MSSAN: Multi-source Separation and Alignment Network for Emotion Classification

Jiaqi Zeng [1]

Abstract

Electroencephalography (EEG) analysis is a great tool in many applications. After feature extraction, we can use the data to do emotion classification. Nowadays, more modalities are utilized in order to improve the performance of emotion recognition, such as eye movement. However, the statistical distribution of the data varies across subjects which leads a poor performance of traditional machine learning methods. Transfer learning allows the distributions of domains to be different which can be used to improve the performance of emotion classification. Most of the transfer learning algorithms now focus on single source domain, but in the context of emotion classification, there is far more than one source domain. We mainly propose a novel model to separate and align different source domains. We also extend the model to segregate multi-modal statistics. The proposed method is evaluated on a subset of the SJTU Emotion EEG Dataset and compared against various traditional machine learning methods. The result shows that our method has a better performance on the classification accuracy. Also, further analysis on various comparison experiments shows the effectiveness of our model.

1. Introduction

A brain-computer interface (BCI) is a direct communication pathway between a brain and an external device. A very important use of BCI is emotion recognition, which means identifying individual's emotional state based on its physiological or non-physiological signals. The non-physiological signals for example, facial expression or body gestures are not accurate and sometimes may be false. Physiological signals, on the contrast, are more reliable. Brain waves generate electrical potential in the brain when neurons are active. This electrical potential can be measured outside the skull, which is nothing but EEG (electroencephalogram) (Wagh & Vasanth, 2019). The EGG signals vary with emotion changes, which means it is very useful for emotion recognition. Overall, the processing steps to guess the emotional state from EEG signal can be clarified in Figure 1.(Wagh & Vasanth, 2019)

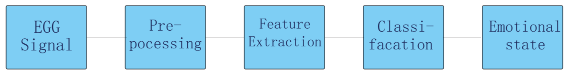


Figure 1. Processing steps for emotion recognition task.

There are many methods to extract features from EEG signals including Fourier transforms and related methods, wavelet transforms, principal component analysis (PCA), independent component analysis (ICA), autoregressive methods, which can significantly improve the performance of emotion classification. However, the statistical distribution of the data varies across subjects. This would limit the transfer ability of traditional machine learning methods which assume that the distributions of the labeled and unlabeled data are the same. In (Mohammadpour et al., 2017)'s experiment on different classification methods, including Neural Networks, SVM, KNN and Naïve Bayes with DWR feature, the classification accuracies of users' emotion only have a slight difference. Also, in (Alexander et al., 2019)'s survey on deep learning method, the accuracy differences between employed model based on DBN, CNN, and RNN is small, and it is likely there is no specific deep learning network structures suitable for emotion recognition task.

Transfer learning allows the domains, tasks, and distributions used in training and testing to be different. There is a priori that there is some structures shared by these different domains. What we do is to learn some representation of these structures and solve the task. In the context of emotion recognition, transfer learning is of great significance. So far, most of the transfer learning algorithm focus on single source domain adaptation. In emotion recognition task, we have the data of many subjects, which means that there is far more than one source domain can be used. Thus, we

[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. Correspondence to: Jiaqi Zeng <Gabyyyyyy@sjtu.edu.cn>.

propose a multi-source transfer learning model which can efficiently use the data from different source domains. The shared features and domain-specific features of different source domains would be both considered. Extensive experiments show that our method can remarkably improve the accuracy of the classification. Also, in this paper, we lead some comparison experiments to verify our theory and the results demonstrate that the proposed method has good explanation.

2. Related work

Transfer learning in BCI is an important research area. In this section, we will introduce the related work in four aspects: (1) Instance based Transfer Learning (2) Feature based Transfer Learning (3) Model based Transfer Learning (4) Deep Transfer Learning.

Instance based Transfer Learning is to re-weight some labeled data in the source domain for use in the target domain (Pan & Yang, 2010). For example, (Hou et al., 2017) developed sensory and motor maps based on the extracted features from recorded EEG signal of the volunteers which improve the effectiveness in designing BCI systems. (Li et al., 2010) used covariate shift adaptation which can adapt to the testing sessions without the need for labeling the testing session data. (Bhattacharyya) proposed a model which yields a weight matrix by maximizing source-trained target EEG classification performance and addresses the issue of inter-subject EEG classification. As the features of the same class from a subject has less variance, the feature vectors tend to form a bounded entity in the feature space which are partially or fully non-overlapping. They transform the features and to facilitate classification, they increase the overlap and use only a part of the target domain data during training.

Feature based Transfer Learning is to find a "good" feature representation that reduces difference between the source and the target domains and the error of classification and regression models (Pan & Yang, 2010). For instance, (Samek et al., 2013) presented a method that transfers non-stationary information between subjects, and thus effectively bridges the gap between training and test data. However, they mainly focus on methods of using information from other subjects and did not study the relations between within-session and between-session nonstationarities. There are some multi-subject methods that tackle the nonstationarity problem. (Devlaminck et al., 2011) presented a multisubject extension to the basic CSP algorithm in order to reduce the number of training trials and to improve performance by learning spatial filters across subjects. The main downside of this method is that it takes much time to compute cross-validation to select good parameter values, making the methods impractical.

Model based Transfer Learning is to discover shared parameters or priors between the source domain and target domain models (Pan & Yang, 2010). (Jayaram et al., 2016) presented a method for learning that transfers knowledge from previous subjects to new ones in any desired spatiotemporal feature space, able to work both on its own and on top of other paradigms. The proposed methods can better deal with both session-to-session and subject-to-subject variability as compared to simple pooling, achieving accuracies comparable to or better than single-session training with far fewer training trials.

Deep transfer learning studies how to utilize knowledge from other fields by deep neural networks and it has recently received increasing attention from researchers and has been successfully applied to numerous real-world applications. Deep learning algorithms attempt to learn high-level features from mass data, which make deep learning beyond traditional machine learning. It can automatically extract data features by unsupervised or semi-supervised feature learning algorithm and hierarchical feature extraction. In contrast, traditional machine learning methods need to design features manually that seriously increases the burden on users. (Hajinoroozi et al., 2017) studied the transferability of STCNN layers in performing for cross-subject and cross-experiment classification using EEG data. For cross-subject prediction, the convolutional layers capture more subject-specific features, whereas for cross-experiment prediction, the convolutional layers capture more general features across experiment. For EEG based BCI classification, they show that fine-tuning can improve the baseline performance, which suggests that transfer learning with STCNN has the ability to transfer general features from source domain to improve the performance in the target domain for EEG based classification.

We found that most transfer learning models in BCI focus on traditional transfer learning, while most of the deep transfer learning methods focus on image classification and they use single-source unsupervised domain adaptation such as (Long et al., 2015). There are only a few methods focus on multi-source domain adaptation, for example, (Zhu et al., 2019) proposed a new framework with two alignment stages which not only respectively aligns the distributions of each pair of source and target domains in multiple specific feature spaces, but also aligns the outputs of classifiers by utilizing the domain specific decision boundaries. Therefore, we proposed a new method based on multi-source domain adaptation to solve the task of emotion recognition.

3. Method

For transfer learning, the distributions of source data and target data are different, which makes it challenging using normal machine learning techniques. First, we introduce the problem of multi-source unsupervised domain adaptation. There are N labeled source domain data $\{X_{si}, Y_{si}\}_{i=1}^{N}$,

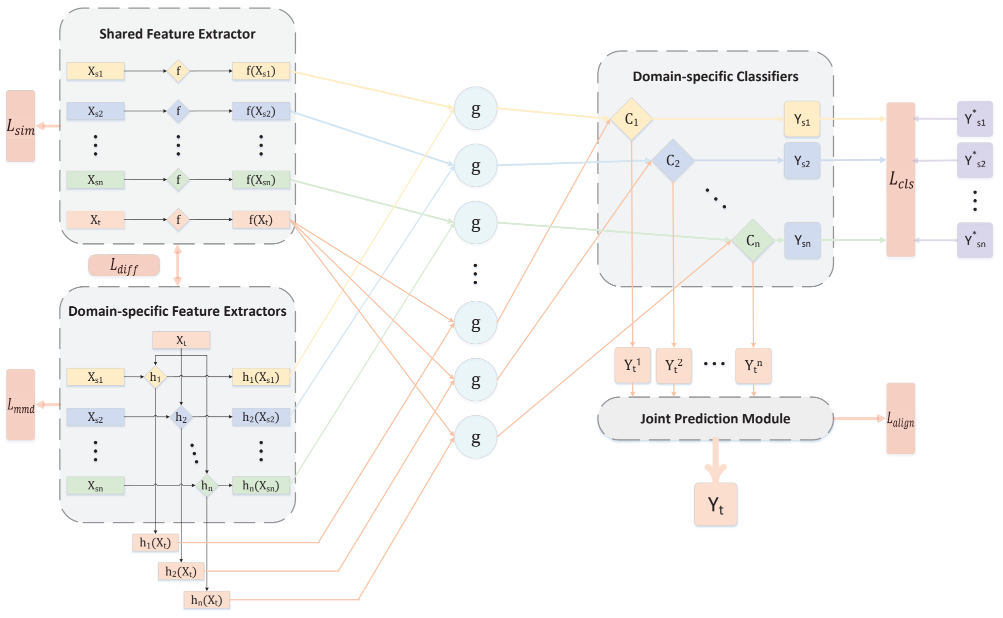


Figure 2. Framework of the proposed Multi-source Separation Alignment Network. The model mainly includes three subnetworks: shared feature extractor, domain-specific extractors, and domain-specific classifiers. Shared features and domain-specific features are combined by the function g before classification. Final prediction for target domain data is generated by the joint prediction module. The loss function consists of similarity loss, difference loss, mmd loss, classification loss, and alignment loss.

which can be regarded as samples drawn from N source domain distributions $\{p_{si}(x,y)\}_{i=1}^{N}$. Additionally, there is a set of target domain data X_t which is sampled from a target domain distribution $p_t(x,y)$ without labels. In our task, the inputs are $\{X_{si}, Y_{si}\}_{i=1}^{N}$ and X_t. We aim to fit the target domain distribution and predict the labels Y_t of target domain data X_t.

Recently, in the vision domain, (Zhao et al., 2018) and (Xu et al., 2018) proposed to minimize the classification loss together with a distance loss between each pari of source and target domains to learn common domain-invariant features for all domains. The formulation can be:

$$min_{F,C} \sum_{j=1}^{N} \mathbf{E}_{x \in X_{sj}} J(C(F(x_i^{sj})), y_i^{sj}) + \lambda \sum_{j=1}^{N} D(F(x_{sj}), F(X_t)),$$

where $J(\cdot,\cdot)$ denotes the classification loss and $D(\cdot,\cdot)$ denotes an estimate of the discrepancy between two domains, which can be CORAL (Sun & Saenko, 2016), MMD (Gretton et al., 2012), Confusion loss (Tzeng et al., 2015) or Reconstruction loss (Bousmalis et al., 2016). $F(\cdot)$ is the common feature extractor to map all domains into a shared feature space and $C(\cdot)$ is the classifier. Besides, (Zhu et al., 2019) improved the above methods by learning domain-specific feature extractors and classifiers. It can be formulated as:

$$min_{\{F\},\{C\}} \sum_{j=1}^{N} \mathbf{E}_{x \in X_{sj}} J(C_{sj}(F_{sj}(x_i^{sj})), y_i^{sj}) + \lambda \sum_{j=1}^{N} D(F_{sj}(x_{sj}), F_{sj}(X_t)).$$

The problem of (Zhao et al., 2018) and (Xu et al., 2018) is that they only consider domain-invariant representations but ignore domain-specific features. On the contrary, the problem of (Zhu et al., 2019) is that they only consider domain-specific representations but ignore domain-invariant features. In fact, (Bousmalis et al., 2016) has shown that shared features and domain-specific features are both of great importance. Therefore, we propose a framework to simultaneously capture these two kinds of features. Our framework is shown as Figure 2. In the following parts, we will introduce our model in details.

3.1. Shared Feature Extractor

We claim that there should exist some shared features among various source domains and target domains. In order to help transfer knowledge from source domains to target domains, we propose a shared encoder $f(\cdot)$, which aims to learn the domain-invariant features. This subnetwork is shared by all domains. To encourage that the original inputs are mapped to a common feature space, we further propose a similarity loss to constrain the network. The loss can be

$$L_{sim} = ||f(X_{sj}) - f(X_t)||_{l_1/l_2}.$$

3.2. Domain-specific Feature Extractor

Transfer learning task has a notable property that the data distributions of training set and testing set are not exactly matched. In this subnetwork, we aim to extract unique features for data from each domain. Therefore, we build N feature extractors $h_j(\cdot)$ and map each pair of source domain data and target domain data into a specific feature space. Given a batch of data $\{x^{sj}\}$ from the j^{th} source domain and $\{x^t\}$ from the target domain, we feed them into the j^{th} domain-specific feature extractor and obtain $\{h_j(x^{sj})\}$ and $\{h_j(x^t)\}$. These N extractors work in parallel and do not share the same weights.

In order to map each pair of source domain data and target domain data into a specific feature space, we need to compare the distribution of $\{h_j(x^{sj})\}$ and $\{h_j(x^t)\}$. In domain adaptation field, there are several methods proposed to achieve this, such as adversarial loss (Tzeng et al., 2015), CORAL loss (Sun & Saenko, 2016) and MMD loss (Gretton et al., 2012). In our model, we choose MMD loss as the estimation to reduce the discrepancy between distributions of source domain and target domain inside a pair.

The Maximum Mean Discrepancy (MMD) is a kernel two-sample test which rejects or accepts the null hypothesis $p = q$ based on the observed samples. When calculating MMD between x_i and x_j, we are supposed to map them into the reproducing kernel Hillbert space (RKHS) endowed with a characteristic kernel k by a mapping function $\phi(\cdot)$. Formally, MMD defines the following difference measure:

$$D_H(p,q) \triangleq ||\mathbf{E}_p[\phi(\mathbf{x}^s)] - \mathbf{E}_q[\phi(\mathbf{x}^t)]||^2_H,$$

where H is the RKHS. The kernel k means $k(\mathbf{x}^s, \mathbf{x}^t) = \phi(\mathbf{x}^s), \phi(\mathbf{x}^t) >$, where $< \cdot, \cdot >$ denotes inner product of vectors. (Gretton et al., 2012) showed that $p = q$ if and only if $D_H(p,q) = 0$. In practice, as estimate of the MMD compares the square distance between the empirical kernel mean embeddings as

$$\hat{D}_H(p,q) = ||\frac{1}{n_s}\sum_{x_i \in D_s}\phi(\mathbf{x}_i) - \frac{1}{n_t}\sum_{x_j \in D_t}\phi(\mathbf{x}_j)||^2_H,$$

where $\hat{D}_H(p,q)$ is an unbiased estimator of $D_H(p,q)$, D_s means the source domain, and D_t means the target domain. In this way, the MMD loss is reformulated as:

$$L_{mmd} = \frac{1}{N}\sum_{j=1}^{N}\hat{D}_H(h_j(X_{sj}), h_j(X_t)).$$

We minimize the reofrmulated MMD loss to obtain domain-invariant representations for each pair of source and target domain.

Furthermore, to ensure that the domain-specific feature extractor and the shared feature extractor can encode different features of $\mathbf{x}$, we propose a difference loss as

$$L_{diff} = \frac{1}{N}\sum_{j=1}^{N} l_{diff}(h_j(\mathbf{x_{sj}}), f(\mathbf{x_{sj}}))$$

Therefore, through the domain-specific feature extractors, we map each pair of source domain and target domain into a specific space, in which the distributions of features of two domains are similar but they are different from the features encoded by the shared feature extractor.

3.3. Domain-specific Classifier

Before classification, we combine the shared features and domain-specific features as following:

$$g(x_{sj}) = g(h_j(\mathbf{x_{sj}}), f(\mathbf{x_{sj}})),$$

$$g(x_t) = g(h_j(\mathbf{x_t}), f(\mathbf{x_t})).$$

In our experiment, we set function $g(\cdot,\cdot)$ as weighted sum or concatenation. For weighted sum, we can formulate it as

$$g(x) = w_1 h_j(x) + w_2 f(x),$$

where w_1 and w_2 are learnable parameters satisfying $0 < w_1 < 1$,$0 < w_2 < 1$, and $w_1 + w_2 = 1$.

We build N domain-specific classifiers C_j for different domains and calculate the cross entropy loss for source domain data classification as

$$L_{cls} = \frac{1}{N}\sum_{j=1}^{N} Cross\ Entropy(Y_{sj}, softmax(C_j(g(X_{sj})))).$$

Besides, we feed the target domain data into all N domain-specific classifiers and for each target domain sample, we generate a set of corresponding predicted classification results $\{C_j(x_t^C)\}_{j=1}^N$. The classifiers are trained on different source domains, hence they might have disagreements on the prediction for target samples especially the target samples near the classification boundaries. Voting might be a possible solution, but it sometimes has poor performance. Intuitively, the same target sample predicted by different classifiers should obtain the same prediction. Therefore, we employ an alignment operation to minimize the discrepancy among predictions of all classifiers. We add an alignment loss to help train the model, through which the domain-specific classifiers can learn to output similar predictions for the same target sample. The prediction after softmax function can be viewed as a distribution, hence we can utilize either Jensen-Shannon Divergence or l_1/l_2 distance as the distance function. In this way, we formulate the alignment loss as:

$$\begin{aligned} L_{align} = & \frac{2}{N \times (N+1)} \sum_{j=1}^{N-1} \sum_{i=j+1}^{N} \\ & \mathbf{E}_{x \sim X_t} [d(C_i(g(x_k)), C_j(g(x_k)))]. \end{aligned}$$

By minimizing the alignment loss, the outputs of all classifiers are similar. We finally predict the labels of target samples through a joint predicting function p, which will be introduced in subsection 3.5.

3.4. Multi-source Separation and Alignment Network

Conclusively, we summarize our framework for multi-source transfer learning as Multi-source Separation and Alignment Network (MSSAN). Our framework is mainly composed of four parts: shared feature extraction, domain-specific feature extraction, domain-specifc classification, and joint prediction. Overall, the loss of our method consists of five parts: similarity loss, MMD loss, difference loss, classification loss, and alignment loss. For details, by minimizing similarity loss, the shared feature extractor encodes shared features of different domains; by minimizing MMD loss, the model maps each pair of source and target domain into a specific feature space; by minimizing difference loss, the network encourages the difference between domain-specific features and shared features; by minimizing classification loss, the network could accurately classify the source domain data; by minimizing alignment loss, the discrepancy of outputs between classifiers are reduced which makes it easier to predict. The total loss is formulated as:

$$L_{total} = L_{cls} + \gamma(aL_{sim} + bL_{diff} + cL_{mmd} + dL_{align}).$$

Here, a, b, c, d are fixed hyper-parameters while γ is a changeable hyper-parameter. Following (Zhu et al., 2019), when training the model, the optimization direction should first consider more about classification loss and then consider more about other losses. A possible explanation is if we consider other loss such as similarity loss and difference loss primarily in the first place, the parameters might be trapped in some local minimums which is unsuitable for classification while our main task is classification. We set γ for the i^{th} iteration as

$$\gamma = \frac{2}{1 + e^{\frac{-10*i}{\#iteration}}} - 1,$$

where $\#iteration$ denotes the total number of iteration for training.

Since we conduct our experiment on EEG data where each sample is a 310-dimension vector, we set the feature extractors and classifiers simply as MLPs. The activation function is set as LeakyReLU. The training mainly follows standard mini-batch stochastic gradient descent (SGD) algorithm. The whole procedure is summarized in Algorithm 1.

Algorithm 1 Multi-source Separation and Alignment Network

```
Input: source domain data $\{X_{sj}, Y_{sj}\}_{j=1}^{N}$, target domain data $X_t$, # training iterations $T$.
Output: predicted labels for target domain data $\hat{Y}_t$
Initialize all network parameters.
for $t = 1$ to $T$ do
  for $j = 1$ to $N$ do
    Randomly sample $m$ images $\{x_{sj}^i, y_{sj}^i\}_{i=1}^{m}$ from the $j^{th}$ source domains.
    Randomly sample $m$ images $\{x_t^i\}_{i=1}^{m}$ from target domain.
    Feed source and target samples to shared feature extractor to get the shared features $f(x_{sj}^i)$ and $f(x_t^i)$.
    Calculate similarity loss using $f(x_{sj}^i)$ and $f(x_t^i)$ where $i = 1, 2, \ldots, m$.
    Feed source and target samples to the corresponding domain-specific feature extractor to get domain-specific features $h_j(x_{sj}^i)$ and $h_j(x_t^i)$.
    Calculate difference loss using $h_j(x_{sj}^i)$ and $f(x_{sj}^i)$ where $i = 1, 2, \ldots, m$.
    Calculate mmd loss using $h_j(x_{sj}^i)$ and $h_j(x_t^i)$ where $i = 1, 2, \ldots, m$.
    Combine shared features and domain-specific features to get $g(x_{sj}^i)$ and $g(x_t^i)$.
    Feed $g(x_{sj}^i)$ and $g(x_t^i)$ to domain-specific classifiers to get classification results $C_j(g(x_{sj}^i))$ and $C_j(g(x_t^i))$.
    Calculate classification loss using $C_j(g(x_{sj}^i))$ and $y_{sj}^i$ where $i = 1, 2, \ldots, m$.
  end for
  Calculate alignment loss using $\{C_j(g(x_t^i))\}_{j=1}^{N}$ where $i = 1, 2, \ldots, m$.
  Calculate the total loss and update the network parameters using gradient descent.
end for
Get the prediction $p(\{C_j(g(X_t))\}_{j=1}^{N})$.
```

3.5. Joint Prediction Module

For a target domain sample, obtained $\{C_j(g(x_t))\}_{j=1}^N\}$ from N domain-specific classifiers, we want to predict the final label. A simple method is to average the outputs of N classifiers. This can be formulated as

$$p(\{C_j(g(x_t))\}_{j=1}^N\}) = \frac{1}{N}\sum_{j=1}^{N} softmax(C_j(g(x_t))).$$

Intuitively, data from different source domains may contribute differently to the prediction on target domain. If a source domain is relatively similar to the target domain, it may contribute more. Otherwise, it may contribute less. The simple average method cannot capture this property. Hence, we want to predict by a weighted sum of outputs from N domains. We formulate this as

$$p(\{C_j(g(x_t))\}_{j=1}^N\}) = \frac{1}{N}\sum_{j=1}^{N} \mathbf{w}_j softmax(C_j(g(x_t))),$$

where $\mathbf{w}_j$ is the j^{th} dimension of a weighting vector $\mathbf{w}$. To measure the similarity between a source domain and the target domain, we propose to loss-based method. We calculate the weight of a source domain based on its classification loss and the mmd loss between it and the target domain. If the mmd loss is relatively low, this pair of domains may have been mapped into a shared feature space. Meanwhile, if the classification loss is low, it means features from this feature space can be utilized to predict effectively. In this way, this source domain is supposed to contribute more to the prediction. Hence, w can be formulated as

$$\mathbf{w} = softmax(\frac{\eta}{\alpha\mathbf{L_{mmd}} + \beta\mathbf{L_{cls}}}),$$

where $\mathbf{L_{mmd}}$ and $\mathbf{L_{cls}}$ are N dimensional vectors, α, β, η are hyper-parameters. Notice that the computation cost will be high if we calculate these loss again during inference process. A possible solution is to save the loss during the training process.

3.6. Generative Encoding for Multi-Modality Emotion Recognition

Inspired by (Du et al., 2017), we further extend our method for multi-modality inputs. The challenge here is how to combine features from different modality. Assume we are faced with multi-view data that appears as pairs $(X, y) = (\{x^{(v)}\}_{v=1}^V, y)$, with observation $x^{(v)}$ from the v-th view and the corresponding class label y. We adopt a multi-modality variational autoencoder (MVAE) which is developed from the basic VAE. The architecture of MVAE is shown in Figure 3. We use latent variable z to generalize multi-view features $\{x^{(v)}\}_{v=1}^V$.

Typically, both the prior $p(z)$ and the approximate posterior $q_\phi(z|X)$ are assumed to be Gaussian distributions. To learn more powerful and expressive models – in particular, models with multi-modal latent variable structures for multi-modal emotion recognition applications – we seek a mixture of Gaussians for $q_\phi(z|X)$, while preserving $p(z)$ as a standard Gaussian.

$$p(z) = \mathbf{N}(z|\mathbf{0}, \mathbf{I})$$

$$q_\phi(z|X) = \sum_{v=1}^{V} \lambda^{(v)}\mathbf{N}(z|\mu_{\phi^{(v)}}(x^{(v)}), \sum_{\phi^{(v)}}(x^{(v)})),$$

where the mean $\mu_{\phi^{(v)}}$ and the covariance $\sum_{\phi^{(v)}}$ are non-linear functions of the observation $x^{(v)}$, with variational parameter $\phi^{(v)}$. $^{(v)}$ is the non-negative normalized weight factor for the v-th view, i.e. $\lambda^{(v)} > 0$ and $\sum_{v=1}^{V} \lambda^{(v)} = 1$.

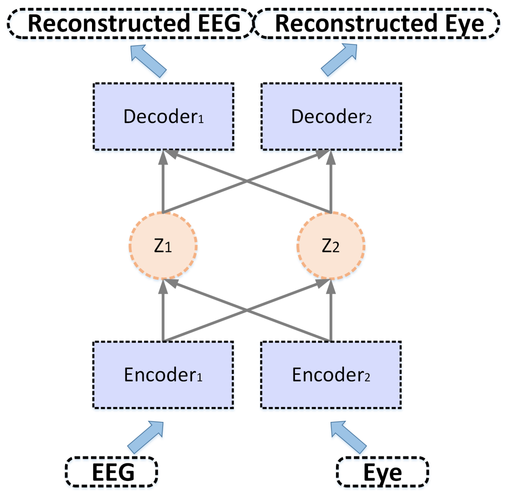


Figure 3. The architecture of MVAE

When decoding, we utilize z as input and generate reconstructed multi-modality data.

$$p_{\theta^{(v)}}(x^{(v)}|z) = f(x^{(v)}; z, \theta^{(v)})$$

The MVAE is pre-trained before conducting MSSAN. After training, we utilize it as a feature extractor and obtain z as a representation of multi-modality feature which is then fed into the MSSAN.

4. Experiment for Single-Modality Emotion Recognition

We evaluate the Multi-source Separation and Alignment Neural Network on an Electroencephalogram(EEG) dataset. We compare the accuracy of our model with some baseline models and start-of-the-art domain adaptation methods. Our model outperforms the above methods in most cases. The mean value and standard deviation of accuracies in 3 target domains achieved by our model outperform all compared deep domain adaptation methods.

4.1. Data Preparation

We solve a four classification problem based on EEG emotion recognition. The dataset used is a subset of the SJTU Emotion EEG Dataset (SEED-4) of the Brain Computing and Machine Intelligence research center (BCMI). To collect the data, 24 movie segments of 4 kinds of emotions (happiness, sadness, neutrality, and fear) are selected as the stimulus materials. Each video lasts about three minutes, and each emotion are included in six movie segments. The dataset includes data of 13 people, in which 10 are set as source domains and the other 3 are set as target domains. The EEG features are differential entropy (DE) features with 62 leads. The feature extraction time window is 4 seconds and the feature dimension is 310.

4.2. Baselines and Implementation Details

Though many machine learning methods existed, not all of them are suitable for the transfer learning task since the data distributions of source domains and target domains are different. To exhibit this, we employ SVM as a baseline. We utilize the one versus one and one versus rest strategies to solve the four-classification problem and achieve a better accuracy with one versus rest. We choose RBF kernel when implementing SVMs. Furthermore, we try to reduce noises by dimension compression before feeding into the SVMs. We train an AutoEncoder with both source domain data and target domain data to implement dimension reduction.

There is a small amount of multi-source domain adaptation work on real-world benchmarks. In our experiment, we introduce a recently proposed state-of-the-art method Multi Feature Space Adaptation Network (MFSAN) (Zhu et al., 2019) as a multi-source baseline. We also choose a state-of-the-art single source domain adaptation method Deep Adaptation Network (DAN) (Long et al., 2015) as a single-source baseline. When evaluating the single-source method, we set data of the 10 people together as the source domain. Since this two method are originally utilized for vision domain, feature extractors are implemented by ResNets. For the EEG Emotion Classification task, we replace them with multi-layer perceptrons.

We use mini-batch stochastic gradient descent (SGD) with momentum of 0.9. The batch size is set as 64. Follow (Ganin & Lempitsky, 2014), we use the learning rate annealing strategy: the learning rate is not selected by a grid search to high computationcost, it is adjusted during SGD using the following formula: $\eta_p = \frac{\eta_0}{(1+\alpha p)^\beta}$, where p is the training progress linearly changing from 0 to 1, $\eta_0 = 0.001$, $\alpha = 10$ and $\beta = 0.75$, which is optimized to promote convergence and low error on the source domain.

4.3. Results

We compare our MSSAN model with baselines on the EEG Emotion Classification dataset. The results of different methods on different target domains are shown in Figure 4.

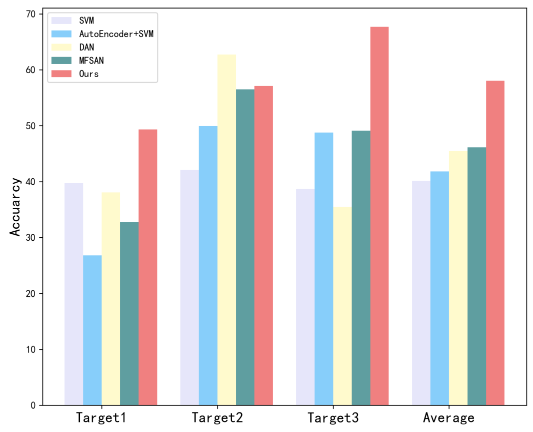


Figure 4. Classification accuracies on different target domains.

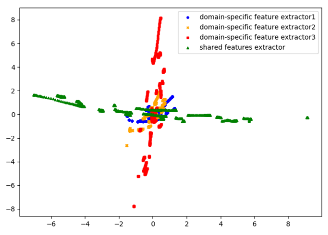
Also, we list the mean value and standard deviation of performance on different targets in Table 1. From these results, we have the following insightful observations:

1. We can see that our model outperforms all compared baseline and state-of-the-art methods in most cases. The encouraging results indicate that it is important to separate the shared features and domain-specific features and learn multiple domain-specific classifiers for each pair of source and target domains.

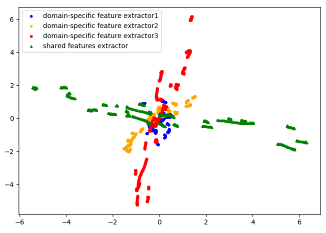
2. The standard deviation of our model is the lowest among three deep learning methods, which indicates the alignment operation may help increase the stability of the performance.

3. Multi-source domain adaptation methods (MFSAN and Ours) outperform single-source domain adaptation method (DAN) in most cases, which shows that moderate amount of source domains contribute to the prediction of the data distribution of target domains.

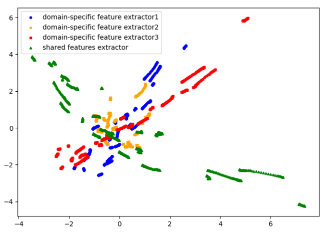
4. Specially designed methods for transfer learning is necessary. Domain adaptation methods, no matter single-source or multi-source, significantly outperform the normal SVM method.



(a) Target domain 1

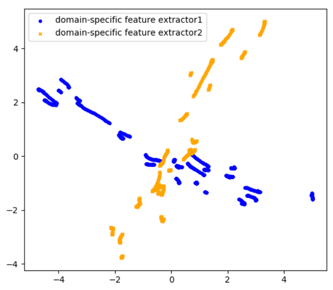


(b) Target domain 2

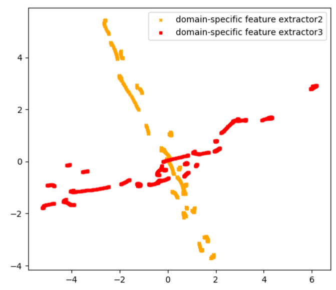


(c) Target domain 3

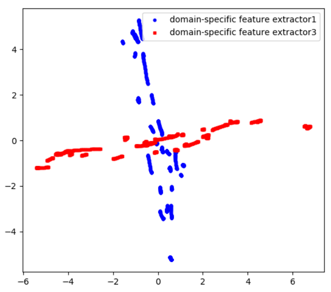
Figure 5. Visualization of features extracted from each feature extractors.



(a)



(b)



(c)

Figure 6. Visualization of features of target domain 2 extracted by each two domain-specific feature extractors.

5. Dimension reduction may improve the performance of methods not designed for transfer learning. However, in our experiment, it is useless for deep domain adaptaion methods.

Table 1. The mean value and standard deviation of classification accuracies with different methods.

STATS.	SVM	AE+SVM	DAN	MFSAN	OURS
MEAN	40.15	41.83	45.43	46.14	58.04
STD.	1.75	13.04	15.05	12.15	9.21

4.4. Further Analysis

In this subsection, we will further analyze the performance of our model from various perspectives. We will show the influence of different settings, the features extracted by different feature extractors, and evaluate the effectiveness of our model.

4.4.1. ANALYSIS ON COMBINATION FUNCTION g

Combination function g is a critical components of our MSSAN model. To find out a better function g, we compare weighted sum and concatenation with the prediction function fixed as average and the results are shown in Table 2. From the results we conclude that the mean value of concatenation method is higher than that of weighted sum, while the standard deviation performs inversely. An possible explanation is that utilizing learnable weights to sum up shared features and domain-specific features may obtain a neutralized feature, in which outstanding and terrible features are rare and moderate features are predominant. On the contrary, direct concatenation keeps the outstanding and terrible features still, which might lead to a relatively major fluctuations on the accuracy.

Table 2. Comparison between performance of different combination function

g	T1	T2	T3	MEAN	STD.
WEIGHTED SUM	51.59	50.41	47.83	49.94	1.92
CONCATENATION	49.35	57.10	67.69	58.04	9.21

4.4.2. Analysis on Joint Prediction Module

With a set of predicted probability distributions from different domain-specific classifiers, the usage of joint prediction module is essential in order to determine the final prediction. In the joint prediction module, two methods are employed in the experiment, including simple average and the loss-based weighted sum. We fix the combination function g as concatenation and get the results as shown in Table 3. We find that on target domain 1 and 2, loss-based method slightly outperform average method. The mean value of average method is slightly higher because of its outstanding performance on target domain 3. The standard deviation of loss-based method is smaller which indicates that this method might have an effect of neutralization.

Table 3. Comparison between performance of different prediction function p

p	T1	T2	T3	Mean	Std.
Loss-based	49.59	57.46	61.70	56.25	6.15
Average	49.35	57.10	67.69	58.04	9.21

4.4.3. Visualization on features extracted by different extractors

To validate the shared feature extractor and domain-specific feature extractors, two significant subnetworks in our model, we try to visualize the features extracted by these extractors. We set the number of source domains as three and employ Principal Component Analysis (PCA) to reduce the dimension of features extracted by these extractors to two. PCA is capable to maintain the features which contribute most to the variance of the data. This is achieved by retaining the low-order principal components and ignoring the high-order principal components. In this way, the most important aspects of the data can be retained by the low-order components. The results are shown in Figure 5. Figure 5(a) and Figure 5(b) are similar, in which we can easily see that the distributions of features extracted by different encoders are apparently different. Figure 5(c) is relatively unclear but we can still recognize the variance of different feature distributions. These indicate the effectiveness of the extractors.

From all 3 figures, it is clear that the distribution of shared feature is nearly orthogonal to all the distributions of domain-specific features. Since we set an difference loss to ensure the model to extract various features, this results indicates that the difference loss might be effective.

Because we reduce the dimension to two, it cannot show more than two orthogonal relations. Furthermore, we explore the relations between distributions of various domain-specific features. We choose a target domain and visualize each two distributions of domain-specific features. As shown in Figure 6, we easily see that these three distributions are orthogonal to each other. This shows that although we do not add a loss between domain-specific features, the model indeed learn the unique features of each pair of source domain and target domain. This demonstrate the validation of the design of shared feature extractor and domain-specific feature extractors from another respect.

4.4.4. Analysis on Dimension of Features

When setting the parameters of the network, dimension of features including shared features and domain-specific features are essential. Based on the previous experiment, we fix the number of layers of shared feature extractor as 1 and the number of layers of domain-specific feature extractors as 4. Because we need to calculate the difference loss, the dimension of shared features and domain-specific features are required to be same. In our experiment, we set the dimension as 64, 128, 256, 310, 512, and 1024. The results are shown in Figure 7.

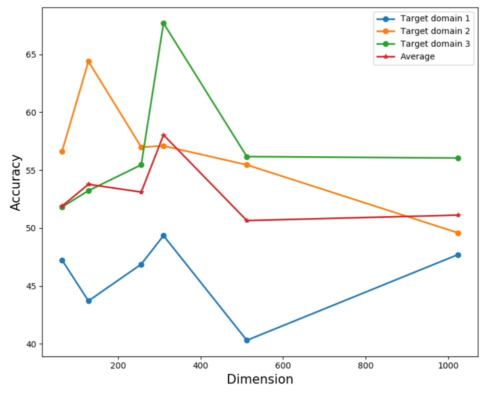


Figure 7. Comparison among performance with different dimensions of features.

From the results we can see that keep the original dimension of 310 performs best in most cases. Reducing dimension or increasing dimension of features may not help capture meaningful information.

5. Conclusion and Future Work

In this work, we focus on multi-source emotion classification. We formulate five kinds of loss for transfer learning method. To improve the classification accuracy, we then propose MSSAN, which considers all these losses and both shared features and domain-specific features. Experiments show that MSSAN outperforms baseline models significantly. In the future, we will conduct more experiments focusing on multi-modality data to verify the validity of our extension.

References

Alexander, Craik, Yongtian, He, Jose, L, and Contreras-Vidal. Deep learning for electroencephalogram (eeg) classification tasks: a review. *Journal of Neural Engineering*, 2019.

Bhattacharyya, S. A many objective optimization approach for transfer learning in eeg classification.

Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., and Erhan, D. Domain separation networks. In *Advances in neural information processing systems*, pp. 343–351, 2016.

Devlaminck, D., Wyns, B., and .Grosse-Wentrup, M. Multisubject learning for common spatial patterns in motor-imagery bci. *Computational intelligence and neuroscience*, 6, 2011.

Du, C., Du, C., Li, J., Zheng, W.-l., Lu, B.-l., and He, H. Semi-supervised bayesian deep multi-modal emotion recognition. *arXiv preprint arXiv:1704.07548*, 2017.

Ganin, Y. and Lempitsky, V. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Hajinoroozi, M., Mao, Z., Lin, Y.-P., and Huang, Y. Deep transfer learning for cross-subject and cross-experiment prediction of image rapid serial visual presentation events from eeg data. In Schmorrow, D. D. and Fidopiastis, C. M. (eds.), *Augmented Cognition. Neurocognition and Machine Learning*, pp. 45–55, 2017.

Hou, J., Li, Y., Liu, H., and Wang, S. Improving the p300-based brain-computer interface with transfer learning. In *2017 8th International IEEE/EMBS Conference on Neural Engineering (NER)*, pp. 485–488, 2017.

Jayaram, V., Alamgir, M., Altun, Y., Scholkopf, B., and Grosse-Wentrup, M. Transfer learning in brain-computer interfaces. *IEEE Computational Intelligence Magazine*, 11(1):20–31, 2016.

Li, Y., Kambara, H., Koike, Y., and Sugiyama, M. Application of covariate shift adaptation techniques in brain–computer interfaces. *IEEE Transactions on Biomedical Engineering*, 57(6):1318–1324, 2010.

Long, M., Cao, Y., Wang, J., and Jordan, M. I. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

Mohammadpour, M., Hashemi, S. M. R., and Houshmand, N. Classification of eeg-based emotion for bci applications. In *2017 Artificial Intelligence and Robotics (IRANOPEN)*, pp. 127–131, 2017.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22 (10):1345–1359, 2010.

Samek, W., Meinecke, F. C., and Müller, K. Transferring subspaces between subjects in brain–computer interfacing. *IEEE Transactions on Biomedical Engineering*, 60 (8):2289–2298, 2013.

Sun, B. and Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pp. 443–450. Springer, 2016.

Tzeng, E., Hoffman, J., Darrell, T., and Saenko, K. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4068–4076, 2015.

Wagh, K. P. and Vasanth, K. Electroencephalograph (eeg) based emotion recognition system: A review. 2019.

Xu, R., Chen, Z., Zuo, W., Yan, J., and Lin, L. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3964–3973, 2018.

Zhao, H., Zhang, S., Wu, G., Gordon, G. J., et al. Multiple source domain adaptation with adversarial learning. 2018.

Zhu, Y., Zhuang, F., and Wang, D. Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5989–5996, 2019.