Spectral Bisection Tree Guided Deep Adaptive Exemplar Autoencoder for Unsupervised Domain Adaptation

Ming Shao[1] **Zhengming Ding**[1] **Handong Zhao**[1] **Yun Fu**[1,2]
[1]Department of Electrical and Computer Engineering, [2]College of Computer and Information Science
Northeastern University, Boston, MA 02115, USA
{mingshao, allanding, hdzhao, yunfu}@ece.neu.edu

Abstract

Learning with limited labeled data is always a challenge in AI problems, and one of promising ways is transferring well-established source domain knowledge to the target domain, i.e., domain adaptation. In this paper, we extend the deep representation learning to domain adaptation scenario, and propose a novel deep model called "Deep Adaptive Exemplar AutoEncoder (DAE^2)". Different from conventional denoising autoencoders using corrupted inputs, we assign semantics to the input-output pairs of the autoencoders, which allow us to gradually extract discriminant features layer by layer. To this end, first, we build a spectral bisection tree to generate source-target data compositions as the training pairs fed to autoencoders. Second, a low-rank coding regularizer is imposed to ensure the transferability of the learned hidden layer. Finally, a supervised layer is added on top to transform learned representations into discriminant features. The problem above can be solved iteratively in an EM fashion of learning. Extensive experiments on domain adaptation tasks including object, handwritten digits, and text data classifications demonstrate the effectiveness of the proposed method.

Introduction

Learning with limited labels has drawn considerable attention in particular with the availability of large amount of training data from different sources. There are a group methods proposed recently that reuse relevant datasets (source) as the auxiliary for effective model learning on the current dataset (target), i.e., transfer learning (Pan and Yang 2010). Most existing transfer learning methods manage to deal with different domains but identical task, i.e., domain adaptation, where the domain shift between the current data (target) and auxiliary data (source) is mitigated.

As to domain adaptation, there are three lines that attract substantial research attention recently: (1) feature space adaptation (Pan et al. 2011; Gong et al. 2012; Long et al. 2014c), (2) classifier adaptation (Bruzzone and Marconcini 2010; Bahadori, Liu, and Zhang 2011; Duan, Xu, and Tsang 2012; Ni, Qiu, and Chellappa 2013), (3) deep feature adaptation (Glorot, Bordes, and Bengio 2011; Mesnil et al. 2012; Chen et al. 2012; Donahue et al. 2014). While feature space adaptation attempts to find common subspace or smooth transitions to mitigate domain discrepancy, classifier adaptation builds transferable classifiers for the target data. Different from them, deep feature adaptation is more flexible due to the adaptable building block. In addition, deep structure is able to abstract domain invariant descriptors through layers related semantics. Nonetheless, few works in this line have been done so far for unsupervised domain adaptation, where target labels are totally missing.

In this paper, following the line of "deep feature adaptation", we propose a novel framework called "Deep Adaptive Exemplar AutoEncoder" (DAE^2), as illustrated in Figure 1. Our model can exploit the semantics, and explicitly couple source and target data in the deep strucutre, which, however, are ignored by the exisiting methods (Glorot, Bordes, and Bengio 2011; Chen et al. 2012). First, we partition source and target data by a spectral bisection tree, and use source-target pairs within the same partition to train linear/non-linear autoencoders (AE). On bottom layers, source-target pairs within larger partitions reflect the underlying data distribution, while on top layers, such pairs within smaller partitions represent the class information. Second, as we only have limited source-target pairs for training, we propose to marginalize over the perturbed terms by minimizing the empirical expectation of loss function in addition to a low-rank coding regularizer which ensures the source and target data are tightly coupled. Then a supervised layer is added on top to generate the cross-domain discriminative features. Extensive experiments on vision, and text datasets demonstrate that the proposed Deep Adaptive Exemplar Autoencoder is able to extract domain invariant features that reduce the divergence between two relevant yet different domains.

Related Work

Adapting feature space methods usually align two domains by seeking for a shared subspace. In (Gong et al. 2012), a geodesic flow kernel is implemented to learn the transitions from source to target domains. Therefore, features projected to the intermediate subspaces are able to represent both source and target data well. Recently, regularizers such as low-rank constraint (Shao, Kit, and Fu 2014) and Maximum Mean Discrepancy (MMD) (Pan et al. 2011; Baktashmotlagh et al. 2013; Long et al. 2014c) have been applied on transfer learning to guide the subspace or kernel learning. However, the shallow structure adopted by above

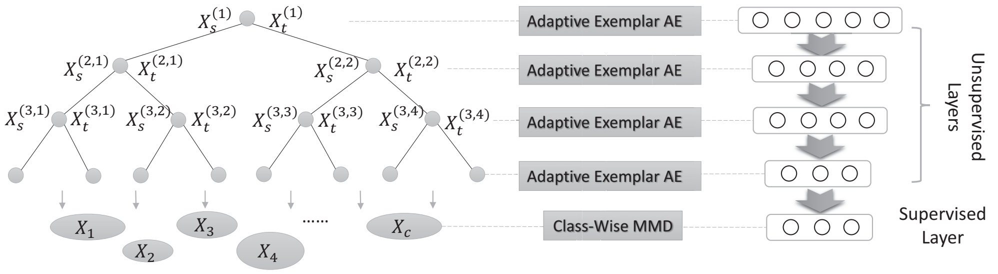


Figure 1: Framework of the proposed method. On the left, the whole dataset is partitioned into a hierarchical structure by a spectral bisection tree. For example, source and target data $X_s^{(1)}$, $X_t^{(1)}$ in the first layer are partitioned into four new small sets: $X_s^{(2,1)}$, $X_t^{(2,1)}$, $X_s^{(2,2)}$, $X_t^{(2,2)}$. The source-target compositions in each layer are used to train a deep adaptive exemplar autoencoder structure on the right, where we obtain L unsupervised layers, and a supervised layer by class-wise MMD criterion and labeled source data $X_1, X_2, ...X_c$ on top.

methods can not explore the semantics well.

A popular strategy to share classifiers is multi-task learning (Argyriou, Evgeniou, and Pontil 2007). Recently, SVM has been broadly discussed on domain adaptation problems regarded with images recognition or video analysis (Bruzzone and Marconcini 2010; Bahadori, Liu, and Zhang 2011; Duan, Xu, and Tsang 2012; Xiao and Guo 2012). In fact, feature space and classifier can be jointly adapted under a single framework (Argyriou, Evgeniou, and Pontil 2007; Long et al. 2014a). However, their shallow structure hardly exploits the semantics, and the requirement of target labels makes it inappropriate for unsupervised domain adaptation.

Deep representation learning can disentangle different explanatory factors of variation (Bengio, Courville, and Vincent 2013), and has been applied for domain adaptation recently (Glorot, Bordes, and Bengio 2011; Chen et al. 2012; Mesnil et al. 2012). However, these methods mix the source and target data and treat them without difference during the model training. Therefore, a large amount of training data may be needed to learn a generic model for both domains. Recently, CNN network "Caffe" attracts substantial research attention for its appealing performance (Donahue et al. 2014); however, its success heavily relies on the tremendous labeled data. Thus, it is more appropriate for visual feature extraction rather than a transfer learning. Another related work is deep multi-view learning (Wang et al. 2015) which maximizes the correlation between different data views. Recently, a deep multi-task learning framework has been proposed in (Yosinski et al. 2014). However, its requirement for target labels does not fit the unsupervised domain adaptation. Most recently, deep low-rank coding (DLRC) is proposed to boost the low-rank transfer learning through the deep structure (Ding, Shao, and Fu 2015) which is similar to our work; however, our data composition strategy, non-linear building block, and supervised layer on top lead to better performance in general.

Data Composition via Spectral Bisection Tree

To explore the semantics of data from coarse to fine, we need to build up a hierarchical structure to generate source-target data pairs fed to the autoencoders. Here we take spectral bisection tree as the partition method since it is running fast and able to discover the underlying data structure.

Building Spectral Bisection Tree

Let us define source and target data as $X_s \in \mathbb{R}^{d\times n_s}$ and $X_t \in \mathbb{R}^{d\times n_t}$ where d is the number of features in each vector, n_s and n_t are the number of source and target data, respectively, therefore, the source target data composition can be written as: $X = [X_s, X_t] \in \mathbb{R}^{d\times n}$, where $n = n_s + n_t$. Given the singular value decomposition of the centered data: $u^{\mathrm{T}}X = \sigma v^{\mathrm{T}}$, where (u, v, σ) is the largest singular value triplet, spectral bisection divides the whole dataset through the hyperplane determined by the singular vector u. Then the dataset can be divided into two parts in the second layer by principles below:

$$X^{(2,1)} = \{x_i | v_i \geq \theta_{\text{med}}\},\ X^{(2,2)} = \{x_i | v_i < \theta_{\text{med}}\}, \quad (1)$$

where the first number of superscripts of X indexes the layer and the second one indexes the partitions in this layer, and θ_{med} is the median entry of vector v to guarantee the partition is balanced. Although the computational complexity for SVD is large when $\min\{d, n\}$ is large, we could use Lanczos algorithm for a fast computation of the first singular value triplet (u, v, σ) (Berry 1992).

After the first data partition by spectral bisection, we have two new partitions: $X^{(2,1)}$, $X^{(2,2)}$ where the superscript indexes the layer and the partition. Specifically, each partition includes both source and target data, which can be represented as: $X^{(2,1)} = [X_s^{(2,1)}, X_t^{(2,1)}]$, $X^{(2,2)} = [X_s^{(2,2)}, X_t^{(2,2)}]$. We can recursively produce exponentially many such partitions which only rely on the number of layers L, such that the total number of partitions is $2^L - 1$.

Source-Target Compositions

The benefit of such hierarchical structure of the compositions of source and target data is obvious: we are able to capture the semantics from coarse to fine. In the bottom layers, source and target data are matched to each other in a global way, while in the top layers, source and target are only matched within each small partition. This means only data within small partitions from different domains are coupled.

However, it is still not clear how to generate such source-target data pairs. An arbitrary composition will fool the autoencoder as no clear semantics are given and inputs/outputs are completely random signals. To that end, we use a non-parametric method to build the source-target data pairs. Specifically, for the j-th partition in the i-th layer, we find the nearest neighbor of $X_s^{(i,j)}$ from $X_t^{(i,j)}$, and build training pairs $(x_{\text{in}}, x_{\text{out}})$ in the following way:

$$\forall x_{\text{in}} \in X_s^{(i,j)}, x_{\text{out}} = NN(x_{\text{in}}, X_t^{(i,j)}), \tag{2}$$

where $NN(\cdot,\cdot)$ represents the nearest neighbor search function, and x_{in}, $X_t^{(i,j)}$ are query, reference, respectively. The resulting input data for an autoencoder is still $X_s^{(i,j)}$, but the output data is a subset of $X_t^{(i,j)}$. We denote $\hat{X}_t^{(i,j)}$ as this new output to differentiate from the original $X_t^{(i,j)}$. In fact, we could also switch source and target data to get more training data pairs, e.g., $X_t^{(i,j)}$ as input and $\hat{X}_s^{(i,j)}$ as output. Combining the original input and output pairs of conventional autoencoder, we have the final training pairs from the source-target composition $X^{(i,j)}$ as: $\left(X_{\text{in}}^{(i,j)}, X_{\text{out}}^{(i,j)}\right) =$

$$(\underbrace{[X_s^{(i,j)}, X_t^{(i,j)}, X_s^{(i,j)}, X_t^{(i,j)}]}_{X_{\text{in}}^{(i,j)}}, \underbrace{[\hat{X}_t^{(i,j)}, \hat{X}_s^{(i,j)}, X_s^{(i,j)}, X_t^{(i,j)}]}_{X_{\text{out}}^{(i,j)}}).$$

Then we combine all such training pairs in the i-th layer $(1 \leq j \leq 2^{(i-1)})$ to build $(X_{\text{in}}^{(i)}, X_{\text{out}}^{(i)})$ in the i-th layer.

Deep Adaptive Exemplar Autoencoder

In this section, we detail how to build deep autoencoders with semantics-awareness source-target data pairs for unsupervised domain adaptation.

Linear Adaptive Exemplar AutoEncoder (AE)

Conventional denoising autoencoder is a single hidden layer neural network, including the input, hidden layer, and output. Suppose the non-linear transform from input to hidden layer is f and the non-linear transform from hidden-layer to output is g, then denoising autoencoder attempts to minimize the following loss function:

$$\mathscr{L}(\widetilde{X}, X) = \frac{1}{4n} \sum_{\substack{x \in X \\ \widetilde{x} \in \widetilde{X}}} (g \circ f(\widetilde{x}) - x)^2, \tag{3}$$

where $\widetilde{x}, \widetilde{X}$ are the contaminated versions of x, X with random dropout or additive Gaussian noises, and n is the number of training samples. Intuitively, denoising autoencoder is able to find transform function robust to corruption simulated by dropout or Gaussian noise because it is able to recover the original data from the contaminated ones. Like conventional single hidden layer neural network, the minimization problem concerned with Eq. (3) can be solved by back propagation and gradient descent algorithms.

However, non-linear activations usually drag down the system efficiency. Motivated by the recent work on marginalized denoising autoencoder proposed in (Chen et al. 2012), we replace the non-linear transforms f, g by a single linear transform function $W \in \mathbb{R}^{d \times d}$. We also remove the corruption sampling scheme by marginalization, leading to a new linear exemplar autoencoder:

$$\min_W \mathbb{E}[\|X_{\text{out}} - WX_{\text{in}}\|_{\text{F}}^2], \tag{4}$$

where $\mathbb{E}[\cdot]$ is the mathematical expectation, and $\|\cdot\|_{\text{F}}$ is the matrix Frobenius norm. As we can see the "corrupted data - original data" training pairs have been replaced by the learned source-target compositions with semantics, which can be seen as **exemplars** found through nearest neighbor search. Finally, the transform matrix W can be solved by the followings:

$$\begin{aligned} &W = \mathbb{E}[PQ^{-1}] = \mathbb{E}[P]\mathbb{E}[Q^{-1}], \\ &\text{where } P = X_{\text{out}}X_{\text{in}}^{\text{T}} \text{ and } Q = X_{\text{in}}X_{\text{in}}^{\text{T}}. \end{aligned} \tag{5}$$

When there are infinite many samples in X_{in} and X_{out}, i.e., $n_s \to \infty$, $n_t \to \infty$, we can obtain the optimized W as both P and Q converge to their expectations. Alternatively, we could also directly compute the empirical expectation of W based on the current observations. For Q, we have:

$$\mathbb{E}[Q] = \mathbb{E} \sum_{x \in X_{\text{in}}} [xx^{\text{T}}] = \sum_{x \in X_{\text{in}}} \mathbb{E}_{p(x_i, x_j)}[xx^{\text{T}}]. \tag{6}$$

However, it is still unclear how to compute the joint probability of transition from the i-th element of x_{in} to j-th element of x_{in}. In this paper, we use the following weighted kernel similarity as the metric for the transition probability:

$$p(x_i, x_j) = \frac{\exp(-(x_i - x_j)^2/2\gamma^2)}{\sum_{i \neq j} \exp(-(x_i - x_j)^2/2\gamma^2)}, \tag{7}$$

where i, j index the elements in x, and γ is the bandwidth of Gaussian kernel. Therefore, Eq. (6) can be rewritten as:

$$\mathbb{E}[Q_{(i,j)}] = \sum_{x \in X_{\text{in}}} p(x_i, x_j) x_i x_j. \tag{8}$$

We can compute $\mathbb{E}[P]$ in a similar way:

$$\mathbb{E}[P_{(i,j)}] = \sum_{\substack{x \in X_{\text{in}} \\ \widetilde{x} \in X_{\text{out}}}} p(\widetilde{x}_i, x_j) \widetilde{x}_i x_j, \tag{9}$$

where subscripts of P, Q indicate the elements in the matrix.

Recently, low-rank coding has been applied to transfer learning scenario to guide the shared subspace learning (Shao, Kit, and Fu 2014; Ding, Shao, and Fu 2015), as it can derive a locality-awareness reconstruction between domains, where source and target data are accurately aligned:

$$\min_Z \text{rank}(Z), \text{s.t.}, W[X_s, X_t] = WX_sZ, \tag{10}$$

where "rank$(\cdot)$" indicates the matrix rank, Z is the low-rank coefficient matrix that can recover the structure of X despite of noise. Integrating Eq. (10) and Eq. (4), we obtain the linear **adaptive** exemplar autoencoder:

$$\begin{aligned} &\min_{W,Z} \mathbb{E}[\|X_{\text{out}} - WX_{\text{in}}\|_{\text{F}}^2] + \lambda\text{rank}(Z) \\ &\text{s.t.}, WX = WX_sZ, \end{aligned} \tag{11}$$

where λ is a balancing parameter. It can be seen that we have casted the original unconstrained optimization problem in Eq. (4) to a new constrained problem. Solutions for problem above will be detailed in the later sections.

Non-Linear Adaptive Exemplar AutoEncoder (AE)

Recall in Eq. (3) we minimize the loss function about input x and output $g \circ f(\widetilde{x})$ under single hidden layer neural network framework. Under the transfer learning scenario with the built source-target pairs, we formulate the non-linear adaptive exemplar autoencoder as:

$$\mathscr{L}(X_{\text{in}}, X_{\text{out}}) = \frac{1}{4n} \sum_{\substack{x_{\text{in}} \in X_{\text{in}} \\ x_{\text{out}} \in X_{\text{out}}}} (g \circ f(x_{\text{in}}) - x_{\text{out}})^2. \tag{12}$$

Since we have limited source-target pairs, we still hope to find the empirical expectation or in other words, marginalizing over x_{in} in Eq. (12). This is equal to minimizing the following expectation w.r.t. x_{in}:

$$\frac{1}{4n} \sum_{\substack{x_{\text{in}} \in X_{\text{in}} \\ x_{\text{out}} \in X_{\text{out}}}} \mathbb{E}_{p(x_{\text{in}}, x_{\text{out}})}[(g \circ f(x_{\text{in}}) - x_{\text{out}})^2]. \tag{13}$$

However, the marginalization is not easy to address due to the existence of hidden layer. We first reformulate this problem by the second-order Taylor expansion at the mean vector of X_{in}, and focus on a single source-target pair:

$$\begin{aligned} \mathscr{L}(x_{\text{in}}, x_{\text{out}}) \approx \quad & \mathscr{L}(\mu_{\text{in}}, x_{\text{out}}) + (\mu_{\text{in}} - x_{\text{in}})^{\mathrm{T}} \nabla_{x_{\text{in}}} \mathscr{L} \\ & + \frac{1}{2}(\mu_{\text{in}} - x_{\text{in}})^{\mathrm{T}} \nabla^2_{x_{\text{in}}} \mathscr{L} (\mu_{\text{in}} - x_{\text{in}}), \end{aligned} \tag{14}$$

where μ_{in} is the empirical expectation of x_{in}, namely, the mean vector of X_{in}, $\nabla_{x_{\text{in}}} \mathscr{L}$ is the first-order derivative of $\mathscr{L}$ w.r.t. x_{in}, and $\nabla^2_{x_{\text{in}}} \mathscr{L}$ is the second-order derivative of $\mathscr{L}$ w.r.t. x_{in}, namely, Hessian matrix. It is easy to check that $\mathbb{E}[x_{\text{in}}] = \mu_{\text{in}}$, and therefore, we can obtain the following derivation:

$$\mathbb{E}[\mathscr{L}(x_{\text{in}}, x_{\text{out}})] \approx \mathscr{L}(\mu_{\text{in}}, x_{\text{out}}) + \frac{1}{2} \text{tr}(\Sigma \nabla^2_{x_{\text{in}}} \mathscr{L}), \tag{15}$$

where $\Sigma = \mathbb{E}[(\mu_{\text{in}} - x_{\text{in}})(\mu_{\text{in}} - x_{\text{in}})^{\mathrm{T}}]$ is the covariance matrix of variable x_{in}.

To facilitate the computation of the expectation derived from Eq. (15), we introduce the following approximation. First, we assume each dimension in x_{in} is generated independently, and therefore, Σ is a diagonal matrix with only variance of each feature of x_{in} on the diagonal. This also means the off-diagonal elements in Hessian matrix $\nabla^2_{x_{\text{in}}} \mathscr{L}$ are also zeros. Second, although there is an explicit formulation for $\nabla^2_{x_{\text{in}}} \mathscr{L}$, we further simplify it by dropping certain terms and removing off-diagonal elements, as suggested in (LeCun et al. 2012; Chen et al. 2014). Finally, we obtain the approximation of element loss function as:

$$\mathbb{E}[\mathscr{L}(x_{\text{in}}, x_{\text{out}})] \approx \mathscr{L}(\mu_{\text{in}}, x_{\text{out}}) + \frac{1}{2} \text{tr}(\Sigma D), \tag{16}$$

where diagonal matrix D has i-th non-zero element as:

$$D_{ii} = \sum_{j=1}^{m} \frac{\partial^2 \mathscr{L}}{\partial f(x_{\text{in}})_j^2} \left(\frac{\partial f(x_{\text{in}})_j}{\partial x_{\text{in},i}} \right)^2, \tag{17}$$

where m is the dimension of hidden layer. Integrating low-rank coding constraint, we have the final objective for non-linear single layer adaptive autoencoder:

$$\begin{aligned} \min_{W,Z} & \sum_{x_{\text{out}} \in X_{\text{out}}} \mathscr{L}(\mu_{\text{in}}, x_{\text{out}}) + \frac{1}{2} \sum_{x_{\text{in}} \in X_{\text{in}}} \text{tr}(\Sigma D) + \lambda \text{rank}(Z) \\ & \text{s.t., } WX = WX_s Z. \end{aligned} \tag{18}$$

Solutions

Linear and non-linear adaptive exemplar autoencoders share many contents except for the loss functions described in Eq. (4) and Eq. (13), respectively, which is only related to the solutions of W. Therefore, we do not differentiate one from another at the beginning, but explain the common contents first. In the following part we use symbol $\mathscr{L}$ for the generic loss function with meanings:

$$\mathscr{L} = \begin{cases} \mathbb{E}[\|X_{\text{out}} - WX_{\text{in}}\|^2_{\text{F}} & \text{Linear;} \\ \displaystyle\sum_{x_{\text{out}} \in X_{\text{out}}} \mathscr{L}(\mu_{\text{in}}, x_{\text{out}}) + \frac{1}{2} \sum_{x_{\text{in}} \in X_{\text{in}}} \text{tr}(\Sigma D) & \text{Non-linear.} \end{cases}$$

The proposed learning objectives in Eq. (11) and Eq. (18) can be solved iteratively by Augmented Lagrange Methods (ALM) (Liu et al. 2013). However, its time consuming operations such as matrix inverse and product will drag down the system performance. To that end, we propose a novel first order Taylor expansion like approximation to accelerate the computation here by removing the quadratic terms. First, we convert the original objectives of adaptive exemplar autoencoder to augmented Lagrangian function:

$$\mathscr{L} + \lambda \|Z\|_* + \langle Y, WX - WX_s Z \rangle + \frac{\tau}{2}(\|WX - WX_s Z\|^2_{\text{F}}),$$

where $\tau > 0$ is a penalty parameter, and Y indicates the Lagrangian multiplier. $\langle , \rangle$ indicates matrices inner product, namely, $\langle A, B \rangle = \text{tr}(A^{\mathrm{T}} B)$. Note that as suggested by work in (Liu et al. 2013), we use the matrix nuclear norm $\|Z\|_*$ as the surrogate of the original rank minimization problem in our formulation. Afterwards, we reformulate the last two terms by combining them into a single quadratic term:

$$\begin{aligned} \mathscr{L} + \lambda \|Z\|_* + h(Z, W, Y, \tau), & \text{ where} \\ h(Z, W, Y, \tau) = \frac{\tau}{2}(\|WX - WX_s Z + Y/\tau\|^2_{\text{F}}). & \end{aligned} \tag{19}$$

It should be noted that problem in Eq. (19) is not jointly solvable over Y, Z and W, but can be optimized over each of them by fixing the rests. Thus, we propose to optimize each of them one after another. In the meanwhile, by considering others as constants, we approximate the term h via Taylor expansion at the current point. At t iteration, we optimize:

Update Z:

$$\begin{aligned} Z_{t+1} &= \arg\min_Z \lambda \|Z\|_* + \frac{\eta_z \tau}{2} \|Z - Z_t\|^2_{\text{F}} + \langle \nabla_Z h_t, Z - Z_t \rangle \\ &= \arg\min_Z \frac{\lambda}{\eta\tau} \|Z\|_* + \frac{1}{2} \|Z - Z_t + \nabla_Z h_t\|^2_{\text{F}}, \end{aligned} \tag{20}$$

where $\nabla_Z h_t = \tau(W_t X_s)^{\mathrm{T}}(W_t X - W_t X_s Z_t + Y_t/\tau)$ is the derivative of h w.r.t. Z, $\eta = \|W_t X_s\|^2_2$. The convex problem above can be solved with exact solution via Singular Value Thresholding (SVT) (Cai, Candès, and Shen 2010).

Update W:

$$W_{t+1} = \arg\min_W \mathscr{L}_t + h(Z_t, W, Y_t, \tau). \tag{21}$$

For **linear** adaptive AE, the problem is convex and we can achieve its closed form solution as:

$$\begin{aligned} W_{t+1} &= (Y_t R_t^{\mathrm{T}} + 2\mathbb{E}[X_{\text{out}} X_{\text{in}}^{\mathrm{T}}])(2\mathbb{E}[X_{\text{in}} X_{\text{in}}^{\mathrm{T}}] - \tau R_t {R_t}^{\mathrm{T}})^{-1} \\ &= (Y_t R_t^{\mathrm{T}} + 2\mathbb{E}[P])(2\mathbb{E}[Q] - \tau R_t {R_t}^{\mathrm{T}})^{-1}, \end{aligned} \tag{22}$$

Algorithm 1 Solving adaptive exemplar autoencoder

```
Input: $X_{\text{in}}^{(l)}, X_{\text{out}}^{(l)}$.
Initialize: $\lambda = 1, Z_t = Y_t = 0, \tau_t = 10^{-6}$,
        $t = 0, \rho = 1.1, \tau_{\max} = 10^6, \varepsilon = 10^{-6}$.
while not converged do
  1. Fix other variables and update $Z_{t+1}$ via Eq. (20);
  2. Fix other variables and update $W_{t+1}$ via Eq. (21);
  3. Update ALM multiplier via
     $Y_{t+1} = Y_t + \tau_t W_{t+1}(X - X_s Z_{t+1})$;
  4. Update $\tau$ via $\tau_{t+1} = \min(\rho\tau_t, \tau_{\max})$;
  5. Check if the objective function converges:
     $\|W_{t+1}(X - X_s Z_{t+1})\|_\infty < \varepsilon$.
end while
Output: Low-rank coding $Z^{(l)}$
```

Algorithm 2 Deep adaptive exemplar autoencoder

```
Input: Source and target data $X_s, X_t$,
      source data labels, number of layers $L$.
Initialize: $\gamma = 1, T = 10$.
for $t$ = 1 to $T$
  1. Partition $X$ into set $\{X^{(i,j)}\}, 1 \leq i \leq L$ by Eq. (1);
  2. Build source-target pairs for adaptive AE by Eq. (3);
  3. for $l$ = 1 to $L$
       Learn adaptive AE in layer $l$ by Algorithm 1;
     end for
  4. Learn $Z^{(L+1)}$ on top by class-wise MMD criterion;
  5. Set $X = Z^{(L+1)}$;
end for
Output: Domain invariant feature $Z^{(L+1)}$
```

where $R_t = X - X_s Z_{t+1}$.

For **non-linear** adaptive AE, the loss function described in Eq. (18) is very similar to the conventional loss function of single hidden layer neural network, and both tr(ΣD) and h are differentiable w.r.t. W. Thus, we can implement gradient descent and back propagation algorithms (Rumelhart, Hinton, and Williams 1988) on Eq. (21) for solutions. We elaborate the ALM based solutions in Algorithm 1, where we follow the parameters setting in (Liu et al. 2013).

Deep Feature and EM Training

In our framework, following the layerwise training procedure (Bengio et al. 2007), we can obtain the deep feature layer by layer. Specifically, given $X_{\text{in}}^{(l)}$ and $X_{\text{out}}^{(l)}$, we could learn the new feature by low-rank coding $Z^{(l)}$ through Algorithm 1, which will be used as the new feature for layer $l+1$: $X^{(l+1)} \leftarrow Z^{(l)}$. Suppose we have L layers in our framework, then we use $[Z^{(1)}; Z^{(2)}...; Z^{(L)}]$ as our learned representations from the proposed deep structure, where ";" denotes column-wise concatenation.

Finally, we add a supervised layer on top to facilitate supervised learning given labels of source data in domain adaptation. Recently, maximum mean discrepancy (MMD) has been widely applied in transfer learning problems by minimizing the distance of centers of two domains in the reproducing kernel Hilbert space (RKHS) (Pan et al. 2011). Here we adopt the JDA (Long et al. 2013b) that exploits class-wise MMD criterion as the objective for common feature space learning. To differentiate the learned features from deep adaptive AE, we use $Z^{(L+1)}$ to represent the discriminative features output by JDA. The learned discriminative feature $Z^{(L+1)}$ can be used for data partition again by the spectral bisection tree, and enable to learn new adaptive exemplar autoencoder. This is essentially an EM style learning: (1) In E step, by projecting data to the learned feature space, we estimate the target labels using nearest neighbor rule. (2) In M step, we minimize our objective in Eq. (19) followed by class-wise MMD. This is essentially the complete procedure of our Deep Adaptive Exemplar AutoEncoder framework, which is elaborated in Algorithm 2. Note we set $\gamma = 1$ and the number of iteration $T = 10$ as they will yield good results in most cases.

Experimental Results

We will first summarize the experimental settings in this section, and then compare our methods with existing state-of-the-art works on several benchmark datasets.

Datasets and Experimental Setting

- **MSRC+VOC** is generated by selecting all 1269 images from MSRC[1] and 1530 images from VOC2007[2]. We resize the image to have 256 pixels in length, and extract dense SIFT (DSIFT) as the basic features.
- **USPS+MNIST**[3] has 10 common handwritten digits from USPS and MNIST. Similar to (Long et al. 2013b), 1800 images are randomly sampled from USPS as one domain while another 2000 images are sampled from MNIST as another domain. All images are down-sampled to 16×16.
- **Office+Caltech-256**[4] has been widely adopted as benchmarks for domain adaptation including 10 common categories from "Office" dataset and "Caltech-256". It has four distinct domains: Amazon (A), Webcam (W), DSLR (D), and Caltech-256 (C) and uses 800-dim SURF+BagOfWords features.
- **Reuters-21578** contains text features in different top and subcategories. Specifically there are three large top categories: *orgs*, *people*, and *place*, a few subcategories within each of them. To fairly compare with other methods, we use the preprocessed version of Reuters-21578 from (Gao et al. 2008) as our basic features.

Comparison methods We compare with recent state-of-the-art domain adaptation methods: TSL (Si, Tao, and Geng 2010), MTrick (Zhuang et al. 2011), TCA (Pan et al. 2011), mSDA (Chen et al. 2012), GFK (Gong et al. 2012), DASA (Fernando et al. 2013), TSC (Long et al. 2013a), LTSL (Shao, Kit, and Fu 2014), TJM (Long et al. 2014c), GTL (Long et al. 2014b), GUMA (Cui et al. 2014), ARRLS (Long et al. 2014a), DLRC (Ding, Shao, and Fu 2015).

[1] http://research.microsoft.com/en-us/projects/objectclassrecognition/

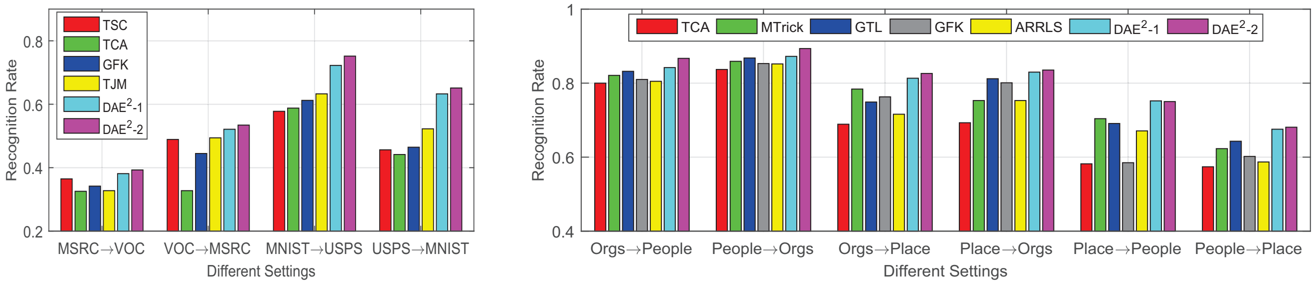
[2] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007

[3] http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html

[4] http://www-scf.usc.edu/∼boqinggo/domainadaptation.html

Table 1: Domain adaptation results (mean ± std %) on the four domains of Office+Caltech-256 dataset. Note A = Amazon, C = Caltech-256, D = DSLR, W = Webcam. We highlight the best performance with **bold** fonts.

Config\Methods	**DASA**	**GFK**	**LTSL**	**TJM**	**TCA**	**mSDA**	**GUMA**	**DLRC**	**DAE^2-1**	**DAE^2-2**
C→W	36.8±0.9	40.7±0.3	39.3±0.6	39.0±0.4	30.5±0.5	38.6±0.8	42.3±0.3	41.7±0.5	42.0±0.7	**45.4±0.6**
C→D	39.6±0.7	38.9±0.9	44.5±0.7	44.6±0.8	35.7±0.5	44.5±0.4	44.7±0.4	46.5±0.6	45.2±0.3	**47.3±0.7**
C→A	39.0±0.5	41.1±0.6	46.9±0.6	46.7±0.7	41.0±0.6	47.7±0.6	46.7±0.6	**49.7±0.4**	45.6±0.5	48.5±0.6
W→C	32.3±0.4	30.7±0.1	29.9±0.5	30.2±0.4	29.9±0.3	33.6±0.4	34.2±0.5	33.8±0.5	31.1±0.3	**34.5±0.5**
W→A	33.4±0.5	29.8±0.6	32.4±0.9	30.0±0.6	28.8±0.6	35.4±0.5	36.2±0.5	36.5±0.7	35.1±0.4	**37.7±0.7**
W→D	80.3±0.8	80.9±0.4	79.8±0.7	89.2±0.9	86.0±1.0	87.9±0.9	73.5±0.4	**94.3±1.1**	89.8±0.5	92.3±0.7
A→C	35.3±0.8	40.3±0.4	38.6±0.4	39.5±0.5	40.1±0.7	40.7±0.6	36.1±0.4	41.7±0.5	40.1±0.6	**45.6±0.4**
A→W	38.6±0.6	39.0±0.9	38.8±0.5	37.8±0.3	35.3±0.8	37.3±0.7	35.9±0.3	41.8±0.9	42.0±0.4	**44.4±0.3**
A→D	37.6±0.7	36.2±0.7	38.3±0.4	39.5±0.7	34.4±0.6	36.3±0.5	38.2±0.8	40.8±0.6	42.0±0.3	**45.3±0.5**



(a) MSRC+VOC and MNIST+USPS

(b) Reuters-21578

Figure 2: Domain adaptation results on MSRC+VOC, MNIST+USPS, and Reuters-21578 datasets.

In this section, we use **DAE^2-1/DAE^2-2** to indicate our linear/nonlinear adaptive exemplar AE, respectively. We set model parameter $\lambda = 1$, and the layer of spectral bisection tree $L = 4$ if not otherwise specified. In all experiments, we strictly follow the setting of unsupervised domain adaptation, with **labeled** source and **unlabeled** target data.

Results and Discussion

In all experiments, we are only accessible to the labels of source domain and use these source labels and data as the references to classify the target data. For different methods, the usages of labeled source data are different. For example, SGF, DASA, TCA, and mSDA are trained in a totally unsupervised way, meaning source labels are not used in the feature learning stage. On the other hand, GFK, LTSL and TSC are trained with source labels, and TJM, ARRLS, DLRC and Ours introduce pseudo target labels to target domains.

Compared to SGF and DASA, GFK, LTSL and TSC achieve better performance in most cases in Table 1. The main reason is they are able to incorporate source labels during the model training to transfer discriminative knowledge to target domain. Similarly, ARRLS, TJM and Ours include pseudo labels of target data to facilitate supervised learning, where labeled source and target data can be accurately aligned. Besides, the EM like iterative learning can further boost the performance, as shown in Table 1, and Figure 2.

Notably, in some cases, mSDA performs better than other competitive algorithms, which indicates that the deep structure of linear denoiser could uncover more discriminative information across two domains. Compared to mSDA, our proposed DAE^2 framework not only builds a deep structure, but also integrates novel source-target data composition methods and low-rank coding term. Therefore, our method could achieve better results than most existing works in benchmark evaluations here, especially on USPS+MNIST, where we achieve significant improvements.

Conclusions

In this paper, we proposed a Deep Adaptive Exemplar Autoencoder framework for unsupervised domain adaptation. First, we partitioned source and target data by a spectral bisection tree, and used learned source-target pairs to train semantics-awareness autoencoders. Second, a new adaptive exemplar autoencoder was learned through the training pairs obtained from the last step, followed by a supervised layer on top. The entire framework can be refined in an EM fashion of learning. Extensive experiments on vision, digits, and text datasets demonstrated the proposed deep domain adaptation framework worked fairly well on benchmark datasets.

Acknowledgement

This research is supported in part by the NSF CNS award 1314484, ONR award N00014-12-1-1028, ONR Young Investigator Award N00014-14-1-0484, NPS award N00244-15-1-0041, and U.S. Army Research Office Young Investigator Award W911NF-14-1-0218.

References

Argyriou, A.; Evgeniou, T.; and Pontil, M. 2007. Multi-task feature learning. In *NIPS*, 41–48. The MIT Press.

Bahadori, M. T.; Liu, Y.; and Zhang, D. 2011. Learning with minimum supervision: A general framework for transductive transfer learning. In *IEEE ICDM*, 61–70. IEEE.

Baktashmotlagh, M.; Harandi, M. T.; Lovell, B. C.; and Salzmann, M. 2013. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, 769–776. IEEE.

Bengio, Y.; Lamblin, P.; Popovici, D.; and Larochelle, H. 2007. Greedy layer-wise training of deep networks. In *NIPS*, 153–160. MIT Press.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE TPAMI* 35(8):1798–1828.

Berry, M. W. 1992. Large-scale sparse singular value computations. *International Journal of Supercomputer Applications* 6(1):13–49.

Bruzzone, L., and Marconcini, M. 2010. Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *IEEE TPAMI* 32(5):770–787.

Cai, J.-F.; Candès, E. J.; and Shen, Z. 2010. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization* 20(4):1956–1982.

Chen, M.; Xu, Z.; Sha, F.; and Weinberger, K. Q. 2012. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 767–774.

Chen, M.; Weinberger, K. Q.; Sha, F.; and Bengio, Y. 2014. Marginalized denoising auto-encoders for nonlinear representations. In *ICML*, 1476–1484.

Cui, Z.; Chang, H.; Shan, S.; and Chen, X. 2014. Generalized unsupervised manifold alignment. In *NIPS*, 2429–2437.

Ding, Z.; Shao, M.; and Fu, Y. 2015. Deep low-rank coding for transfer learning. In *IJCAI*, 3453–3459.

Donahue, J.; Jia, Y.; Vinyals, O.; Hoffman, J.; Zhang, N.; Tzeng, E.; and Darrell, T. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 647–655.

Duan, L.; Xu, D.; and Tsang, I. 2012. Domain adaptation from multiple sources: A domain-dependent regularization approach. *IEEE TNNLS* 23(3):504–518.

Fernando, B.; Habrard, A.; Sebban, M.; and Tuytelaars, T. 2013. Unsupervised visual domain adaptation using subspace alignment. In *IEEE ICCV*, 2960–2967. IEEE.

Gao, J.; Fan, W.; Jiang, J.; and Han, J. 2008. Knowledge transfer via multiple model local structure mapping. In *ACM SIGKDD*, 283–291. ACM.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, 513–520. ACM.

Gong, B.; Shi, Y.; Sha, F.; and Grauman, K. 2012. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE CVPR*, 2066–2073. IEEE.

LeCun, Y. A.; Bottou, L.; Orr, G. B.; and Müller, K.-R. 2012. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer. 9–48.

Liu, G.; Lin, Z.; Yan, S.; Sun, J.; Yu, Y.; and Ma, Y. 2013. Robust recovery of subspace structures by low-rank representation. *IEEE TPAMI* 35(1):171–184.

Long, M.; Ding, G.; Wang, J.; Sun, J.; Guo, Y.; and Yu, P. S. 2013a. Transfer sparse coding for robust image representation. In *IEEE CVPR*, 407–414. IEEE.

Long, M.; Wang, J.; Ding, G.; Sun, J.; and Yu, P. S. 2013b. Transfer feature learning with joint distribution adaptation. In *ICCV*, 2200–2207. IEEE.

Long, M.; Wang, J.; Ding, G.; Pan, S. J.; et al. 2014a. Adaptation regularization: A general framework for transfer learning. *IEEE TKDE* 26(5):1076–1089.

Long, M.; Wang, J.; Ding, G.; Shen, D.; and Yang, Q. 2014b. Transfer learning with graph co-regularization. *IEEE TKDE* 26(7):1805–1818.

Long, M.; Wang, J.; Ding, G.; Sun, J.; and Yu, P. S. 2014c. Transfer joint matching for unsupervised domain adaptation. In *IEEE CVPR*, 1410–1417. IEEE.

Mesnil, G.; Dauphin, Y.; Glorot, X.; Rifai, S.; Bengio, Y.; Goodfellow, I. J.; Lavoie, E.; Muller, X.; Desjardins, G.; Warde-Farley, D.; et al. 2012. Unsupervised and transfer learning challenge: a deep learning approach. *ICML Transfer Learning Workshop* 27:97–110.

Ni, J.; Qiu, Q.; and Chellappa, R. 2013. Subspace interpolation via dictionary learning for unsupervised domain adaptation. In *IEEE CVPR*, 692–699. IEEE.

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

Pan, S. J.; Tsang, I. W.; Kwok, J. T.; Yang, Q.; et al. 2011. Domain adaptation via transfer component analysis. *IEEE TNN* 22(2):199–210.

Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1988. Learning representations by back-propagating errors. *Cognitive modeling* 5:696–699.

Shao, M.; Kit, D.; and Fu, Y. 2014. Generalized transfer subspace learning through low-rank constraint. *International Journal of Computer Vision* 109(1-2):74–93.

Si, S.; Tao, D.; and Geng, B. 2010. Bregman divergence-based regularization for transfer subspace learning. *IEEE TKDE* 22(7):929–942.

Wang, W.; Arora, R.; Livescu, K.; and Bilmes, J. 2015. On deep multi-view representation learning. In *ICML*, 1083–1092.

Xiao, M., and Guo, Y. 2012. Semi-supervised kernel matching for domain adaptation. In *AAAI*, 1183–1189.

Yosinski, J.; Clune, J.; Bengio, Y.; and Lipson, H. 2014. How transferable are features in deep neural networks? In *NIPS*, 3320–3328.

Zhuang, F.; Luo, P.; Xiong, H.; He, Q.; Xiong, Y.; and Shi, Z. 2011. Exploiting associations between word clusters and document classes for cross-domain text categorization? *Statistical Analysis and Data Mining: The ASA Data Science Journal* 4(1):100–114.