Preprocessing and Meta-Classification for Brain-Computer Interfaces

Paul S. Hammon* and Virginia R. de Sa

***Abstract*—A brain-computer interface (BCI) is a system which allows direct translation of brain states into actions, bypassing the usual muscular pathways. A BCI system works by extracting user brain signals, applying machine learning algorithms to classify the user's brain state, and performing a computer-controlled action. Our goal is to improve brain state classification. Perhaps the most obvious way to improve classification performance is the selection of an advanced learning algorithm. However, it is now well known in the BCI community that careful selection of preprocessing steps is crucial to the success of any classification scheme. Furthermore, recent work indicates that combining the output of multiple classifiers (meta-classification) leads to improved classification rates relative to single classifiers (Dornhege *et al.*, 2004). In this paper, we develop an automated approach which systematically analyzes the relative contributions of different preprocessing and meta-classification approaches. We apply this procedure to three data sets drawn from BCI Competition 2003 (Blankertz *et al.*, 2004) and BCI Competition III (Blankertz *et al.*, 2006), each of which exhibit very different characteristics. Our final classification results compare favorably with those from past BCI competitions. Additionally, we analyze the relative contributions of individual preprocessing and meta-classification choices and discuss which types of BCI data benefit most from specific algorithms.**

***Index Terms*—Brain-computer interface (BCI), feature extraction, meta-classification, preprocessing.**

I. Introduction

OVER the last two decades the field of brain-computer interfaces (BCIs) has developed with the goal of providing a direct means of communicating internal brain states to the external world [4]. The ultimate goal of BCI research is to build a complete system which allows the user to directly communicate with the external world through modulation of his or her brain signals. A complete BCI system includes acquisition of brain signals, processing and classification of the acquired signals, feedback of the interpreted brain state, and use of the classified signals to perform a task. Improvements of BCI systems as a whole can be achieved by improving any of these subsystems. This paper discusses an automated procedure for analyzing which preprocessing and meta-classification techniques are most effective for different types of BCI data. We apply this procedure to three different types of BCI data and discuss the results. This paper grew out of a submission to BCI Competition III [3], an international competition designed to bring together researchers from signal processing, machine learning, and brain sciences to identify and hopefully improve the current state-of-the-art in BCI. We entered this competition for data set I with an earlier version of the approach described here and placed second out of a field of 27 different submissions [3].

This paper is organized as follows. Section II describes the characteristics of the three different data sets used in this paper. Sections III–V describe the preprocessing, classification, and meta-classification algorithms analyzed in this study. Sections VI and VII analyze our results and discuss which algorithms are most effective for different types of BCI data.

II. Data Sets

Our analysis is performed on three different data sets from BCI Competition 2003 [2] and BCI Competition III [3]. Each of the three data sets are from somewhat different BCI paradigms and have distinct characteristics, which are discussed in the remainder of this section. Each data set includes a set of labeled training examples and unlabeled test examples. For the purpose of the competition, examination of the unlabeled test data was allowed.

The main feature in BCI Competition 2003 Set Ia [5] is slow cortical potentials (SCPs)—slow-building positivity or negativity over the central midline (Cz). These data are from a 3.5-s period during which the user received visual feedback of the voltage at electrode Cz. Data are recorded from six EEG electrodes and sampled at 256 Hz. There are 268 labeled training examples from a mix of first- and second-day trials, and 293 unlabeled test examples all drawn from second-day trials. We will refer to this data set as EEG-SCP.

Two of the data sets used in our study are based on real and imagined movement: data set I from BCI Competition III [6], and data set IV from BCI Competition 2003 [7]. There are two distinct types of features one would expect from electrocorticograph (ECoG) or electroencephalograph (EEG) recordings of real or imagined movement: event-related desynchronization (ERD) and movement-related potentials (MRP). The ERD signal feature is manifested as a power reduction in the mu (8–12 Hz) and beta (13–28 Hz) bands located above portions of sensorimotor cortex during real or imagined movement [8]. MRPs have been recorded prior to actual movement (and are expected in imagined movement) and can be seen in EEG data as an increasing negativity over motor cortex with peak negative deflection slightly after movement onset [9].

Manuscript received January 20, 2006; revised July 13, 2006. This work was supported in part by the National Science Foundation (NSF) under Grant DGE-0333451 and in part by NSF CAREER Award 0133996. *Asterisk indicates corresponding author.*

*P. S. Hammon is with the department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0409 USA (e-mail: phammon@ucsd.edu).

V. R. de Sa is with the department of Cognitive Science, University of California at San Diego, La Jolla, CA 92093-0515 USA (e-mail: desa@cogsci.ucsd.edu).

Digital Object Identifier 10.1109/TBME.2006.888833

Competition 2003 set IV data is from an actual movement task with two classes representing left versus right hand self-paced typing. There are 500 ms of data sampled at 1 kHz and ending 130 ms before key press. Thus, although the task requires actual movement, the data used for classification are recorded prior to movement onset. The data are recorded from 28 Ag/AgCl EEG electrodes and consist of 316 labeled training examples and 100 unlabeled test examples. Average spectra and time series of the two classes of training data reveal no prominent ERD, but there is a small MRP signal. All of the training and test data were recorded on the same day. We will refer to this data set as EEG-MRP.

Competition III Set I is from the ECoG of an epileptic patient making cued imagined movements of either the tongue or left small finger. Each trial is 3 s long beginning 500 ms after the visual cue and is sampled at 1 kHz from an 8×8 grid of platinum ECoG electrodes. The data set consists of 278 labeled training examples and 100 unlabeled test examples. One of the challenges of this data set is that the training set and test set data were recorded in separate sessions about one week apart. An analysis of the average temporal and spectral content of the two classes shows a large ERD accompanied by a weaker MRP. We will refer to this data set as ECoG-ERD.

III. Classification Approach

In this paper, we break brain signal classification down into three distinct parts: combining preprocessing algorithms to extract feature vectors, training individual classifiers on the feature vectors, and combining multiple classifiers using meta-classification. Here we define a feature vector as a vectorized set of processed data from a single trial which is ready for training or testing by a learning algorithm. In this paper, we focus on preprocessing and meta-classification for several reasons. First, data from brain signals can be quite high-dimensional, and potentially full of artifacts. Proper application of preprocessing steps can reduce data dimensionality and emphasize portions of the data with discriminative power, thereby reducing computation time and improving classification rates. Another reason for concentrating on preprocessing and feature extraction—as opposed to developing advanced learning algorithms—is that there are a number of good, multi-purpose machine learning algorithms available, making the choice of a machine learning algorithm less critical to overall classification success. Furthermore, meta-classification approaches have been shown to be successful on BCI data [1]. Our approach involves several steps. First we combine different preprocessing techniques to generate a large number of feature vectors. Next individual classifiers are trained on each feature vector, and then meta-classification algorithms are trained using the most promising individual classifiers.

When choosing a classifier to apply to the test data, it is important to avoid selecting one which generalizes poorly due to overfitting [10]. To avoid this, we randomly select 50% of the complete training data to form a reduced training set, while the other 50% is set aside as a holdout set. All parameter selection is done on the reduced training set: preprocessing parameter selection, individual classifier training, and meta-classifier training.

TABLE I
Preprocessing Stages

Preprocessing stage	Description
1. Subsampling	mandatory (EEG-MRP, ECoG-ERD)
2. Frequency Filtering	high-pass, low-pass, band-pass, or none
3. Channel Scaling	optional
4. Channel Selection	optional
5. Spatial Filtering	ICA, CSP, both, or neither
6. Frequency Decomposition	Welch, AR modeling, wavelets, or none
7. Postprocessing	log-variance or none

Additionally, the individual classifiers used for meta-classification are selected based on cross-validation performance on the reduced training set. All trained classifiers are then tested on the holdout set, and the classifier with the best performance on the holdout set is selected for use with the test set. Before testing, the winning classifier is re-trained on the complete (reduced + holdout) training set. After determining the classification results for the winning classifier, all other classifiers were tested on the test set for later analysis (see Section VI).

IV. Preprocessing

We construct a feature vector by running several different preprocessing algorithms on the raw data. We employ a total of seven different preprocessing stages which are combined sequentially to generate a variety of different feature vectors. This set includes: subsampling, frequency filtering, channel scaling, channel selection, spatial filtering, frequency decomposition, and postprocessing. Parameters for individual preprocessing algorithms are selected individually for each data set using ten-fold cross-validation on the reduced training set. To generate a feature vector, we follow the stages from Table I in the order listed. We test all combinations allowed in Table I, resulting in $1 \times 4 \times 2 \times 2 \times 4 \times 4 \times 2 = 512$ different feature vectors. The preprocessing stages were designed to minimize nonsensical combinations of processing stages, but any poorly motivated combinations that do arise will be weeded out due to poor performance on the holdout set. Each preprocessing step is described below.

A. Subsampling

The first preprocessing step is to reduce the dimensionality of the data by subsampling ECoG-ERD and EEG-MRP to 250 Hz (EEG-SCP data were recorded at 256 Hz and are not subsampled). This step is taken solely to improve processing time and memory overhead. It is not expected to result in any loss of important information, as the main frequencies of interest remain well below the Nyquist rate of the subsampled data.

B. Frequency Filtering

We implemented high-pass and low-pass FIR filters to address two different issues in the data sets. A high-pass filter with a cutoff of 8 Hz is used to eliminate wandering direct current levels of the individual channels. Alternating current power pickup is attenuated by a low-pass filter with 45 Hz cutoff

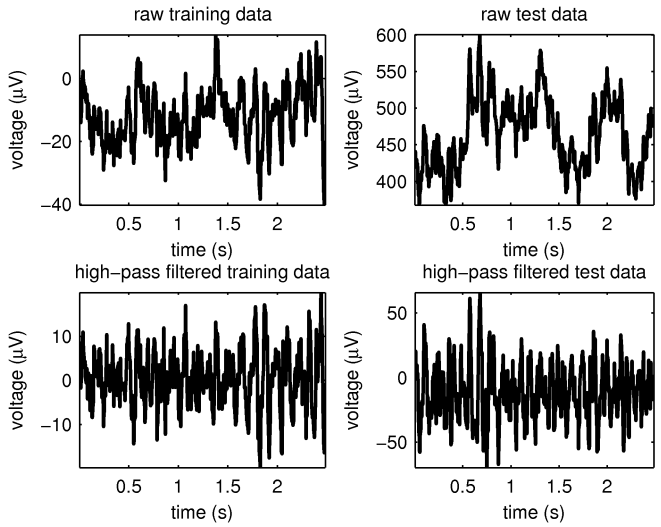

Fig. 1. This figure shows the motivation for high-pass filtering and channel scaling with the ECoG-ERD data. The test data have a higher amplitude and considerable offset compared to the training data. After high-pass filtering, the test data still exhibit a higher variance than the training data. A final step of scaling the filtered channel data fixes this.

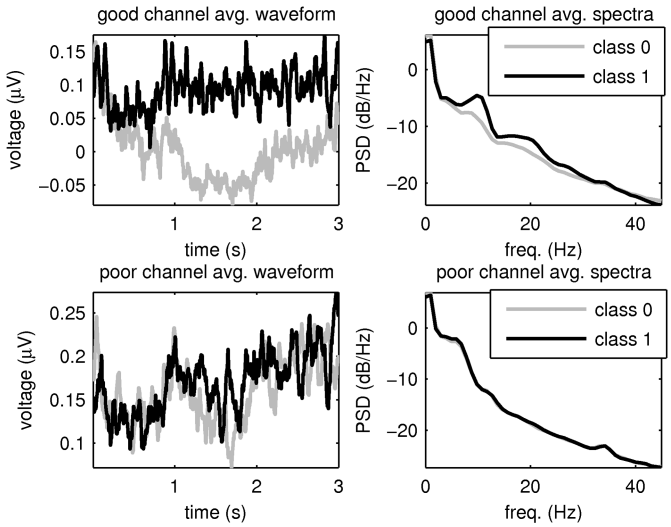

Fig. 2. This figure shows an example of how "good" channels are chosen for the ECoG-ERD data. Channels are labeled as "good" if their time series average or spectral average were visually discriminable. Note how the average spectra and time series in the top panels (from a "good" channel) show some separation, whereas the bottom panels (from a "bad" channel) show average signals with large overlap.

(chosen below the 50 Hz frequency of European power). This preprocessing stage includes high-pass, low-pass, both (band-pass) or no filtering. The effect of high-pass filtering is illustrated in Fig. 1.

C. Channel Scaling

A preprocessing step inspired by examination of the (unlabeled) test data from ECoG-ERD is scaling of the EEG or ECoG channels to mean zero and standard deviation one. This preprocessing step was added after noting that some of the channels in the test data have considerably different variances than in the training data, as shown in Fig. 1. This difference may be due to increased resistance over the time that the electrodes are applied (or implanted), resulting in increased voltage readings.

D. Channel Selection

We hand-selected a set of channels which seemed more discriminative than average. This selection process was carried out separately for the three data sets and involved looking at average spectral and average time domain responses at each channel for the two different classes. Channels were selected as "good" if they visibly discriminated, on average, between the two classes (see Fig. 2). In our analysis, we used the complete training set for this step.[1] It is worth noting that there are automated approaches available for channel selection [11], but this simple procedure was sufficient for our purposes. We note that these "good" channels tend to be spatially contiguous, and we believe that they likely lie over the motor cortex in ECoG-ERD and EEG-MRP.

[1]In general, it is best to use only the reduced training set even for visual selection of preprocessing parameters. In the case of our data, our approach likely did little harm as randomly selected 50% splits of the training data from all three data sets show average behavior that is similar to that of the complete training data.

E. Spatial Filtering

One type of preprocessing which is commonly used in BCI systems is spatial filtering. The goal of this technique is to create a new set of derived channels which enhance the separability of the data.

One type of spatial filtering which we tested as a preprocessing step is independent component analysis (ICA). The ICA model posits a set of statistically independent signals (according to some dependency metric) which are linearly mixed according to $X = AS$, where the (mixed) data is represented as X, the independent sources as S, and the mixing matrix as A. The goal of ICA is to determine an estimate $\hat{A}$ of the matrix A and invert it to estimate the underlying independent signals: $\hat{S} = WX$, where $W = \hat{A}^{-1}$.

The net electrical potential recorded at the scalp can be interpreted as the superposition of a number of different distributions of scalp electrical potential, each of which results from the electrical activity of a different brain source [12]. In this sense, EEG and ECoG data satisfy the ICA assumption of linear mixing. By applying ICA to the EEG or ECoG channel data, we get a new set of ICA components which, in general, separate out some of the different brain sources and artifacts.

There are a whole class of ICA algorithms which differ on specific assumptions about the data and definitions of independence. We use a MATLAB implementation [13] of fastICA [14], which defines nongaussianity as its measure of independence. As a preprocessing step to ICA we run principal components analysis (PCA) to decorrelate and reduce the number of channels. In the case of noisy data, this preprocessing step often improves the performance of ICA [14]. The number of PCA components to use with fastICA was chosen individually for the three different data sets using cross-validation on the reduced training set.

Common spatial patterns (CSP) is another type of spatial filtering commonly used in BCI systems [15]. The goal of the CSP algorithm is to generate spatial filters which maximally discriminate between two classes. We want to construct a projection matrix which when applied to the data creates a new mixture which has the property that the variance in some channels is maximal for class 1 and minimal for class 0, while a different set of channels have maximal variance for class 0 and minimal variance for class 1. We retain the $2m$ CSPs that best discriminate the two classes, using tenfold cross-validation on the reduced training set to choose a value in the range $m = 1$ through 10 (see [15] for details). Because CSP takes the class labels into account when determining the spatial filter, feature vectors which include this algorithm have the possibility of overfitting. Another aspect worth considering when working with CSPs is that they perform best on appropriately frequency-filtered data, which we do not explicitly require (see Table I and Section VI-A). However, any feature vectors which perform poorly on the holdout set due to either of these two issues will not be selected for use on the test set, and should not have a detrimental effect on error rates of the final selected classifiers.

F. Frequency Decomposition

The sensorimotor rhythms described in Section II have characteristics which are especially pronounced in the frequency domain. Thus, one of the processing steps we include in our feature extraction is spectral estimation. One common approach to spectral estimation is Welch's method for power spectral density estimation [16]. Welch's method estimates the power spectra by breaking the signal into overlapping segments, each of which is windowed and then transformed using the fast Fourier transform (FFT). The individual FFT responses are then averaged together and scaled to yield the spectral estimate. We use MATLAB's implementation of Welch's method, pwelch.m, with a Hamming window on eight segments of 50% overlap. The power at each point in the power spectrum is used as the feature vector.

An alternative approach to spectral estimation uses autoregressive (AR) modeling. An AR system is of the form $y(t) = -\sum_k a_k y(t-k) + w(t)$, where $w(t)$ is a white-noise random process and a_k are the autoregressive coefficients. Thus, the output of this system results from white noise being passed through an all-pole filter. Due to the all-pole restriction, AR models can give good approximations of signals which have "peaky" spectra, such as that of the ECoG-ERD channel shown in Fig. 2.

While there are a number of different approaches for estimating the AR coefficients a_k, we found the Burg Algorithm [17] to yield estimates which worked well for classification. We use MATLAB's implementation of the algorithm, arburg.m. Instead of the AR-derived spectra, we use the AR coefficients for classification. The AR model order was determined separately for each data set using ten-fold cross-validation on the reduced training set with a range of possible orders of 2 through 10.

One final frequency decomposition approach we consider is the discrete wavelet transform (DWT). This transform represents the data in a format which retains some information from both the frequency and the time domain. We run one-dimensional DWTs on each channel of the data. The wavelet function (Daubechies, or Symlets), order (2 through 8) and decomposition level (1 through 6) are selected for each data set using ten-fold cross-validation on the reduced training set.

G. Postprocessing

A final processing step often used with CSPs is to calculate the fraction of total variance accounted for by each of the $2m$ retained CSPs, and then use the logarithm of this output as a feature vector. The motivation for this processing step is that the total fraction of the variance accounted for by each of the different common spatial patterns should be very good features for discriminating between the two classes. We chose to include this as a processing option, but did not restrict its use to CSPs.

V. Learning Algorithms

A. Support Vector Machines

We chose support vector machines (SVMs) [18] as a learning algorithm because they have performed well as a classifier in past BCI competitions (e.g., [19]), and because they generally perform well on a variety of classification problems. Additionally, SVMs allow for rapid classification from trained models and are capable of handling very high-dimensional input vectors. We used an SVM implementation called $\mathrm{SVM}^{\mathrm{light}}$ [18] with a MATLAB wrapper [20]. SVM classifiers are created for each feature vector by first selecting a kernel function and a misclassification penalty using cross-validation. We test linear, polynomial, and radial basis kernel functions, and misclassification penalties of 0.01, 0.1, 1, 10, and 100.

B. L_1-Regularized Logistic Regression

We chose to compare the classification accuracy of SVMs with L_1-regularized logistic regression because the latter has a sample complexity (the number of examples needed to train the classifier) which grows logarithmically in the number of irrelevant features, while the sample complexity of SVMs (and other rotationally invariant learning algorithms) grows linearly with the number of irrelevant features [21]. One of the constraints shared by all of the BCI data sets in this study is relatively few training examples compared to the dimensionality of the data. We expected that L_1-regularized logistic regression would out-perform SVMs if many of the dimensions in the feature vector were unrelated to the training labels.

Logistic regression is a binary classifier that models the probability of class 1 as

$$p(y = 1|x;\theta) = \frac{1}{1 + \exp(-\theta^T x)} \tag{1}$$

where x is the data, y is a binary class label, and θ is a parameter vector.

Training the classifier involves computing a maximum-likelihood estimate of the parameter θ. First we form the log-likelihood function

$$\ell(\theta) = \log \prod_i^n p(y^{(i)}|x^{(i)};\theta) = \sum_{i=1}^n \log p(y^{(i)}|x^{(i)};\theta) \tag{2}$$

where i indexes over the individual training samples of the same class. To find an estimate of θ, we maximize the regularized log-likelihood

$$\hat{\theta} = \arg\max_{\theta}\{\ell(\theta) - \alpha\|\theta\|_1\}. \quad (3)$$

We implemented our own version of L_1-regularized logistic regression using batch gradient descent.

In order to choose a good value for the regularization parameter α, we test $\alpha = \{0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ and select the value which gives the best cross-validation error on the reduced training set. Our gradient descent algorithm uses an initial learning rate of 0.001 with a constant annealing factor of 0.99.

C. Meta-Classification

After training the two different learning algorithms on each feature vector, we also perform meta-classification by combining multiple classifiers of both types (SVM and logistic) which performed well in cross-validation tests on the reduced training set. We expect meta-classification to improve classification accuracy on the test set by reducing the effects of noise and overfitting and by combining information from independent features. We compare five different types of meta-classifiers.

1) HIERARCHY: A group of individual classifiers is selected from the complete set of 512 based on reduced training set classification rates. Then the scaled outputs of this set of classifiers are used as the input to train a hierarchical classifier of the same type (either SVM or logistic regression).
2) CONCAT: The feature vectors of a set of classifiers are concatenated to form new feature vectors. These new feature vectors are then used to train a new classifier. Although this method does not make use of multiple classifiers and is, therefore, not a true meta-classification approach, we include it here for comparison.
3) PROD: The outputs of a set of classifiers are scaled to be in the range of $[0, 1]$, interpreted as probabilities, multiplied together, and then re-normalized. Feature vectors with probability greater than 0.5 are assigned to class 1, and all others are assigned to class 0.
4) AVG: Similar to PROD, but in this case the individual probabilities are simply averaged together. Feature vectors are assigned to class 1 for average probability over 0.5 and class 0 otherwise.
5) VOTE: Each classifier is given a single vote for class 1 or class 0. The final class is determined by simple majority.

The classification schemes HIERARCHY and CONCAT are described in [1]. We tested CONCAT with two different approaches: with individual scaling of all features to mean zero and standard deviation one before concatenation, and without any scaling of the features ("no scale"). From [1], we expect HIERARCHY to out-perform CONCAT.

The two different probability-based classifiers, PROD and AVG are discussed in [22]. PROD is expected to give good classification results when the individual classifiers are mostly independent and have well-calibrated probabilities, while AVG works best when the different classifiers are highly correlated and is more robust with respect to poorly calibrated probabilities. Thus, we expect HIERARCHY or AVG to perform best on the holdout and test sets. In order to compute probabilities from the SVM scores, we use a simple scaling method: take the raw SVM scores, which typically lie in the rough range $[-a, a]$, and scale them to the range $[0, 1]$ by adding the minimum score and then dividing by the new maximum score. More advanced methods exist for producing probability estimates from unbounded classifier output scores, such as Platt scaling and isotonic regression [23]. These methods can improve probability estimates, but they generally require more training examples then were available in the reduced training set, and they often have their own parameters which would take an additional level of cross-validation to estimate. Thus, we chose to use the simplest method.

When choosing the classifiers for meta-classification we first create a sorted list of all individual classifiers (including both logistic and SVM) based on their error rates on the reduced training set. Using this list, we devised two different methods for selecting individual classifiers for use in meta-classification.

1) TOPN: select the top performing individual classifiers, where $N = 3$, 5, 7, or 9.
2) $< 25\%$: select all individual classifiers with less than 25% cross-validation error on the reduced training set (we do not test this with CONCAT due to memory limitations).

VI. Results

Although we eventually tested every classifier on the test set, we first chose a classifier as our mock competition entry for each data set. For EEG-SCP and ECoG-ERD, we simply selected the classifiers with lowest error on the holdout set as our mock entry. For EEG-MRP, we noted that the cross-validation error on the reduced training set and the error on the holdout set were noticeably different, and we attributed this to especially noisy data. We decided to re-train all individual classifiers on the complete training set and use the average cross-validation error to select classifiers for inclusion in meta-classification, hoping that the additional training examples would have a more positive effect than any overfitting possibly induced by this approach.

In the case of a tie while choosing the mock competition classifier, we selected the classifier which we expected would work best on the test set. Specifically, we chose HIERARCHY and AVG over PROD, and meta-classification using more classifiers over fewer classifiers.

The classifiers we chose for our mock competition entries, as well as where they would have placed in their respective competitions, are listed in "mock" segment of Table II. After our mock competition, we tested all of our classifiers on the test set. The classifiers which had the best performance of all tested in this study are listed in the "best" segment of Table II.

There are several interesting conclusions to draw from these results. First of all, meta-classifiers typically perform better on the test set than individual classifiers. This suggests that meta-classification strategies will generally outperform single classifiers, which is also suggested in [1]. Furthermore, including more classifiers in the meta-classification scheme will generally further increase performance. It is also worth noting that the classifiers we selected for our mock entries into the data set

TABLE II
TRAINING AND TEST ERROR RATES FOR BOTH MOCK COMPETITION ENTRIES AND BEST OF ALL TESTED CLASSIFIERS

Data Set	Report	Classifier	Holdout	Test	Place
EEG-SCP	mock	$< 25\%$ VOTE	0.13	0.07	1 of 16
	best	$< 25\%$ VOTE	0.13	0.07	
EEG-MRP	mock	$< 25\%$ VOTE	0.15	0.12	1 of 16
	best	$< 25\%$ VOTE	0.15	0.12	
ECoG-ERD	mock	$< 25\%$ HIER. SVM	0.05	0.12	2 of 28
	best	single SVM	0.08	0.09	

competitions ended up being the best overall classifiers for two of the three data sets considered. This suggests that our automated method for selecting preprocessing and meta-classification approaches is effective.

While the specific meta-classifier selected in our mock competition for ECoG-ERD performed well on the test set, a more in-depth analysis reveals that most classifiers from this data set perform quite poorly on the test set. The difficulty with this data set was due to the major statistical differences of the data in the training and test sets, as shown in Fig. 1. In a sense, this data set presents an ill-posed learning problem: train a classifier on data drawn from one distribution such that it will perform well on data from a second, unknown distribution. Restricting allowable preprocessing steps can improve performance on average (see Section VI-A3).

A. *Preprocessing Analysis for Individual Classifiers*

We developed a simple approach for analyzing whether a preprocessing step was associated with improved classification error for individual classifiers (meta-classifiers are analyzed in Section VI-B). We compute the average classification rate for all classifiers including a given preprocessing step, and then we subtract the average classification rate for all classifiers which do not include that preprocessing step. The resulting number indicates how much a preprocessing step improves classification rates on average. We display the results graphically by mapping the improvement in classification rate to a grayscale value, with the largest increase mapping to white, the largest decrease to black, and no change to gray. These results are shown in Fig. 3.

One overall result worth noting is that SVMs and logistic regression perform about equally well, with neither algorithm consistently outperforming the other. This was somewhat surprising, as we expected L_1-regularized logistic regression to outperform SVMs because the high dimensionality of the feature vectors makes it more likely that some dimensions are uncorrelated with the labels. The fact that these two classifiers perform similarly on the three data sets serves to highlight our assertion that preprocessing and meta-classification are more important in BCI data analysis than the classifier itself. Investigating the fourth panel of Fig. 3 serves to further reinforce this point: each of the three data sets studied benefited most from a different set of preprocessing options. We discuss the individual data sets in the following subsections.

One other item worth discussing is interactions among individual preprocessing steps. While we did not do this type

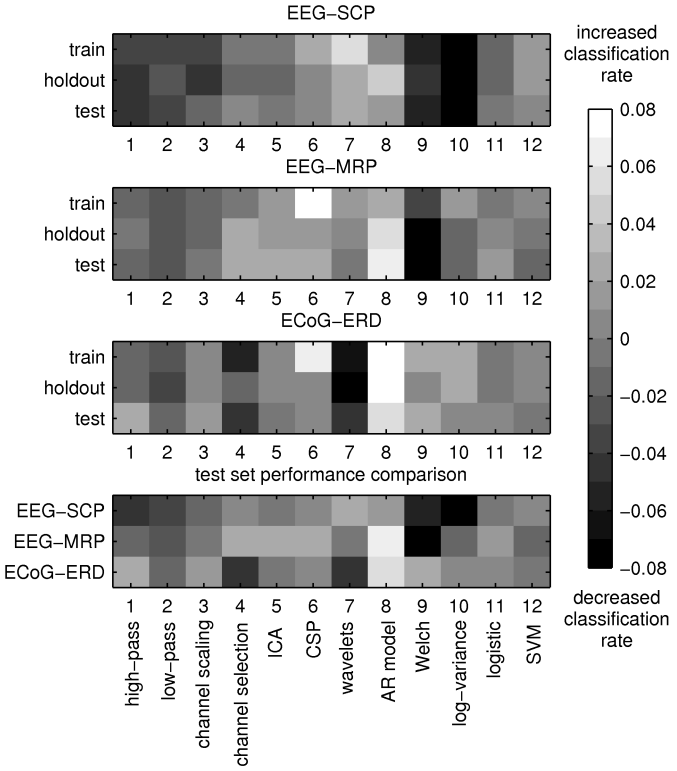

Fig. 3. This figure graphically represents which preprocessing steps lead to successful classification for EEG-SCP, EEG-MRP, and ECoG-ERD. Lighter shades indicate that including a preprocessing step improves classification accuracy on average, darker shades indicate a reduction in accuracy, and gray indicates no change. The top three cells allow for comparison between performance on reduced training (cross-validation error), holdout, and test portions of each data set, and the bottom cell allows for comparison between data sets.

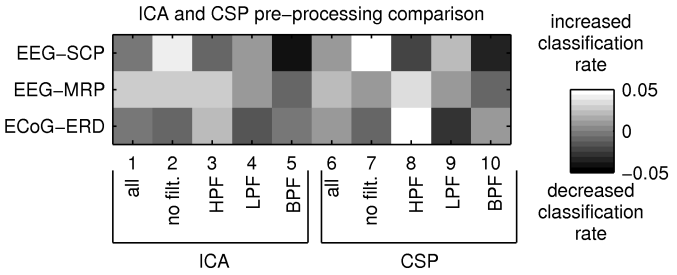

Fig. 4. This figure graphically represents the interactions between ICA, CSP, and frequency filtering preprocessing steps. HPF, LPF, and BPF stand for high-, low-, and band-pass filters, respectively. Average results for all classifiers including ICA and CSP are listed in the "all" column. Lighter shades indicate that including a preprocessing step improves classification accuracy on average, darker shades indicate a reduction in accuracy, and gray indicates no change. Classification rates are shown for test sets only.

of analysis for most preprocessing steps, we did consider the effects of frequency filtering on the success of ICA and CSP preprocessing steps. It is generally recognized among BCI researchers that the success of the CSP algorithm depends on frequency filtering the data over an appropriate frequency range. In Fig. 4, we compare the relative increase (lighter shades) and decrease (darker shades) of classification accuracy when combining frequency filtering (high-pass, low-pass, band-pass, or none) and spatial filtering (ICA, CSP). Instead of breaking down training, holdout, and test set classification rates as in Fig. 3,

we only report the test set classification rates. The results of this analysis indicate that frequency filtering within a frequency range appropriate for the individual data sets improves classification to a similar degree for both ICA and CSP. Specifically, the classification for the EEG-SCP data set works best when either no filtering or low-pass filtering is combined with ICA/CSP, while the EEG-MRP and ECoG-ERD data sets have higher classification rates when combining high-pass or band-pass filtering and ICA/CSP.

1) EEG-SCP: Fig. 3 reveals several interesting things about this data set. First of all, no preprocessing steps show large performance differences between training, holdout, and test sets, indicating that the data are relatively classifiable and that little overfitting is taking place. The prominent characteristic in this data set is the slow cortical potential (SCP), a shift in mean signal level which evolves over several seconds. Thus, one would expect poor performance from preprocessing steps which do not emphasize this aspect of the data. This is exactly what we see, with AR and Welch frequency decomposition, high-pass filtering, and the log-variance transform processing steps performing especially poorly on this data set. Note that high-pass filtering selectively removes slowly evolving signals and performs particularly poorly.

2) EEG-MRP: The differences between training and holdout error for this data set are evident in Fig. 3. For example, CSPs overfit on the training data. Channel selection, ICA, and CSPs appear to be good preprocessing steps. AR modeling performs well on the test set, but does not appear as an especially good choice on the training set. Welch's method and low-pass filtering are not good choices. Overall, this data set was relatively noisy, making classification and analysis difficult.

3) ECoG-ERD: The differences between the training and test portions of this data set are reflected in the changes in preprocessing performance between the holdout and test sets. While AR modeling and the log-variance transform perform especially well on the training and holdout sets, their performance decreases on the test set. One of the best performers on the test set is high-pass filtering, which helps to remove some of the level shifts and drifting channels in the test set (see Fig. 1). ICA and CSP also perform well on the test set.

An informal analysis of interactions among different preprocessing steps revealed that many of the classifiers which performed well on the training data but poorly on the test data lacked either high-pass or low-pass filtering, or included channel scaling. Re-running our meta-classification analysis with the additional requirements of enforcing high-pass and low-pass filtering and excluding channel scaling led to large performance improvements in general. While the test set classification rate of best-performing meta-classifiers did not change substantially, average classification rates of the complete set of meta-classifiers improved dramatically. Thus, when working with data sets that exhibit session-to-session differences, our approach outlined in this paper will be most effective when *a priori* knowledge is used to restrict the allowable preprocessing choices.

B. Meta-Classification Analysis

We analyze the effectiveness of the different meta-classification options using the same approach as described in Section VI-A: meta-classification options which improve classification rates on average are mapped to lighter shades, those that decrease classification rates are mapped to darker shades, and those with no change are mapped to gray. The results are shown in Fig. 5. For brevity, we include only test set classification rates.

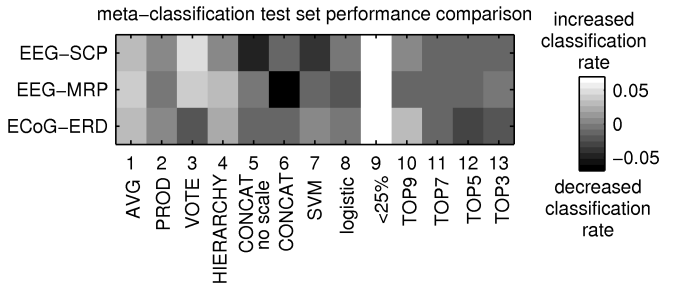

Fig. 5. This figure graphically represents which meta-classification approaches lead to successful classification for EEG-SCP, EEG-MRP, and ECoG-ERD. Lighter shades indicate that including a meta-classification approach improves classification accuracy on average, darker shades indicate a reduction in accuracy, and gray indicates no change. Classification rates are shown for test sets only.

One trend worth noting is that meta-classification approaches in general do well, even in the case when the individual classifiers selected do somewhat poorly (as is the case with ECoG-ERD). CONCAT in general performs poorly, and PROD performs somewhat worse than AVG or VOTE. PROD may work better with larger training sets or the addition of Platt scaling or isotonic regression, which would allow better class probability estimates. AVG and HIERARCHY perform well in all three data sets, with VOTE performing well on all but ECoG-ERD. Including more classifiers improves classification rates for all three data sets, with the greatest improvements coming when including all classifiers with less than 25% reduced training set cross-validation error.

VII. Conclusion

We have compared preprocessing, machine learning, and meta-classification approaches on three different types of BCI data sets. This comparison reveals the merits of individual preprocessing and meta-classification approaches for different classes of data, and should be instructive when designing new BCI systems. We have also demonstrated a general feature extraction and classification approach which should be useful for designing classifiers tailored to specific subjects and experiments.

Our approach works well when the training and test sets are similar, and careful selection of preprocessing steps can make it effective when training and test sets differ in a predictable way. Most preprocessing steps considered were useful for at least one of the data sets, though few algorithms were consistently beneficial across all three data sets. Thus, it is important to carefully select a set of preprocessing steps which are well-suited for a specific BCI classification problem. Our classification method suggests one way of doing this.

Meta-classifiers are clearly superior to single classifiers in the BCI data sets that we studied. Meta-classification using the VOTE, AVG, or HIERARCHY approaches all perform well.

Including more individual classifiers in the meta-classification scheme can dramatically improve overall performance.

We found no clear winner between SVMs and L_1-regularized logistic regression, though logistic regression may perform somewhat better with noisy data. This supports our claim that careful selection of preprocessing and meta-classification algorithms is more important than the specific learning algorithm used in a BCI system.

ACKNOWLEDGMENT

The authors would like to thank Dr. B. Allison for useful discussions and recommendations of pertinent background material. They would also like to thank the three anonymous reviewers for their detailed and helpful comments and suggestions.

REFERENCES

[1] G. Dornhege, B. Blankertz, G. Curio, and K.-R. Müller, "Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 993–1002, Jun. 2004.

[2] B. Blankertz, K.-R. Müller, G. Curio, T. M. Vaughan, G. Schalk, J. R. Wolpaw, A. Schlögl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schröder, and N. Birbaumer, "The BCI competition 2003: Progress and perspectives in detection and discrimination of EEG single trials," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 1044–1051, Jun. 2004.

[3] B. Blankertz, K.-R. Müller, D. J. Krusienski, G. Schalk, J. R. Wolpaw, A. Schlögl, G. Pfurtscheller, J. del R. Millán, M. Schröder, and N. Birbaumer, "The BCI competition III: Validating alternative approaches to actual BCI problems," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 14, no. 2, pp. 153–159, Jun. 2006.

[4] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain-computer interfaces for communication and control," *Clin. Neurophysiol.*, vol. 113, pp. 767–791, 2002.

[5] N. Birbaumer, H. Flor, N. Ghanayim, T. Hinterberger, I. Iverson, E. Taub, B. Kotchoubey, A. Kübler, and J. Perelmouter, "A brain-controlled spelling device for the completely paralyzed," *Nature*, vol. 398, pp. 297–298, 1999.

[6] T. Lal, T. Hinterberger, G. Widman, M. Schröder, J. Hill, W. Rosenstiel, C. Elger, B. Schölkopf, and N. Birbaumer, "Methods towards invasive human brain computer interfaces," in *Proc. Advances Neural Information Processing Systems (NIPS 04)*, 2005, vol. 17, pp. 737–744.

[7] B. Blankertz, G. Curio, and K.-R. Müller, "Classifying single trial EEG: Towards brain computer interfacing," in *Proc. Advances Neural Information Processing Systems (NIPS 01)*, 2002, vol. 14, pp. 157–164.

[8] C. Neuper, R. Scherer, M. Reiner, and G. Pfurtscheller, "Imagery of motor actions: Differential effects of kinesthetic and visual-motor mode of imagery in single-trial EEG," *Cognit. Brain Res.*, vol. 25, pp. 668–677, 2005.

[9] G. Dornhege, B. Blankertz, G. Curio, and K.-R. Müller, "Combining features for BCI," in *Proc. Advances Neural Inf. Processing Systems (NIPS 02)*, 2003, vol. 15, pp. 1115–1122.

[10] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA: MIT Press, 2004.

[11] T. N. Lal, M. Schröder, T. Hinterberger, J. Weston, M. Bogdan, N. Birbaumer, and B. Schölkopf, "Support vector channel selection in BCI," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 1003–1010, Jun. 2004.

[12] S. Baillet, J. C. Mosher, and R. M. Leahy, "Electromagnetic brain mapping," *IEEE Signal Process. Mag.*, vol. 18, no. 6, pp. 14–30, Nov. 2001.

[13] J. Hurri, H. Gävert, J. Särelä, and A. Hyvärinen, FastICA Package for Matlab 2005 [Online]. Available: http://www.cis.hut.fi/projects/ica/fastica/code/dlcode.shtml

[14] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. New York: Wiley, 2001.

[15] J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg, "Designing optimal spatial filters for single-trial EEG classification in a movement task," *Clin. Neurophysiol.*, vol. 110, pp. 787–798, 1999.

[16] M. Hayes, *Statistical Digital Signal Processing and Modeling*. New York, NY: Wiley, 1996.

[17] S. L. Marple, *Digital Spectral Analysis: With Applications*. Upper Saddle River, NJ: Prentice-Hall, 1987.

[18] T. Joachims, *Advances in Kernel Methods—Support Vector Learning*. Cambridge, MA: MIT Press, 1999, ch. Making large-Scale SVM Learning Practical.

[19] M. Kaper, P. Meinicke, U. Grossekathoefer, T. Lingner, and H. Ritter, "BCI competition 2003—Data set IIb: Support vector machines for the speller paradigm," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 1073–1076, Jun. 2004.

[20] T. Briggs, Matlab/MEX interface to SVM-light 2005 [Online]. Available: http://www.ship.edu/ thb/mexsvm/

[21] A. Y. Ng, "Feature selection, L1 vs. L2 regularization, and rotational invariance," in *Proc. 21st Int. Conf. Machine Learning*, 2004, pp. 592–599.

[22] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.

[23] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proc. 8th Int. Conf. Knowledge Discovery and Data Mining (KDD'02)*, 2002, pp. 694–699.

Paul S. Hammon received the B.S. degree in electrical engineering in 2003 and the M.S. degree in electrical engineering in 2005 (specializing in intelligent systems, robotics, and control) from the University of California at San Diego, La Jolla. He is currently working towards the Ph.D. degree in electrical engineering at the same university.

His main research interests include signal processing and machine learning applied to neural signals.

Virginia R. de Sa received the Ph.D. degree in computer science from the University of Rochester, Rochester, NY. She did postdoctoral work under an NSERC fellowship at the University of Toronto, Toronto, ON, Canada, and at the University of California at San Francisco in theoretical neurobiology as a Sloan fellow.

She joined the Department of Cognitive Science at the University of California at San Diego, La Jolla, in 2001 and is a member of the interdisciplinary programs in neuroscience, computational neuroscience, and cognitive science. Her interests include using machine learning methods to analyze neural signals, creating brain-motivated machine learning algorithms, and creating models of visual processing.