data from one subject using 30 training letters. The *x*-axis represents time, and the *y*-axis represents the different channels. The location of channels is as follows: 1–10 is frontal, 11–19 is fronto-central, 20–27 is central, 28–36 is centro-parietal, 37–50 is parietal and 51–58 is occipital. Dark blue represents strong negative values and dark red strong positive values.

response to a target event compared to a non-target event. It therefore renders the discriminative information of each channel at each time point. For example, for the subject in figure 8 the target template for the occipital channels shows positive values between 0 and 0.2 s. Furthermore, the fronto-central, central and parietal channels show positive values between 0.2 and 0.5 s where we expect the P300 component. At the same time, the occipital channels show negative values. Finally, the occipital channels show positive values between 0.5 and 0.6 s.

5. Conclusion

In discriminative learning, the aim is to discover the dependence of an unobserved variable on an observed variable. Implicitly, only the posterior probability $p(y|x)$ is of interest. Therefore, this type of learning is agnostic with respect to the nature of the observations and the underlying processes that generate them. In many applications such as image-based digit recognition [23], text classification [24], computer vision [25] and brain–computer interfaces [26] the discriminative approach has been applied with success. Also the standard decoding technique in speller system data is based on discriminative modelling. The problem is reformulated as finding the labels of each epoch using a classifier, for example an SVM.

Generative methods, on the other hand, are based on the joint $p(x, y)$ or the conditional probability $p(x|y)$, from which the posterior $p(y|x)$ can be obtained by means of Bayes' theorem. In some cases, these methods are preferable to provide the learning algorithm with knowledge about the problem at hand. However, the task of finding the probability models may be hard if the model underlying the data set is complex, e.g. non-Gaussian, and if the data are multi-dimensional.

This is the first work that introduces a generative modelling-based decoding method for visual speller data in a probabilistic framework. The generative model constructs templates for the brain response to each stimulus event. Using these templates, a brain signal is generated as a function of the stimulus sequence. Prior knowledge about overlap of the long-latency brain signals is modelled explicitly when constructing the brain response to a sequence of stimulus events. Under the assumption that the noise is additive, Gaussian and white in the frequency band of the task-related brain signals, an easy MAP decision is obtained for predicting the letter. The decoding rule explicitly states how letter frequency information can be incorporated. A channel selection procedure as in earlier generative modelling work for speller systems, which may be computation intensive, is obsolete. This holds under the assumption that the noise is Gaussian and has been decorrelated across channels. If this assumption is violated, a channel selection procedure could improve the results.

We compared the generative method to two state-of-the-art discriminative methods on visual speller data from six subjects which used two different codebooks and two different stimulus types. The performance of the discriminative methods on the RC FLASH data was in good agreement with a recent large-scale study by Guger *et al* [27] who reported an average letter prediction accuracy of 90% using a discriminative method trained on 900 examples (which would correspond to 12 training letters in our datasets). Our results show that the proposed generative method competes with state-of-the-art discriminative approaches for large training set sizes. Moreover, the learning curve of the generative method-based decoding is steeper than of the discriminative methods. The generative method needed only half the amount of training data to achieve 90% letter prediction accuracy. Therefore, the method is very promising for BCI research where a short acquisition of training data is desirable, especially when working with patients.

The generative approach gives us more insight into the modulation process of the brain in response to the stimuli than the discriminative approach. The target template in the generative model conveys which channels and which time points contain discriminative information, whereas the classifier weights in the discriminative method merely reflect which combination of channels and time points results in a good classification score.

The good performance of the generative method indicates that the model of the brain signal generation was close to reality. Nevertheless, we think that the generative model may be improved. For example, the fact that the performance of the averaging-based generative model was better than that of least-squares indicates that the superposition assumption is too strong. That there are nonlinear effects in the speller system was already pointed out by [18] who showed that target responses with a small target-to-target (TTI) interval have reduced amplitude, at least when the standard flash stimulus is used. Modelling this and other nonlinear effects may lead to a more realistic generative model and, correspondingly, an even better performance.

Appendix A. Characterization of the noise

In order to perform the MAP decision, we need to previously assume the following facts about the signal and the noise.

- The signal h_k corresponding to channel k has a low-pass characteristic with cut-off frequency f_0, and admits a representation on an orthonormal basis of unitary signals $\{\phi_l(t)\}$, $l = 1, \ldots, L$. This allows us to indistinctly refer to h_k as a signal or as a vector in an L-dimensional Hilbert vector space.
- The noise e_k has zero mean and follows a Gaussian distribution. We also assume that the noise is white in the relevant low-pass frequency range of the signal, with a constant power spectral density equal to $\eta_k/2$.

We need to statistically characterize the noise in the signal vector space defined above. This can be achieved by applying basic linear systems theory ([28], p 28). Let $\Phi_l(f)$ be each element of the signal basis, expressed in the frequency domain. In that case, the variance of the noise σ_{kl}^2 in each dimension of the signal vector space is given by the projection of the noise on each element of the basis:

$$\sigma_{kl}^2 = \int_{-f_0}^{f_0} \frac{\eta_k}{2} |\Phi_l(f)|^2 \, \mathrm{d}f = \eta_k f_0. \tag{A.1}$$

Equation (A.1) shows that the variance of the noise is identical for each component of the basis. Moreover, a property of linear filters is that a Gaussian input leads to a Gaussian output. Therefore, for the given assumptions about the noise, the distribution in the vector space is described by a spherical Gaussian, i.e. having a covariance matrix given by $\mathbf{C} = \sigma_k^2 \mathbf{I}$. We may write the distribution for each channel as

$$e_k \sim \frac{1}{\left(2\pi\sigma_k^2\right)^{L/2}} \exp\left(-\frac{\|b_k - h_k\|^2}{2\sigma_k^2}\right). \tag{A.2}$$

Appendix B. MAP estimation on decorrelated signals

Let us assume that the noise in different channels can be decorrelated by means of a linear transformation $\mathbf{W}$, in the sense that the noise is now uncorrelated across channels and has the same variance $\sigma_{W,k}^2 = \sigma_W^2$. The transformation matrix $\mathbf{W}$ can be calculated on a part of the recording in which no stimuli have been presented to the subject, or on the complete recording if the noise is much stronger than the signal. Because decorrelation means statistical independence for Gaussian random variables, we can describe the joint distribution of the noise for all channels $e^W \sim p(h^W + e^W | h^W) = p(b^W | h^W)$ as

$$p(b^W|h^W) = \prod_k \frac{1}{\left(2\pi\sigma_{W,k}^2\right)^{L/2}} \exp\left[-\frac{\|b_k^W - h_k^W\|^2}{2\sigma_{W,k}^2}\right] \tag{B.1}$$

$$= \prod_k \frac{1}{\left(2\pi\sigma_W^2\right)^{L/2}} \exp\left[\frac{-\|b_k^W - h_k^W\|^2}{2\sigma_W^2}\right]. \tag{B.2}$$

We make use of the property of lossless (full-rank) linear transformations on probability distributions: $p(\mathbf{A}^T x) = (\det \mathbf{A})^{-1} p(x)$. Then the MAP criterion in (8) can be reformulated as

$$\hat{h} = \underset{h}{\operatorname{argmax}}[p(\mathbf{W}^{-1}b^W|\mathbf{W}^{-1}h^W)p(h)] \tag{B.3}$$

$$= \underset{h}{\operatorname{argmax}}[(\det \mathbf{W})p(b^W|h^W)p(h)] \tag{B.4}$$

$$= \underset{h}{\operatorname{argmax}}[p(b^W|h^W)p(h)]. \tag{B.5}$$

This implies that performing the MAP decision on the decorrelated signals as in (B.3) is equivalent to performing the MAP decision on the original signals as in (8).

References

[1] Bishop C M 2007 *Pattern Recognition and Machine Learning* (New York: Springer)

[2] Farwell L A and Donchin E 1988 Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials *Electroencephalogr. Clin. Neurophysiol.* **70** 510–23

[3] Hill N J, Farquhar J, Martens S M M, Biessman F and Schölkopf B 2009 Effects of stimulus type and of error-correcting code design on BCI speller performance *Advances in Neural Information Processing* vol 21 ed D Schuurmans and Y Bengio pp 665–72

[4] Sutter E E 1992 The brain response interface: communication through visually induced electrical–brain responses *J. Microcomput. Appl.* **15** 31–45

[5] Kaper M, Meinicke P, Grossekathoefer U, Lingner T and Ritter H 2004 BCI competition 2003–data set IIb: support vector machines for the P300 speller paradigm *IEEE Trans. Biomed. Eng.* **51** 1073–6

[6] Krusienski D J, Sellers E W, Cabestaing F, Bayoudh S, McFarland D J, Vaughan T M and Wolpaw J R 2006 A comparison of classification techniques for the P300 speller *J. Neural Eng.* **3** 299–305

[7] Thulasidas M, Guan C and Wu J 2006 Robust classification of EEG signal for brain–computer interface *IEEE Trans. Neural Syst. Rehabil. Eng.* **14** 24–9

[8] Rakotomamonjy A and Guigue V 2008 BCI competition III: dataset II—ensemble of SVMs for BCI P300 speller *IEEE Trans. Biomed. Eng.* **55** 1147–54

[9] Rivet B, Souloumiac A, Attina V and Gibert G 2009 xDAWN algorithm to enhance evoked potentials: application to brain–computer interface *IEEE Trans. Biomed. Eng.* **56** 2035–43

[10] Kaper M and Ritter H 2004 Progress in P300-based brain–computer interfacing *IEEE Int. Workshop on Biomed. Circuits Syst. BIOCAS* pp S35INV–9

[11] Ng A Y and Jordan M I 2001 On discriminative versus generative classifiers: a comparison of logistic regression and naive Bayes *NIPS* 841–8

[12] Rubinstein Y D and Hastie T 1997 Discriminative versus informative learning *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining* pp 49–53

[13] Birbaumer N 2006 Brain–computer interface research: coming of age *Clin. Neurophysiol.* **117** 479–83

[14] Dobkin B H 2007 Brain–computer interface technology as a tool to augment plasticity and outcomes for neurological rehabilitation *J. Physiol.* **579** 637–42

[15] Sutton S, Braren M, Zubin J and John E R 1965 Evoked-potential correlates of stimulus uncertainty *Science* **150** 1187–8

[16] Fabiani M, Gratton G and Federmeier K D 2007 Event-related brain potentials: methods, theory, and applications *Handbook of Psychophysiology* ed J T Cacioppo, L G Tassinary and G G Berntson (Cambridge: Cambridge University Press) pp 85–119

[17] Rugg M D and Coles M G H 1996 *Electrophysiology of Mind: Event-Related Brain Potentials and Cognition* (Oxford: Oxford University Press)

[18] Martens S M M, Hill N J, Farquhar J and Schölkopf B 2009 Overlap and refractory effects in a brain–computer interface speller based on the visual P300 event-related potential *J. Neural Eng.* **6** 026003

[19] Kay S M 1993 *Fundamentals of Statistical Signal Processing, Vol I (Estimation Theory)* (Englewood Cliffs, NJ: Prentice Hall) p 9

[20] Cortes C and Vapnik V 1995 Support vector networks *Mach. Learn.* **20** 273–97

[21] Schölkopf B and Smola A J 2002 *Learning with Kernels* (Cambridge, MA: MIT Press)

[22] Boser B E, Guyton I M and Vapnik V 1992 A training algorithm for optimal margin classifiers *Proc. 5th Annual ACM Workshop on Computational Learning Theory* ed D Haussler (Pittsburgh, PA: ACM) pp 144–152

[23] Vapnik V 1995 *The Nature of Statistical Learning Theory* (Berlin: Springer)

[24] Dumais S 1998 Using SVMs for text categorization *IEEE Intell. Syst. Mag.* 13(4)

[25] Deselaers T, Keysers D and Ney H 2005 Discriminative training for object recognition using image patches *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* **2** 157–62

[26] Lotte F, Congedo M, Lecuyer A, Lamarche F and Arnaldi B 2007 A review of classification algorithms for EEG-based brain–computer interfaces *J. Neural Eng.* **4** R1–13

[27] Guger C, Daban S, Sellers E, Holzner C, Krausz G, Carabalona R, Gramatica F and Edlinger G 2009 How many people are able to control a P300-based brain–computer interface (BCI)? *Neurosci. Lett.* **462** 94–8

[28] Proakis J G 2008 *Digital Communications* 5th edn (New York: McGraw-Hill)