Abstract—Motor imagery classification is known to be highly user dependent. Subspace alignment has been somewhat successful in allowing for unsupervised transfer from one training user to a new user. In this paper we develop a method to weight contributions from subspace alignment to multiple training users to give improved unsupervised transfer performance on the new test user. Ablation analyses show that both the subspace alignment and weighting are critical for improved performance. We also discuss how weighting uses the labels of the training users to better interpret subspace alignment.

Index Terms—Common spatial patterns (CSP), motor imagery, brain-computer interface (BCI), subspace alignment, electroencephalography (EEG), unsupervised transfer learning

I. INTRODUCTION

Motor imagery (MI) brain-computer interface (BCI) is a popular category of BCI systems in which the user imagines parts of her/his body such as the right or left hand and the BCI attempts to detect which limb was imagined by the user [1], [2]. MI-BCI is trained on the electroencephalography (EEG) signal recorded from each user in a calibration session. First, the supervised method of common spatial patterns (CSP) is trained for feature extraction [3] and secondly, a linear classifier on the selected features is trained to select the imagined class [4]. In practice, often times the calibration session is short to avoid fatigue for the user and training of the CSP filters and the classifier are done for each user separately. Due to the non-stationarity and variations in temporal and spatial information of the motor imagery signal, the quality of the trained classifier is negatively affected when the number of training trials is limited. This is often the case in practice. One way to overcome this limitation is to use existing motor imagery data from other users to train a classifier for a new user. However, as described below, differences in brain anatomy and in motor imagery performance between users [5], usually results in poor performance for direct transfer of one user’s MI-BCI to another.

The motor imagery signal is user generated and in EEG-based systems this signal is recorded at the scalp; however, the signal recorded at the scalp depends not only on the location of the involved neurons but also the orientation of the neurons’ dendrites which affects the orientation of the current flow [6]. Therefore, the MI signal across users varies and a classifier trained for one user cannot be readily used for another [7], [8]. Even for the same user, frequent recalibration is often necessary to accommodate possible drifts in the generated motor imagery signal [9], [10]. There are many transfer learning attempts that use existing data to train a classifier for a new user in an unsupervised fashion, i.e., with unlabeled calibration data for the new user [11]. Among these methods, subspace alignment [12] finds a linear mapping to adapt the features from the source domain to the target domain, however, it does not use the available labels from the source domain.

In this work, we propose a novel improvement on subspace alignment that utilizes data from multiple users and weights the subspace-aligned features from the source users by how much in agreement they are in their transfer to the target (new) user. Our method does not use labels from the target user and hence is completely unsupervised with respect to the target user, but is able to leverage multiple training users to assess their applicability when subspace aligned with the target user.

We compare our proposed method with a few baseline methods and show its superiority in improving the subspace alignment features on three different motor imagery datasets.

II. DATASETS

We test our method on three different EEG motor imagery datasets. Pre-processing and data analyses were done in MATLAB.

A. Dataset IVa, BCI Competition III

This dataset [13] from BCI competition III [14] contains 2-class (right hand and foot) motor imagery EEG data from 5 subjects. Data were originally recorded from 118 Ag/AgCl
electrodes; however, we selected 68 channels for analysis based on [15]. For each subject, 280 trials were available. The size of the training set and test set differed for each subject. For both classes combined, there were 168, 224, 84, 56, 28 training trials and 112, 56, 196, 224, 252 test trials for subjects s1, s2, s3, s4 and s5, respectively. The preprocessing of this dataset replicated the steps in [15] and [16]. The time interval of 0.5-2.5s after cue were selected for each trial and the signal was bandpass filtered between 8-30Hz with Matlab’s butter function with order 5, which gives a tenth order Butterworth filter.

B. Dataset IIa, BCI Competition IV

This dataset [17] from BCI competition IV [18] contains 4 motor imagery classes (i.e., left hand, right hand, foot, and tongue) from 9 subjects, and we used two of the classes for analysis (left hand and right hand). Data were recorded with 22 Ag/AgCl electrodes. For each subject, there were 72 training trials and 72 test trials per class. The preprocessing of this dataset was the same as the procedure used for Dataset IVa, BCI competition III. The time interval of 0.5-2.5s after the cue were selected for each trial and the signal was bandpass filtered between 8-30Hz with Matlab’s butter function with order 5, which gives a tenth order Butterworth filter.

C. Dataset Three

This dataset from [10], [19] contains 2-class (left hand and right hand) motor imagery EEG data from 12 subjects. Data were recorded with 64 channels. After balancing the right and left trials, for each subject, there were 93 one-second trials per class in the training session. In the test session, there were 179, 215, 85, 238, 169, 282, 211, 208, 181, 269, 111, 248 trials per class for subjects s1 to s12 respectively. The preprocessing of this dataset used the code from [19]. Data were filtered between 7-30 Hz Matlab’s butter function with order 3, which gives a sixth order Butterworth filter.

III. METHODS

A. Common Spatial Patterns (CSP)

Motor imagery results in desynchronization (or decreased power) of EEG activity in alpha/mu (8-13 Hz) and beta (13-30 Hz) frequency bands over the relevant areas of somato-motor cortex. For distinguishing between motor imagery of two different body parts, it is critical to discriminate spatial differences in this desynchronization at the scalp. Common spatial patterns (CSP) is one of the most commonly used algorithms for extracting such features. CSP finds spatial filters that maximize the variance (or power) of the band-limited spatially filtered signals from one class while minimizing it for the other class [3], [4]. Let $\xi_i$ be the $i$th trial from a preprocessed EEG dataset, and $\xi_i \in \mathbb{R}^{C \times T}$, where $C$ is the number of EEG channels and $T$ is the number of time samples. For each of the two classes, $N_1$ and $N_2$ represent the set comprising trials for classes 1 and 2, respectively. The normalized covariance matrix $\Sigma_y$ can be estimated as:

$$\Sigma_y = \frac{1}{|N_y|} \sum_{i \in N_y} \xi_i \xi_i^\top \frac{\text{trace}(\xi_i \xi_i^\top)}{\text{trace}(\xi_i \xi_i^\top)}$$ (1)

where $y \in \{1, 2\}$ and $|N_y|$ represents the number of trials in $N_y$, $\xi^\top$ indicates the transpose of $\xi$, and the trace is the sum of the diagonal entries. The CSP filter that maximizes variance for class 1 (while minimizing it for class 2) can be found by maximizing the following objective function:

$$\max_w \frac{w^\top \Sigma_1 w}{w^\top (\Sigma_1 + \Sigma_2) w}.$$ (2)

A full set of filters can be optimized by solving the generalized eigenvalue problem $\Sigma_1 w = \lambda(\Sigma_1 + \Sigma_2)w$ [4]. Concatenating the eigenvectors corresponding to the $m$ largest and smallest eigenvalues as columns of matrix $W$, the spatially filtered signal of a trial $\xi_i$ is obtained as $Z = W^\top \xi_i$. The feature extracted by the $k$th spatial filter is:

$$f_k = \log \left( \frac{\text{var}(Z_k)}{\sum_{j=1}^{2m} \text{var}(Z_j)} \right)$$ (3)

where $Z_k (k = 1 \ldots 2m)$ is the $k$th row of matrix $Z$. In this paper, we used $m = 3$ for the CSP parts of all algorithms as is commonly used [4].

B. Subspace Alignment (SA)

Subspace alignment (SA) is an algorithm proposed by Fernando et al. [12], and it is has been widely used for domain adaptation [20] [21]. We used this algorithm to reduce the domain discrepancy between training and test data in our method. In SA, given source data $X_S$ and target data $X_T$, principal component analysis (PCA) is applied separately to the domain discrepancy between training and test data in our method. In SA, given source data $X_S$ and target data $X_T$, principal component analysis (PCA) is applied separately to the source subspace $S_S$ and lower dimensional target subspace $S_T$. Then $X_S$ and $X_T$ are each projected to their respective subspaces by $X_S S_S$ and $X_T S_T$. The source subspace is then mapped to the target subspace through linear transformation. The transformation matrix $M$ is learned by minimizing the Bregman matrix divergence:

$$M^* = \arg\min_M (\|S_S M - S_T\|_F^2)$$ (4)

where $\| \cdot \|_F$ is the Frobenius norm. The Frobenius norm has the orthogonal invariance property, so Equation (4) can be rewritten as:

$$M^* = \arg\min_M (\|S_S^\top S_S M - S_T^\top S_T\|_F^2) = \arg\min_M (\|M - S_T^\top S_T\|_F^2).$$ (5)

The solution to the above is $M^* = S_T^\top S_T$, and thus the subspace aligned coordinate system is $S_a = S_S S_T^\top S_T$. The subspace alignment [12] algorithm is presented in Algorithm 1. We used 2 dimensional subspaces on Dataset IVa from BCI competition III and Dataset IIa from BCI competition IV. For
Dataset Three, we determined the dimension by using subject 1’s training data as training and subject 12’s training data as test data and vice versa. We performed CSP with SA for different dimensions and looked at the classification accuracy of LDA on subject 1 and 12’s training data. The dimension determined for Dataset Three was 6. No test data for any subject was used in this selection process.

**Algorithm 1: Subspace Alignment**

| Input: | Source data $X_S$, Target data $X_T$, Subspace dimension $d$ |
| Output: | Subspace aligned data $X_{S,a}$, $X_{T,a}$ |
| $S_S$ ← PCA($S$, $d$); |
| $S_T$ ← PCA($T$, $d$); |
| $S_a$ ← $S_SS_T$; |
| $X_{S,a} = S_S S_a$; |
| $X_{T,a} = S_T S_a$; |
| return $X_{S,a}$, $X_{T,a}$ |

**C. Proposed Algorithm: SA_weighted**

Let $\{X_1, \ldots, X_n \}$ indicate EEG data from $n$ subjects where subject $j$ is the selected test subject and the rest are training subjects. Each subject contains training trials $X_{tr}^i$ and test trials $X_{te}^i$, and trials are labeled with -1 (class 1) and 1 (class 2).

This algorithm loops through each of the training subjects. For a training subject $i$, CSP is trained from labeled training trials $X_{tr}^i$ and test trials $X_{te}^i$, and the test subject’s test trial $X_{te}^j$ to extract corresponding features. Subspace alignment is then applied to the extracted features to perform domain adaptation. Then Linear Discriminant Analysis (LDA) is trained on the aligned training features and produces a prediction of the labels of the aligned test features. The prediction is produced based on subject $i$’s training data, so it is named $prediction_i$.

After getting $prediction_i$, an inner loop is used to determine the weighting for $prediction_i$. The inner loop loops through subjects other than $i$ and our test subject $j$. For a subject $k$ in the inner loop, the features of his/her training data $X_{tr}^k$ are extracted by the CSP filters trained on $X_{tr}^i$ from the outer loop. Subspace alignment is then applied to the extracted features of $X_{tr}^k$ and $X_{te}^j$. LDA is trained on the aligned training features from subject $k$ and produces a prediction of the labels of the aligned test features. This prediction is trained on subject $k$’s training data, so it is named $prediction_k$.

For each test trial of the test subject, the inner loop predictions from every training subject $k$ are added together. Since the labels of the predictions are either -1 or 1, then after addition, the $sign$ function is applied to get the combined predicted labels. $prediction_i$ is then compared with the combined predicted labels to see $prediction_i$’s agreement rate with the combined labels. The agreement rate is compared with 0.5 before being used as the weighting for $prediction_i$ to generate an ensemble prediction in the outer loop through:

$$ensemble\_prediction = \sum_{i=1}^{n} weight_i \times prediction_i$$

An agreement rate below chance level would suggest that this subject has opposite prediction compared to others, and it is possible that the subspace alignment has mapped class one training data closer to class two test data. We will explain this with an example:

In Fig. 1 and Fig. 2, we plotted the features of subject 1’s training data and subject 3’s test data after SA from dataset IVa BCI competition III. Fig. 1 shows the results with the correct labels for subject 1 and subject 3. But, SA is unsupervised, so if we switch the labels of training data from class one and class two for subject 1, the SA results don’t change (Fig. 2). SA generally works due to a tendency for subjects to be somewhat aligned (for example due to contralateral cortical organization for right vs left imagery), but as it is unsupervised, misalignment is still possible, and if they are close to 180 degrees out of phase, the alignment can still be very useful if the signs are flipped.

The final predicted labels of the test data $X_{te}^j$ are generated after applying the $sign$ function to the $ensemble\_prediction$. The pseudo-code for SA_weighted is depicted in Algorithm 2.

**D. Baseline Algorithms**

We included the results of the supervised CSP algorithm (called CSP) with LDA for classification, as a reference for the performance of the unsupervised algorithms.

The unweighted version of our proposed algorithm (called unweighted) and the weighted but without SA version (called...
Fig. 2. Subspace alignment results of subject 1 and subject 3 from dataset IVa BCI competition III after manually reversing the labels of subject 1’s data before SA. The training data of subject 1 (circles) and test data of subject 3 (crosses) are aligned. The color represents the label of the trials, where red represents class 1 and blue represents class 2.

Algorithm 2: The proposed algorithm: SA_weighted

Input: 
n training dataset \{X_{tr}^{1}, \ldots X_{tr}^{n}\} and the jth subjects’ test data \(X_{te}^{j}\)
Output: prediction of \(X_{te}^{j}\)’s labels
initialize ensemble_pred as a zero array;
for \(i = 1 : n \) and \( i \neq j \) do
  load training data \(X_{tr}^{i}\) for class 1 and \(X_{tr}^{i}\) for class 2;
  initialize \(\Sigma_{tr}^{i}\) and \(\Sigma_{tr}^{i}\) as the average covariance matrix of \(X_{tr}^{i}\) and \(X_{tr}^{i}\);
  \(W_i = \text{CSP}(\Sigma_{tr}^{i}, \Sigma_{tr}^{i})\);
  calculate features \(f_{tr}^{i}\) and \(f_{te}^{i}\) using \(W_{i}\);
  construct aligned subspace \(X_{sa}^{tr} \) and \(X_{sa}^{te} \);
  train LDA using \(X_{sa}^{tr} \);
  apply LDA on \(X_{sa}^{te}\) to get prediction_i;
initialize pseudoLabel_i as an zero array;
for \(k = 1 : n \) and \( k \neq j \) and \( k \neq i \) do
  calculate features \(f_{tr}^{k}\) and \(f_{te}^{k}\) using \(W_{i}\);
  construct aligned subspace \(X_{sa}^{tr} \) and \(X_{sa}^{te} \);
  train LDA using \(X_{sa}^{tr} \);
  apply LDA on \(X_{sa}^{te}\) to get prediction_k;
  pseudoLabel_i += prediction_k;
end
pseudoLabel_i = sign(pseudoLabel_i); weight = 0.5 – ((sum(|pseudoLabel_i - prediction_i|)/number of trials))/2; ensemble_pred += weight * prediction_i;
end
labels = sign(ensemble_pred);
return labels

Algorithm 3: Combine_SA

Input: \(n\) training dataset \(\{X_{tr}^{1} \ldots X_{tr}^{n}\}\) and the jth test dataset \(X_{te}^{j}\)
Output: prediction of \(X_{te}^{j}\)’s label
initialize \(X_{tr}^{\text{n combine}_1}\) and \(X_{tr}^{\text{n combine}_2}\) as empty cells;
for \(i = 1 : n \) and \( i \neq j \) do
  load training data \(X_{tr}^{1}\) for class 1 and \(X_{tr}^{1}\) for class 2;
  append \(X_{tr}^{1}\) into \(X_{tr}^{\text{combine}_1}\) and append \(X_{tr}^{2}\) into \(X_{tr}^{\text{combine}_2}\);
end
initialize \(\Sigma_1, \Sigma_2\) as the average covariance matrix of \(X_{tr}^{\text{combine}_1}\) and \(X_{tr}^{\text{combine}_2}\);
\(W = \text{CSP}(\Sigma_1, \Sigma_2)\);
calculate features \(f_{tr}\) and \(f_{te}\) using \(W\);
construct aligned subspace \(X_{sa}^{tr}\) and \(X_{sa}^{te}\);
train LDA using \(X_{sa}^{tr}\);
apply LDA on \(X_{sa}^{te}\) to get prediction;
return prediction;

IV. RESULTS

We tested the prediction accuracy of different methods on the three datasets. The accuracy results on dataset IVa from BCI competition III, dataset IIa from BCI competition IV, and dataset Three are presented in Table I, Table II, and Table III respectively. In Table I and Table II, the proposed SA_weighted algorithm outperforms the other unsupervised methods, while it is close to combine_SA in Table II. In Table III, combine_SA outperforms the rest of the algorithms, but the accuracy of SA_weighted is comparable or better among all methods.

The weightings found by our algorithm are shown in Fig. 3, Fig. 4 and Fig. 5. While generally positive, there are some notable negative values which, as mentioned earlier, are possibly due to SA being an unsupervised algorithm.

V. DISCUSSION

We have presented a method for unsupervised transfer by weighting predictions from different training subjects after subspace alignment to the test subject. This method performs generally well across different datasets and in several cases... 

noSA) are ablation analyses included separately for compari- son to the full algorithm, i.e., SA_weighted.

We also include another method (called combine) which involves concatenating the training data from all training subjects and learning one set of CSP filters and LDA classifier for all the training data. The CSP filters and LDA classifier is then applied to the test data from the test subject. This method serves as a baseline for unsupervised transfer learning.

The combine_SA is another baseline algorithm that we used to compare the results with our proposed algorithm. combine_SA is identical to the combine algorithm except that SA is applied to the test data to align it with the combined training data. The pseudo-code for this algorithm is depicted in Algorithm 3.
outperforms the supervised performance obtained by training only on each subject’s own training data. Dataset IVa from BCI competition III has different numbers of training trials for each subject, and our method outperforms other unsupervised transfer algorithms presented in Table 1. The bias caused by different training set sizes has a strong influence on methods like combine_SA while it inflicts little influence on our method as we are weighting the predicted labels created by each training model based on their agreement with other subjects.

For Dataset Ila from BCI competition IV and Dataset Three, each subject has the same number of training trials, and our method performs comparably to others. In Dataset Three, combine_SA outperforms our method, and we found that the weightings for each training subject in Dataset Three have less variability (except for s6, s10, and s12, which have poor performance on supervised CSP) compared to those in the other two datasets. The less variability in weightings causes the method to perform more like the unweighted version. Even though it does not improve the overall accuracy, the lower weightings on the less reliable training subjects is a valuable feature and improves performance on some subjects.

REFERENCES


Fig. 3. Weightings in SA_weighted on Dataset IVa from BCI competition III

Fig. 4. Weightings in SA_weighted on Dataset IIa from BCI competition IV

Fig. 5. Weightings in SA_weighted on Dataset Three