Transductive Transfer Machine

Nazli Farajidavar, Teofilo deCampos, Josef Kittler

CVSSP, Univeristy of Surrey, Guildford, Surrey, UK GU2 7XH

Abstract. We propose a pipeline for transductive transfer learning and demonstrate it in computer vision tasks. In pattern classification, methods for transductive transfer learning (also known as unsupervised domain adaptation) are designed to cope with cases in which one cannot assume that training and test sets are sampled from the same distribution, i.e., they are from different domains. However, some unlabelled samples that belong to the same domain as the test set (i.e. the target domain) are available, enabling the learner to adapt its parameters. We approach this problem by combining three methods that transform the feature space. The first finds a lower dimensional space that is shared between source and target domains. The second uses local transformations applied to each source sample to further increase the similarity between the marginal distributions of the datasets. The third applies one transformation per class label, aiming to increase the similarity between the posterior probability of samples in the source and target sets. We show that this combination leads to an improvement over the state-of-the-art in cross-domain image classification datasets, using raw images or basic features and a simple one-nearest-neighbour classifier.

1 Introduction

In many machine learning tasks, such as object classification, it is often not possible to guarantee that the data used to train a learner offers a good representation of the distribution of samples in the test set. Furthermore, it is often expensive to acquire vast amounts of labelled training samples in order to provide classifiers with a good coverage of the feature space. Transfer learning methods can offer low cost solutions to these problems, as they do not assume that training and test samples are drawn from the same distribution [1]. Such techniques are becoming more popular in Computer Vision, particularly after Torralba and Efros [2] discovered significant biases in object classification datasets. However, much of the work focuses on *inductive* transfer learning problems, which assume that labelled samples are available both in source and target domains. In this paper we focus on the case in which only unlabelled samples are available in the target domain. This is a *transductive* transfer learning (TTL) problem, i.e., the joint probability distribution of samples and classes in the source domain $P(\mathtt{X}^{src}, \mathtt{Y}^{src})$ is assumed to be different, but related to that of a target domain joint distribution $P(\mathtt{X}^{trg}, \mathtt{Y}^{trg})$, but labels $\mathtt{Y}^{trg}$ are not available in the target set. We follow a similar notation to that of [1] (see Table 1). Some papers in the literature refer to this problem as Unsupervised Domain Adaptation.

TTL methods can potentially improve a very wide range of classification tasks, as it is often the case that a domain change happens between training and application of algorithms, and it is also very common that unlabelled samples are available in the target domain. For example, in image classification, the training set may come from high quality images (e.g. from DSLR cameras) and the target test set may come from mobile devices. Another example is action classification where training samples are from tennis and test samples are from badminton. TTL methods can potentially generalise classification methods for a wide range of domains and make them scalable for big data problems.

In this paper, we propose Transductive Transfer Machine (TTM), a framework that combines methods that adapt the marginal and the conditional distribution of the samples, so that source and target datasets become more similar, facilitating classification. A key novelty is a sample-based adaptation method, TransGrad, which enables a fine adjustment of the probability density function of the source samples. Our method obtains state-of-the-art results in cross-domain vision datasets using a simple nearest neighbour classifier, with a significant gain in computational efficiency in comparison to related methods.

In [3], we present a follow-up work which adds a step that automatically selects the most appropriated classifier and its kernel parameter. The present paper gives more details of the derivations of the methods in the pipeline and includes further evaluations of its main steps.

In the next section, we briefly review related works and give an outline of our contribution. Section 3 presents the core components of our method and further discusses the relation between them and previous works. This is followed by a description of our framework and an analysis of our algorithm. Experiments and conclusions follow in sections 4 and 5.

2 Related work

According to Pan and Yang's taxonomy [1], Transfer Learning (TL) methods can be of the following types: *Inductive*, when labelled samples are available in both source and target domains, *Transductive*, when labels are only available in the source set and *Unsupervised*, when labelled data is not present. For the reasons highlighted in Section 1, we focus on Transductive TL problems (TTL). They relate to sample selection bias correction methods [4, 5], where training and test distributions follow different distributions but the label distributions remain the same. It is common to apply semi-supervised learning methods for transductive transfer learning tasks, e.g. Transductive SVM [6]. In [7], a domain adapted SVM was proposed, which simultaneously learns a decision boundary and maximises the margin in the presence of unlabelled patterns, without requiring density estimation. In contrast, Gopalan et al. [8], used a method based on Grassmann manifold in order to generate intermediate data representations to model cross-domain shifts. In [9], Chu et al. proposed to search for an instance based re-weighting matrix applied to the source samples. The weights are based on the similarity between the source and target distributions using the Kernel Mean

Matching algorithm. This method iteratively updates an SVM classifier using transformed source instances for training until convergence.

Transfer learning methods can be categorised based on *instance re-weighting* (e.g. [10, 9]), *feature space transformation* (e.g. [11, 12]) and *learning parameters transformation* (e.g. [7, 13]). Different types of methods can potentially be combined. In this paper, we focus on *feature space transformation* and approach the TTL problem by finding a set of transformations that are applied to the source domain samples $G(\mathtt{X}^{src})$ such that the joint distribution of the transformed source samples becomes more similar to that of the target samples, i.e. $P(G(\mathtt{X}^{src})), \mathtt{Y}^{src}) \approx P(\mathtt{X}^{trg}, \mathtt{Y}^{trg*})$, where $\mathtt{Y}^{trg*}$ are the labels estimated for target domain samples.

Long et al. [12] proposed a related method which does Joint Distribution Adaptation (JDA) by iteratively adapting both the marginal and conditional distributions using a procedure based on a modification of the Maximum Mean Discrepancy (MMD) algorithm [11]. JDA uses the pseudo target labels to define a shared subspace between the two domains. At each iteration, this method requires the construction and eigen decomposition of an $n \times n$ matrix whose complexity can be up to $O(n^3)$.

Our pipeline which first searches for a global transformation such that the marginal distribution of the two domains becomes more similar and then with the same objective applies a set of local transformations to each transformed source domain sample. Finally in an iterative scheme, our algorithm aims to reduce the difference between the conditional distributions in source and target spaces where a class-based transformation is applied to each of the transformed source samples. The complexity of the latter step is linear on the number of features in the space, i.e., $O(f)$.

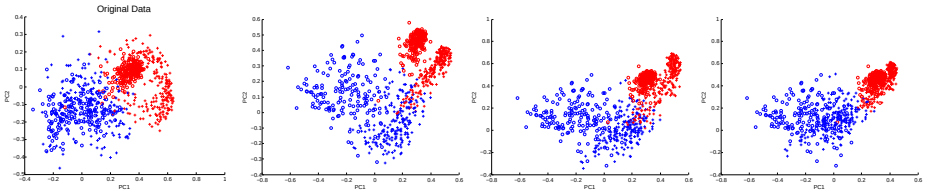
3 The Transductive Transfer Machine

We propose the following pipeline (see Table 1 for the notation):

1. **MMD** – A global linear transformation G^1 is applied to both $\mathtt{X}^{src}$ and $\mathtt{X}^{trg}$ such that the marginal $P(G^1(\mathtt{X}^{src}))$ becomes more similar to $P(G^1(\mathtt{X}^{trg}))$.
2. **TransGrad** – For a finer grained adaptation of the marginal, a local transformation is applied to each transformed source domain sample $G_i^2(G^1(\mathbf{x}_{src}^i))$.
3. **TST** – Finally, aiming to reduce the difference between the conditional distributions in source and target spaces, a class-based transformation is applied to each of the transformed source samples $G_{y^i}^3(G_i^2(G^1(\mathbf{x}_{src}^i)))$.

Figure 1 illustrates the effect of the three steps of the pipeline above on a dataset composed of subset of digits 1 and 2 from the MNIST and USPS datasets. The effect of step (MMD) is to bring the mean of the two distributions closer to each other while it projects the data into its principal components directions of the full data including the source and target. [1]. We use a marginal

[1] In Figure 1(a), the feature space is visualised in 2D using PCA projection and only two classes are shown, but the MMD computation was done on a higher dimensional



(a) MMD (b) TransGrad (c) TST

Fig. 1. Effect of the steps of the TTM pipeline on digits 1 and 2 of the MNIST→USPS datasets, visualised in 2D through PCA. The source dataset (MNIST) is indicated by stars, the target dataset (USPS) is indicated by circles, red indicates samples of digit 1 and blue indicates digit 2 (better viewed on the screen). This figure has been reproduced from [3] with permission.

distribution adaptation method which relates to the works of [14, 15, 12]. This uses the empirical Maximum Mean Discrepancy (MMD) to compare different distributions and compute a lower-dimensional embedding that minimises the distance between the expected values of samples in source and target domains.

For the second step of our pipeline (TransGrad), we proposed a method that distorts the source probability density function towards target clusters. We employ a sample-wise transformation that uses likelihoods of source samples given a GMM that models target data. Up to our knowledge, this is the first time a sample-based transformation is proposed for transfer learning.

In the final step (TST), the source class-conditional distributions are iteratively transformed to become more similar to their corresponding target conditionals. A related approach has been followed in [12] using pseudo-labels to iteratively update a supervised version of MMD. We adopt a method that uses insights from Arnold et al. [16], who used the ratio between the expected class-based posterior probability of target samples and the expected value of source samples per class. This effectively re-scales the source feature space. Our method is a more complex transformation, as each individual feature is both scaled and translated, with different parameters per class. We describe early experiments with TST in [17].

The next subsections detail each of the steps above.

3.1 Shared space detection using MMD

In the first step of our pipeline, we look for a shared space projection that reduces dimensionality of the data whilst minimising the reconstruction error. As explained in [12], one possibility for that is to search for an orthogonal transformation matrix $\mathtt{A} \in \mathbb{R}^{f \times k}$ such that the embedded data variance is maximised as follows:

$$\max_{\mathtt{A}^\top \mathtt{A} = \mathtt{I}} tr(\mathtt{A}^\top \mathtt{X} \mathtt{H} \mathtt{X}^\top \mathtt{A}) \ , \tag{1}$$

space on samples from 10 classes. For these reasons it may not be easy to see that the means of source and target samples became closer after MMD.

Table 1. Notation and acronyms used most frequently in this paper (also used in [3]).

$\mathtt{X} = [\mathbf{x}^1, \cdots, \mathbf{x}^i, \cdots, \mathbf{x}^n]^\top \in \mathbb{R}^{n \times f}$	Input data matrix with n samples of f features
$\mathbf{x}^i = (x_1^i, \cdots, x_j^i, \cdots, x_f^i)^\top$	Feature vectors
$\mathtt{Y} = (y^1, \cdots, y^n)^\top$	Array of class labels associated to $\mathtt{X}$
$\mathcal{Y} = \{1, \cdots, C\}$	Set of classes
$\mathtt{X}^{src} \in \mathbb{R}^{n_{src} \times f}$, $\mathtt{X}^{trg} \in \mathbb{R}^{n_{trg} \times f}$	Source and target data matrices
Λ_{src}	Classification model trained with $\mathtt{X}^{src}$
$G(\mathtt{X})$	Transformation function
θ	transfer rate parameter
T	Number of iterations
$\lambda = \{w_k, \boldsymbol{\mu}_k, \Sigma_k, k = 1, \cdots, K\}$	GMM parameters with K components
$E^{src}[x_j, y^i], E^{trg}[x_j, y^i]$	Joint expectation of feature j and label y^i
$D(p, q)$	Dissimilarity between two distributions
$\nabla_{\mathbf{b}^i} \mathcal{L}(\lambda_{trg} \vert \mathbf{x}^{src})$	Gradient of the log likelihood with respect to $\mathbf{b}^i$
γ	TransGrad translation regulator
TL, ITL, TTL	Transfer Learning, Inductive TL, Transductive TL
MMD	Maximum Mean Discrepancy
TransGrad	Sample-based transformation using gradients
TST	Class-based Translation and Scaling Transform

where $\mathtt{X} = [\mathtt{X}^{src}; \mathtt{X}^{trg}] \in \mathbb{R}^{f \times n_{src} + n_{trg}}$ is the input data matrix that combines source and target samples, $tr(\cdot)$ is the trace of a matrix and $\mathtt{H} = \mathtt{I} - \frac{1}{n_{src}+n_{trg}} \mathbb{1}$ is a centring matrix where $\mathbb{1}$ is a $(n_{src}+n_{trg}) \times (n_{src}+n_{trg})$ matrix of ones. The optimisation problem can be efficiently solved by eigen-decomposition. However, the above PCA-based representation may not reduce the difference between source and target domains. Following [14, 18, 15, 12] we adopt the Maximum Mean Discrepancy (MMD) as a measure to compare different distributions. This algorithm searches for a projection matrix, $\mathtt{A} \in \mathbb{R}^{f \times k}$ which aims to minimise the distance between the samples means of the source and target domains:

$$\left\| \frac{1}{n_{src}} \sum_{i=1}^{n_{src}} \mathtt{A}^\top \mathbf{x}^i - \frac{1}{n_{trg}} \sum_{j=n_{src}+1}^{n_{src}+n_{trg}} \mathtt{A}^\top \mathbf{x}^j \right\|^2 = tr(\mathtt{A}^T \mathtt{X} \mathtt{M} \mathtt{X}^\top \mathtt{A}) \tag{2}$$

where $\mathtt{M}$ is the MMD matrix and is computed as follows:

$$\mathtt{M}^{ij} = \begin{cases} \frac{1}{n_{src} n_{src}}, & \mathbf{x}^i, \mathbf{x}^j \in \mathtt{X}^{src} \\ \frac{1}{n_{trg} n_{trg}}, & \mathbf{x}^i, \mathbf{x}^j \in \mathtt{X}^{trg} \\ -\frac{1}{n_{src} n_{trg}}, & \text{otherwise.} \end{cases}$$

The optimisation problem then is to minimise (2) such that (1) is maximised, i.e. solve the following eigen-decomposition problem: $(\mathtt{X}\mathtt{M}\mathtt{X}^\top + \epsilon \mathtt{I})\mathtt{A} = \mathtt{X}\mathtt{H}\mathtt{X}^\top \mathtt{D}$, obtaining the eigenvectors $\mathtt{A}$ and the eigenvalues on the diagonal matrix $\mathtt{D}$. The effect is to obtain a lower dimensional shared space between the source and target domains. Consequently under the new representation $\mathtt{A}^\top \mathtt{X}$, the marginal distributions of the two domains are drawn closer to each other.

3.2 Sample-based adaptation with TransGrad

We propose a sample-based transformation to perform a finer PDF adaptation of the source domain. We assume that the transformation from the source to the target domain is locally linear, i.e. a sample's feature vector $\mathbf{x}^i$ from the source domain is mapped to the target space by

$$G_i^2(\mathbf{x}^i) = \mathbf{x}^i + \gamma \mathbf{b}^i \ , \tag{3}$$

where the f dimensional vector $\mathbf{b}^i$ represents a local offset in the target domain and γ is a translation regulator. In order to impose as few assumptions as possible, we shall model the unlabelled target data, $\mathbf{X}^{trg}$ by a mixture of Gaussian probability density functions $p(\mathbf{x}) = \sum_{k=1}^{K} w_k p(\mathbf{x}|\lambda_k)$ whose parameters are denoted by $\lambda = \{w_k, \boldsymbol{\mu}_k, \Sigma_k, k = 1, \cdots, K\}$ where w_k, $\boldsymbol{\mu}_k$ and Σ_k denote the weight, mean and covariance matrix of Gaussian component k respectively, K denotes the number of Gaussians and $p(\mathbf{x}|\lambda_k) = \mathcal{N}(\boldsymbol{\mu}_k, \Sigma_k)$.

We formulate the problem of finding optimal translation parameters $\mathbf{b}$ as one of maximising the likelihood of the translated source sample measured in the target domain. Under the assumption of $\mathbf{x}^i$ being independent and identically distributed, the likelihood of a source sample after transformation can be written as

$$p(G_i^2(\mathbf{x}_i^{src})|\lambda) = \sum_{k=1}^{K} w_k p(\mathbf{x}_i^{src} + \mathbf{b}^i|\lambda_k) \ , \tag{4}$$

where $p(G_i^2(\mathbf{x}^i) = \mathbf{x}^i + \mathbf{b}^i|\lambda_k)$ is defined as

$$p(G_i^2(\mathbf{x})|\lambda_k) = \frac{1}{(2\pi)^{f/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}(G_i^2(\mathbf{x}) - \boldsymbol{\mu}_k)^\top \Sigma_k^{-1}(G_i^2(\mathbf{x}) - \boldsymbol{\mu}_k)\right\} \ . \tag{5}$$

Let $P(G_i^2(\mathbf{x}^{src})|\lambda_k)$ be the contribution of the k^{th} Gaussian mixture component to the likelihood of the transformed source point $G_i^2(\mathbf{x}^{src})$, i.e.

$$P(G_i^2(\mathbf{x}^{src})|\lambda_k) = \frac{p(G_i^2(\mathbf{x}^{src})|\lambda_k) w_k}{\sum_{k=1}^{K} w_k p(G_i^2(\mathbf{x}^{src})|\lambda_k)} \ . \tag{6}$$

We wish to maximise $P(G_i^2(\mathbf{x}^{src})|\lambda) = \prod_{k=1}^{K} P(G_i^2(\mathbf{x}^{src})|\lambda_k)$, or more conveniently, its natural logarithm, with respect to the unknown parameter $\mathbf{b}^i$. The gradient of the log likelihood, $\mathcal{L}(\lambda_{trg}|\mathbf{x}^{src}) = \log(P(\mathbf{x}^{src}|\lambda_{trg}))$ with respect to $\mathbf{b}^i$ can be written as

$$\begin{aligned} \nabla_{\mathbf{b}^i}\mathcal{L}(\lambda_{trg}|\mathbf{x}^{src}) = \sum_{k=1}^{K} & P(\mathbf{x}^i + \mathbf{b}^i|\lambda_k) \times \\ & \nabla_{\mathbf{b}^i}\left\{-\tfrac{1}{2}(\mathbf{x}^i + \mathbf{b}^i - \boldsymbol{\mu}_k)^\top \Sigma_k^{-1}(\mathbf{x}^i + \mathbf{b}^i - \boldsymbol{\mu}_k)\right\} \ . \end{aligned} \tag{7}$$

Differentiating with respect to $\mathbf{b}^i$, we get

$$\nabla_{\mathbf{b}^i}\mathcal{L}(\lambda_{trg}|\mathbf{x}^{src}) = \sum_{k=1}^{K} P(\mathbf{x}^i + \mathbf{b}^i|\lambda_k)\Sigma_k^{-1}[-\mathbf{x}^i + \boldsymbol{\mu}_k - \mathbf{b}^i] \ . \tag{8}$$

Setting the gradient to zero, we get a relationship between the offset $\mathbf{b}^i$ and the Gaussian component parameters

$$\mathbf{b}^i = \frac{\sum_{k=1}^{K} P(\mathbf{x}^i + \mathbf{b}_0^i|\lambda_k)\Sigma_k^{-1}(\mathbf{x}^i - \boldsymbol{\mu}_k)}{\sum_{k=1}^{K} P(\mathbf{x}^i + \mathbf{b}_0^i|\lambda_k)\Sigma_k^{-1}} \ . \tag{9}$$

where $\mathbf{b}_0^i$ is an initial value of $\mathbf{b}^i$, which is set to a vector of zeros. In our experiments, we ran (9) only once, though one could iterate it further. We do not recommend the use of a very large number of iterations, as all transformed samples could eventually collapse to their nearest $\boldsymbol{\mu}_k$.

In practice, equation 3 translates $\mathbf{x}_i^{src}$ using the combination of the translations between $\mathbf{x}_i^{src}$ and $\boldsymbol{\mu}_k$, weighted by the likelihood of $G_i(\mathbf{x}_i^{src})$ given λ_k.

3.3 Conditional distribution adaptation with TST

In order to adapt the class-conditional distribution mismatch between the corresponding clusters of the two domains, we introduce a set of linear class-specific transformations. To achieve this, one can assume that a Gaussian Mixture Model fitted to the source classes can be adapted in a way that it matches to target classes. While the general GMM uses full covariance matrices, we follow Reynolds et al. [19] and use only diagonal covariance matrices. This way, the complexity of the estimation system becomes linear in f. In our experiments, we further simplify the model for this step of the pipeline by using only one Gaussian model per class.

In order to adapt the class conditional distributions one can start with an attempt to match the joint distribution of the features and labels between corresponding clusters of two domains. However, as explained in Section 1, labelled samples are not available in the target domain. We thus use posterior probability of the target instances to build class-based models in the target domain. We restrict our class-based adaptation method to a translation and scale transformation (abbreviated as TST). This approximation makes the computational cost very attractive.

The proposed adaptation is introduced by means of a class-based transformation $G_{y^i}(\mathtt{X})$ which aims to adjust the mean and standard deviation of the corresponding clusters from the source domain, i.e., each feature j of each sample $\mathbf{x}^i$ is adapted as follows

$$G_{y^i}(x_j^i) = \frac{x_j^i - E^{src}[x_j, y^i]}{\sigma_{j,y_i}^{src}} \sigma_{j,y^i}^{trg} + E_{\Lambda_{src}}^{trg}[x_j, y^i] \ , \forall i = 1\colon n_{src} \ , \tag{10}$$

where $E^{src}[x_j, y^i]$ is the joint expectation of the feature x_j and labels y^i, and σ_{j,y^i}^{src} is the standard deviation of feature x_j of the source samples labelled as y^i, defined by

$$E^{src}[x_j, y^i] = \frac{\sum_{i=1}^{n_{src}} x_j^i \mathbb{1}_{[y]}(y^i)}{\sum_{i=1}^{n_{src}} \mathbb{1}_{[y]}(y^i)} \ . \tag{11}$$

Here $\mathbb{1}_{[y]}(y^i)$ is an indicator function[2].

[2] Equations (11) and (12) rectify equations from [16], as we discussed in [17].

An estimation of the target join expectation is thus formulated as

$$E^{trg}[x_j, y] \approx E^{trg}_{\Lambda_{src}}[x_j, y] = \frac{\sum_{i=1}^{n_{trg}} x_j^i P_{\Lambda_{src}}(y|\mathbf{x}_i)}{\sum_{i=1}^{n_{trg}} P_{\Lambda_{src}}(y|\mathbf{x}_i)} \tag{12}$$

and we propose to estimate the standard deviation per feature and per class using

$$\sigma^{trg}_{j,y^i} = \sqrt{\frac{\sum_{n=1}^{n_{trg}} (x_j^n - E^{trg}_{\Lambda_{src}}[x_j, y^i])^2 P_{\Lambda_{src}}(y^i|\mathbf{x}_n)}{\sum_{n=1}^{n_{trg}} P_{\Lambda_{src}}(y^i|\mathbf{x}_n)}} \ . \tag{13}$$

In other words, in a common TTL problem, the joint expectation of the features and labels over source distribution, $E^{src}[x_j, y^i]$, is not necessarily equal to $E^{trg}[x_j, y^i]$. Therefore, one can argue that if the expectations in the source and target domains are similar, then the model Λ learnt on the source data will generalise well to the target data. Consequently the less these distributions differ, the better the trained model will perform.

Since the target expectation $E^{trg}_{\Lambda_{src}}[x_j, y^i]$ is only an approximation based on the target's posterior probabilities, rather than the ground-truth labels (which are not available in the target set), there is a danger that samples that would be miss-classified could lead to negative transfer. To alleviate this, we follow Arnold et al.'s [16] suggestion and smooth out the transformation by applying the following:

$$G^3_{y^i}(x_j^i) = (1-\theta)x_j^i + \theta G_{y^i}(x_j^i) \ , \tag{14}$$

with $\theta \in [0, 1]$.

3.4 Iterative refinement of the conditional distribution

Matching the marginal distributions does not guarantee that the conditional distribution of the target can be approximated to that of the source. To our knowledge, most of the recent works related to this issue [7, 20–22] are Inductive TL methods and they have access to some labelled data in the target domain which in practice makes the posteriors' estimations easier.

Instead, our class-specific transformation method (TST), reduces the difference between the likelihoods $P(G^3_y(\mathbf{x}^{src})|y = c)$ and $P(\mathbf{x}|y = c)$ by using the target posteriors estimated from a model trained on gradually modified source domain (eq. 14). Hence, these likelihood approximations will not be reliable unless we iterate over the whole distribution adaptation process and retrain the classifier model using $G^3_y(\mathbf{x}^{src})$.

3.5 Stopping criterion

In order to automatically control the number of the iterations in our pipeline, we introduce a domain dissimilarity measure inspired by sample selection bias corrections techniques [4, 23].

Many of the sample selection bias techniques are based on weighting samples $\mathbf{x}_i^{src}$ using the ratio $w(\mathbf{x}_i^{src}) = P(\mathbf{x}_i^{trg})/P(\mathbf{x}_i^{src})$. This ratio can be estimated using

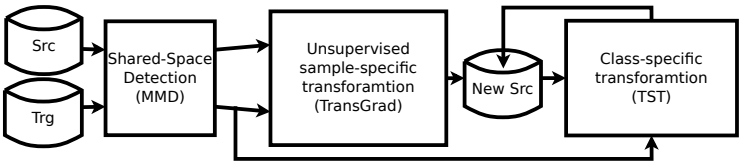


Fig. 2. The Transductive Transfer Machine (TTM).

a classifier that is trained to distinguish between source and target domains, i.e., samples are labelled as either belonging to class *src* or *trg*. Based on this idea, we use this classification performance as a measure of dissimilarity between two domains, i.e., if it is easy to distinguish between source and target samples, it means they are dissimilar. The intuition is that if the domain dissimilarity is high, then more iterations are required for achieving a better match between the domains.

3.6 Algorithm and computational complexity

The proposed method is illustrated in Fig. 2 and algorithm 1. Its computational cost is as follows, where n is the size of the dataset, f is its dimensionality and K is the number of GMM components:

1. MMD: $O(n^2)$ for constructing the MMD matrix, $O(nf^2)$ for covariance computation and and $O(f^3)$ for eigendecomposition.
2. TransGrad: $O(nK)$ for Expectation step of GMM computation, $O(nKf)$ for the computation of diagonal covariance matrices and $O(K)$ for the Maximisation step of the GMM computation. Once the GMM is built, the TransGrad transformation itself is $O(nKf)$.
3. TST: $O(Kf)$ for class specific TST transformations.
4. NN classifier: zero for training and $O(n^2 f)$ for reapplying the classifier.

The *max_iter* parameter is set to 10 for all the experiments, though in the majority of cases, the iterations stop before that because of the criterion of section 3.5.

Algorithm 1 TTM: Transductive Transfer Machine

Input: $\mathtt{X}^{src}, \mathtt{Y}^{src}, \mathtt{X}^{trg}$
Output: $\mathtt{Y}^{trg}$
1. MMD: search for the shared subspace between the two domains (eq. 2)
2. TransGrad: adjust the marginal distribution mismatch between the two domains (eq. 3)
while $(T < max\_iter)$ and $(\|D(G^t(\mathtt{X}^{src}), \mathtt{X}^{trg}\|) > threshold)$ **do**
  3. Find the feature-wise TST transformation (eqs: 12, 13, 10)
  4. Transform the source domain clusters (eq. 14)
  5. Retrain the classifier using the transformed source
end while

For each of the T iterations, the classifier is re-applied and TST is computed. Therefore, the overall complexity of our training algorithm is dominated by the cost of training a GMM (which is low by using diagonal covariances) and iteratively applying a classifier. The core transformations proposed in this paper, TransGrand and TST are $O(nKf)$ and $O(nf)$, respectively, i.e., much cheaper than most methods in the literature.

4 Experimental Evaluation

4.1 Datasets and Feature Extraction

USPS, MNIST, COIL20 and Caltech+office are four benchmark datasets widely adopted to evaluate computer vision and pattern recognition algorithms.

USPS dataset consists of 7,291 training images and 2,007 test images of size 16×16 [24]. *MNIST* dataset has a training set of 60,000 examples and a test set of 10,000 examples of size 28×28. USPS and MNIST datasets follow very different distributions but they share 10 classes of digits. We followed the settings of [12] for USPS→MNIST using their randomly selected samples composed of 1,800 images in USPS as the source data, and 2,000 images in MNIST to form the target data and also switch source-target pairs to get another dataset MNIST→USPS. The images were rescaled to 16×16 pixels, and each represented by a feature vector encoding the gray-scale pixel values. Hence the source and target data can share the same feature space.

COIL20 contains 20 objects classes with 1,440 images [25]. The images of each object were taken 5 degrees apart as the object is rotated on a turntable and each object has 72 images. Each image is 32×32 pixels with 256 gray levels. In our experiments, we followed the settings of [12] and partitioned the dataset into two subsets. *COIL1* and *COIL2*: COIL1 contains all images taken with objects in the orientations of $[0°, 85°] \cup [180°, 265°]$ (quadrants 1 and 3); COIL2 contains all images taken in the orientations of $[90°, 175°] \cup [270°, 355°]$ (quadrants 2 and 4). In this way, subsets COIL1 and COIL2 follow different distributions. One dataset, COIL1→COIL2, was constructed by selecting all 720 images in COIL1 to form the source data, and all 720 images in COIL2 to form the target data. Source-target pairs were switched to form another dataset COIL2→COIL1. Following Long et al. [12], we carried out a pre-processing l_2-normalisation on the raw features of MNIST, USPS, COIL1 and COIL2 datasets.

CALTECH+OFFICE [26, 27] is composed of a 10-class sampling of four datasets; Amazon (images downloaded from online merchants), Webcam (low-resolution images by web camera), DSLR (high-resolution images by a digital SLR camera) and Caltech-256. For the settings we followed [26]: 10 common classes are extracted from all four datasets: Back-pack, Touring-bike, Calculator, Head-phones, Computer-keyboard, Laptop, Computer-monitor, Computer-mouse, Coffee-mug and Video-projector. Each dataset is assumed as a different domain and there are between 8 and 151 samples per category per domain, and 2533 images in total. We followed the feature extraction and experimental settings used in previous works [26, 27]. Briefly, SURF features were extracted and

the images encoded with 800-bins histograms with the codebook trained from a subset of Amazon images. The histograms were then normalised and z-scored to follow a normal distribution in each dimension. We further performed experiments on *CALTECH+OFFICE* where DeCAF features are used as descriptors. DeCAF features are extracted by first training a deep conventional model in a fully supervised setting using a state-of-the-art method [28]. The outputs from the 6^{th} Neural Network layer was used as the visual features, leading to 4096 dimensional DeCAF features.

The second column in Table 2 shows the baseline dissimilarity measure (sec. 3.5) between the two transfer domains.

4.2 Experiments and Results

We coin the iterative version of all our proposed algorithms as Transductive Transfer Machine (TTM) where TTM0 refers to when we have an iterative version of TST, TTM1 is the combination of the MMD and TST and finally TTM2 is the TTM1 with a further intermediate sample-wise marginal adaptation (TransGrad). We have evaluated the performance of these three methods and compared the performance with two state-of-the-art approaches [26, 12] using the same public datasets and the same settings as those of [12, 26]. Further comparisons with other transductive transfer learning methods such as Transfer Component Analysis [29], Transfer Subspace Learning [30] and Sampling Geodesic Flow (SGF) using the Grassmann manifolds [31] are reported in [12, 26].

Table 2 shows a comparison between our methods and the state-of-the-art methods. As one can note, all the transfer learning methods improve the accuracy over the baseline. Furthermore, our TTM methods generally outperform all the other methods. The main reason for that is that our methods combine three different adaptation techniques which jointly implement a complex transformation that would be difficult to determine in a single step. The order in which these transformations are applied, global (MDD) + sample-based (TransGrad) + conditional (TST), is important because neither MMD nor TransGrad take class labels into account. TST achieves better results if it is applied to data in which the difference between source and target domains is not too large, as it uses estimates of $P_{\Lambda_{src}}(y|\mathbf{x})$ based on classifiers learnt on the (adapted) source domain. If the marginals were far off the desired solution, the classifier could generate poor estimates of $P_{\Lambda_{src}}(y|\mathbf{x})$, leading to poor transfer. Similarly, the TransGrad transformation is less constrained than MMD, which is why it is important that it is applied after MMD. These three steps complement each other, as each applies transformations of a different nature.

Table 2 shows that in most of the tasks our methods give the best results. The average performance accuracy of TTM2 on 12 transfer tasks is **56.20%**, which is an improvement of **1.32%** over the best performing previous method JDA [12]. JDA also benefits from jointly adapting the marginal and conditional distributions but their approach has the global and class specific adaptations along each other at each iteration which in practice these two might cancel the

Table 2. Recognition accuracies with Nearest Neighbour classifiers on target domains using TTL algorithms. The datasets are abbreviated as M: MNIST, U: USPS, C: Caltech, A: Amazon, W: Webcam, and D: DSLR.

TTL Experiment	Domain Dissimilarity	NN Baseline	GFK (PLS, PCA) [26]	JDA (1NN) [12]	TTM0 (TST + NN)	TTM1 (MMD + TTM0)	TTM2 (TransGrad + TTM1)
M→ U	0.984	65.94	67.22	67.28	75.94	76.61	**77.94**
U→M	0.981	44.70	46.45	59.65	59.79	59.41	**61.15**
COIL1→2	0.627	83.61	72.50	89.31	88.89	88.75	**93.19**
COIL2→1	0.556	82.78	74.17	88.47	**88.89**	88.61	88.75
C→ A	0.548	23.70	41.4	44.78	39.87	44.25	**46.76**
C→ W	0.78	25.76	40.68	**41.69**	41.02	39.66	41.02
C→ D	0.786	25.48	41.1	45.22	50.31	44.58	**47.13**
A→ C	0.604	26.00	37.9	39.36	36.24	35.53	**39.62**
A→ W	0.743	29.83	35.7	37.97	37.63	**42.37**	39.32
A→ D	0.85	25.48	36.31	**39.49**	33.75	29.30	29.94
W→ C	0.752	19.86	29.3	**31.17**	26.99	29.83	30.36
W→ A	0.717	22.96	**35.5**	32.78	29.12	30.69	31.11
W→ D	0.51	59.24	80.89	89.17	85.98	89.17	**89.81**
D→ C	0.78	26.27	30.28	31.52	29.65	31.25	**32.06**
D→ A	0.790	28.50	**36.1**	33.09	31.21	29.75	30.27
D→ W	0.471	63.39	79.1	89.49	85.08	**90.84**	88.81

effect of each other hence limiting the final model from being well fitted into the target clusters. While in JDA the number of iterations is fixed to 10, in our algorithm we based this number on a sensible measure of domain dissimilarity.

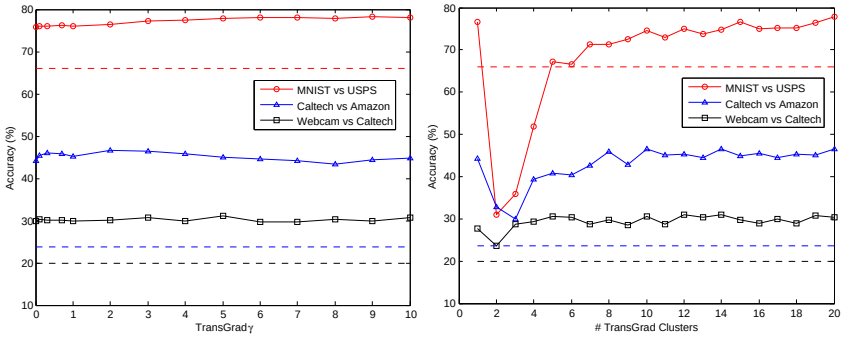
GFK [26] performs well on some of the Office+Caltech experiments but poorly on the others. The reason is that the subspace dimension should be small enough to ensure that different sub-spaces transit smoothly along the geodesic flow, which may not be an accurate representation of the input data. JDA and TTM perform much better by learning an accurate shared space.

For a comparison using state-of-the-art features, in Table 3 we present further results using Deep Convolutional Activation Features (DeCAF) features [32]. We followed the experimental setting in [26] for unsupervised domain adaptation for Caltech+office dataset, except that instead of using SURF, we used DeCAF. In this set of experiments we compared our TTM method with methods that adapt the classifiers hyperplanes or using auxiliary classifiers, namely; the Adaptive Support Vector Machines (SVM-A) [7], Domain Adaptation Machine (DAM) [33] and DA-M2S [34]. DAM was designed to make use of multiple source domains. For a single source domain scenario, the experiments were repeated 10 times by using randomly generated subsets of source and target domains and the mean performance is reported in Table 3.

Note that in Table 3 the baseline without any transformation using DeCAF features and NN classifier is significantly better than the results of Table 2, simply because DeCAF features are better than SURF. As one can see our TTM

Table 3. Results on Caltech+office dataset using DeCAF features. The methods are abbreviated as: M0: Baseline (no transfer), M1: SVM-A[7], M2: DAM [33], M3: DA-M2S (w/o depth) [34], M4: JDA (1NN) [12] and M5: TTM (NN).

	C→A	C→W	C→D	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W
M0	85.70	66.10	74.52	70.35	64.97	57.29	60.37	62.53	98.73	52.09	62.73	89.15
M1	83.54	81.72	74.58	74.36	70.58	96.56	85.37	96.71	78.14	91.00	76.61	83.89
M2	84.73	82.48	78.14	76.60	74.32	93.82	**87.88**	96.31	81.27	**91.75**	79.39	84.59
M3	84.27	82.87	75.83	78.11	71.04	**96.62**	86.38	**97.12**	77.60	91.37	78.14	83.31
M4	89.77	83.73	86.62	82.28	78.64	80.25	83.53	90.19	100	85.13	**91.44**	98.98
M5	**89.98**	**86.78**	**89.17**	**83.70**	**89.81**	81.36	80.41	88.52	**100**	82.90	90.81	**98.98**



(a) TransGrad γ (b) TransGrad k

Fig. 3. Effect of different γ values and number of GMM clusters in the TransGrad step of our framework on the final performance of the pipeline for three cross-domain experiments. The dashed line shows the baseline accuracy for each experiment.

method outperforms the other state-of-the-art approaches in most of the cases, gaining on average 2.10% over the best performing state-of-the-art method of M2S(w/o depth).

To validate that TTM can achieve an optimal performance under a wide range of parameter values, we conducted sensitivity analysis on MNIST→USPS, Caltech→Amazon and Webcam→Caltech. We ran TTM with varying values of the regulator γ of the TransGrad step, and the results are in figure 3(a). One can see that for all these datasets, the performance improves as γ grows but it plateaus when $\gamma = 5$. For this reason we used $\gamma = 5$ in all experiments in the remaining of this paper.

We also ran TTM with varying number Gaussians K in the TransGrad step for the target GMM. Theoretically as the number of GMM components increases the translations get more accurate and the performance becomes more stable. We plot the classification accuracy w.r.t. K in figure 3(b). Note that for $K = 1$, TransGrad contributes to an improvement over the baseline, as it induces a global shift towards the target set. But in general, for values of K smaller than the number of classes, we do not actually expect TransGrad to help, as it will

shift samples from different classes towards the same clusters. This explains why the performance increases with K for $K > 2$. Based on this result, we adopted $K = 20$ in all other experiments of this paper.

We have also compared the time complexity of our TTM algorithm against JDA [12] in the transfer task from MNIST digits dataset to USPS digits dataset. Both algorithms were implemented in Matlab and were evaluated on a Intel Core2 64bit, 3GHz machine running Linux. We averaged time measurements of 5 experiments. The JDA algorithm took 21.38 ± 0.26 seconds and our full TTM framework took 4.42 ± 0.12 seconds, broken down as: 0.40 ± 0.01 seconds to find the appropriate shared space using the MMD, 1.90 ± 0.06 to perform the sample-wise marginal distribution adaptations using TransGrad and finally 2.42 ± 0.12 seconds to apply the iterative conditional distribution adaptations (TST). Therefore, the proposed TTM outperforms JDA in most of the cases requiring one fifth of its computational time.

5 Conclusions

In this paper, we introduced transductive transfer machine (TTM), which aims to adapt both the marginal and conditional distributions of the source samples so that they become more similar to those of target samples, leading to an improvement in the classification results in transfer learning scenarios.

TTM's pipeline consists of the following steps: first, a global linear transformation is applied to both source and target domain samples, so that their expected values are matched. Then we proposed a novel method that applies a sample-based transformation to source samples. This leads to a finer adaptation of their marginal distribution, taking into account the likelihood of each source sample given the target PDF. Finally, we proposed to iteratively adapt the class-based posterior distribution of source samples using an efficient linear transformation whose complexity mostly depends on the number of features. In addition, we proposed to use an unsupervised similarity measure to automatically determine the number of iterations needed. Our approach outperformed state-of-the-art methods on various datasets, with a lower computational cost.

In [3], we present a follow-up work which adds a step that automatically selects the most appropriated classifier and its kernel parameter, leading to a significant improvement of the results presented here.

It is worth pointing out that TTM is a general framework with applicability beyond object recognition and could be easily applied to other domains, even outside Computer Vision. For future work, we suggest studying combinations of TTM with semi-supervised learning methods and feature learning algorithms. Another exciting direction is to combine TTM with voting classification algorithms (c.f. [35]).

Acknowledgements. We are grateful for the support of EPSRC/dstl contract EP/K014307/1 (Signal processing in a network battlespace) and EPSRC project S3A, EP/L000539/1. During part of the development of this work, TdC had been working in Neil Lawrence's group at the University of Sheffield.

References

1. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering **22** (2010) 1345–1359
2. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: Proc of the IEEE Conf on Computer Vision and Pattern Recognition, CVPR. (2011)
3. FarajiDavar, N., deCampos, T., Kittler, J.: Adaptive transductive transfer machines. In: Proc British Machine Vision Conf (BMVC), Nottingham (2014)
4. Cortes, C., Mohri, M., Riley, M., Rostamizadeh, A.: Sample selection bias correction theory. In: Proceedings of the 19th international conference on Algorithmic Learning Theory, Berlin, Heidelberg, Springer-Verlag (2008) 38–53
5. Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schölkopf, B.: Covariate shift by kernel mean matching. Dataset shift in machine learning **3** (2009) 131–160
6. Joachims, T.: Transductive inference for text classification using support vector machines. In: Proc Int Conf Machine Learning, ICML, San Francisco, CA, USA (1999) 200–209
7. Bruzzone, L., Marconcini, M.: Domain adaptation problems: A dasvm classification technique and a circular validation strategy. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI) **32** (2010) 770–787
8. Gopalan, R., Li, R., Chellappa, R.: Unsupervised adaptation across domain shifts by generating intermediate data representations. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI) (2014) DOI 10.1109/TPAMI.2013.249.
9. Chu, W.S., De la Torre, F., Cohn, J.F.: Selective transfer machine for personalized facial action unit detection. In: Proc of the IEEE Conf on Computer Vision and Pattern Recognition, CVPR. (2013)
10. Dai, W., Chen, Y., Xue, G., Yang, Q., Yu, Y.: Translated learning: Transfer learning across different feature spaces. In: Neural Information Processing Systems. (2008) 353–360
11. Borgwardt, K.M., Gretton, A., Rasch, M.J., Kriegel, H., Schalkopf, B., Smola, A.J.: Integrating structured biological data by kernel maximum mean discrepancy. In: Proc Int Conf Intelligent Systems for Molecular Biology. (2006)
12. Long, M., Wang, J., Ding, G., Yu, P.: Transfer learning with joint distribution adaptation. In: Proc Int Conf on Computer Vision, ICCV. (2013)
13. Aytar, Y., Zisserman, A.: Tabula rasa: Model transfer for object category detection. In: Proc Int Conf on Computer Vision, ICCV. (2011)
14. Gretton, A., Borgwardt, K., Rasch, M., Scholkopf, B., Smola, A.: A kernel method for the two sample problem. In: Proc of the Neural Information Processing Systems, NIPS, MIT Press (2007) 513–520
15. Sun, Q., Chattopadhyay, R., Panchanathan, S., Ye, J.: A two-stage weighting framework for multi-source domain adaptation. In: Proc of the Neural Information Processing Systems, NIPS. (2011) 505–513
16. Arnold, A., Nallapati, R., Cohen, W.W.: A comparative study of methods for transductive transfer learning. In: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops. ICDMW, Washington, DC, USA, IEEE Computer Society (2007) 77–82
17. FarajiDavar, N., deCampos, T., Kittler, J., Yan, F.: Transductive transfer learning for action recognition in tennis games. In: VECTaR workshop, in conjunction with ICCV. (2011)

18. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. In: Proceedings of the 21st international jont conference on Artifical intelligence, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2009) 1187–1192
19. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. In: Digital Signal Processing. (2000)
20. Chen, M., Weinberger, K.Q., Blitzer, J.: Co-training for domain adaptation. In: Proc of the Neural Information Processing Systems, NIPS. (2011) 2456–2464
21. Quanz, B., Huan, J., Mishra, M.: Knowledge transfer with low-quality data: A feature extraction issue. In Abiteboul, S., Bolhm, K., Koch, C., Tan, K., eds.: Proceedings of the 27th International Conference on Data Engineering (ICDE), Hannover, Germany, IEEE Computer Society (2011) 769–779
22. Zhong, E., Fan, W., Peng, J., Zhang, K., Ren, J., Turaga, D.S., Verscheure, O.: Cross domain distribution adaptation via kernel mapping. In: Int Conf Knowledge Discovery and Data mining, KDD, ACM (2009) 1027–1036
23. Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of Statistical Planning and Inference **90** (2000) 227–244
24. Cun, Y.L., Boser, B., Denker, J.S., Howard, R.E., Habbard, W., Jackel, L.D., Henderson, D.: Handwritten digit recognition with a back-propagation network. In Touretzky, D.S., ed.: Advances in Neural Information Processing Systems 2. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1990) 396–404
25. Nene, S.A., Nayar, S.K., Murase, H.: Columbia university image library COIL-20 (1996) Available from http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php (retrieved on 30 June 2014).
26. Gong, B., Shi, Y., Sha, F., Grauman, K.: Geodesic flow kernel for unsupervised domain adaptation. In: Proc of the IEEE Conf on Computer Vision and Pattern Recognition, CVPR. (2012) 2066–2073
27. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (SURF). Comput. Vis. Image Underst. **110** (2008) 346–359
28. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Proc of the Neural Information Processing Systems, NIPS. (2012) 1106–1114
29. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks **22** (2011) 199–210
30. Si, S., Tao, D., Geng, B.: Bregman divergence-based regularization for transfer subspace learning. IEEE Trans. Knowl. Data Eng. **22** (2010) 929–942
31. Gopalan, R., Li, R., Chellappa, R.: Domain adaptation for object recognition: An unsupervised approach. In Metaxas, D.N., Quan, L., Sanfeliu, A., Gool, L.J.V., eds.: Proc Int Conf on Computer Vision, ICCV. (2011) 999–1006
32. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: DeCAF: A deep convolutional activation feature for generic visual recognition. Technical Report CoRR arXiv:1310.1531, Cornell University Library (2013)
33. Duan, L., Tsang, I.W., Xu, D., Chua, T.: Domain adaptation from multiple sources via auxiliary classifiers. In: Proceedings of the 26th Annual International Conference on Machine Learning. ICML, New York, NY, USA, ACM (2009) 289–296
34. L. Chen, W. Li, D.X.: Recognizing RGB images by learning from RGB-D data. In: IEEE International Conference on Computer Vision and Pattern Recognition. CVPR (2014)
35. Gao, J., Fan, W., Jiang, J., Han, J.: Knowledge transfer via multiple model local structure mapping. In: Knowledge Discovery and Data Mining. (2008) 283–291