Learning Transferrable Representations for Unsupervised Domain Adaptation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Supervised learning with large scale labelled datasets and deep layered models has caused a paradigm shift in diverse areas in learning and recognition. However, this approach still suffers from generalization issues under the presence of a domain shift between the training and the test data distribution. Since unsupervised domain adaptation algorithms directly address this domain shift problem between a labelled source dataset and an unlabelled target dataset, recent papers [10, 32] have shown promising results by fine-tuning the networks with domain adaptation loss functions which try to align the mismatch between the training and testing data distributions.

Nevertheless, these recent deep learning based domain adaptation approaches still suffer from issues such as high sensitivity to the gradient reversal hyperparameters [10] and overfitting during the fine-tuning stage. In this paper, we propose a unified deep learning framework where the representation, cross domain transformation, and target label inference are all jointly optimized in an end-to-end fashion for unsupervised domain adaptation. Our experiments show that the proposed method significantly outperforms state-of-the-art algorithms in both object recognition and digit classification experiments by a large margin. We will make our learned models as well as the source code available immediately upon acceptance.

1 Introduction

Recently, deep convolutional neural networks [16, 25, 29] have propelled unprecedented advances in artificial intelligence including object recognition, speech recognition, and image captioning. Although these networks are very good at learning state of the art feature representations and recognizing discriminative patterns, one major drawback is that the network requires huge amounts of labelled training data to fit millions of parameters in the complex network. However, creating such datasets with complete annotations is not only tedious and error prone, but also extremely costly. In this regard, the research community has proposed different mechanisms such as semi-supervised learning [26, 37, 35], transfer learning [22, 30], weakly labelled learning, and domain adaptation. Among these approaches, domain adaptation is one of the most appealing techniques when a fully annotated dataset (e.g. ImageNet [6], Sports1M [13]) is already available as a reference.

The goal of unsupervised domain adaptation, in particular, is as follows. Given a fully labeled source dataset and an unlabeled target dataset, to learn a model which can generalize to the target domain while taking the domain shift across the datasets into account. The majority of the literature [12, 28, 8, 27, 31] in unsupervised domain adaptation formulates a learning problem where the task is to find a transformation matrix to align the labelled source data distribution to the unlabelled target data distribution. Although these approaches have shown promising results, they show accuracy degradation because of the discrepancy between the learning procedure and the actual target inference procedure. In this paper, we aim to address this issue by incorporating the unknown target labels into the training procedure.

In this regard, we formulate a unified deep learning framework where the feature representation, domain transformation, and target labels are all jointly optimized in an end-to-end fashion. The proposed framework first takes as input a batch of labelled source and unlabelled target examples, and maps this batch of raw input examples into a deep representation. Then, the framework computes the loss of the input batch based on a two stage optimization in which it alternates between inferring the labels of the target examples transductively and optimizing the domain transformation parameters.

Concretely, in the transduction stage, given the fixed domain transform parameter, we jointly infer all target labels by solving a discrete multi-label energy minimization problem. In the adaptation stage, given a fixed target label assignment, we seek to find the optimal asymmetric metric between the source and the target data. The advantage of our method is that we can jointly learn the optimal feature representation and the optimal domain transformation parameter, which are aware of the subsequent transductive inference procedure.

Following the standard evaluation protocol in the unsupervised domain adaptation community, we evaluate our method on the digit classification task using MNIST [18] and SVHN[20] as well as the object recognition task using the Office [24] dataset, and demonstrate state of the art performance in comparison to all existing unsupervised domain adaptation methods.

2 Related Work

This paper is closely related to two active research areas: (1) Unsupervised domain adaptation, and (2) Transductive learning.

Unsupervised domain adaptation: [15] casts the zero-shot learning [21] problem as an unsupervised domain adaptation problem in the dictionary learning and sparse coding framework, assuming access to additional attribute information. Recently, [2] proposed the active nearest neighbor algorithm, which combines the component of active learning into the domain adaptation problem and makes a bounded number of active queries to users. Also, [12, 8, 27] proposed subspace alignment based approaches to unsupervised domain adaptation where the task is to learn a joint transformation and projection in which the difference between the source and the target covariance is minimized. However, these methods learn the transformation matrices on the whole source and target dataset without utilizing the source labels.

[31] utilizes a local max margin metric learning objective [34] to first assign the target labels with the nearest neighbor scheme and then learn a distance metric to enforce the negative pairwise distances to be larger than the positive pairwise distances. However, this method learns a symmetric distance matrix shared by both the source and the target domains so the method is susceptible to the discrepancies between the source and the target distributions. Recently, [10, 32] proposed a deep learning based method to learn domain invariant features by providing the reversed gradient signal from the binary domain classifiers. Although this method performs better than aforementioned approaches, their accuracy is limited since domain invariance does not necessarily imply discriminative features in the target domain.

Transductive learning: In the transductive learning [9], the model has access to unlabelled test samples during training. [23] utilizes a semi-supervised label propagation algorithm into the semi-supervised transfer learning problem assuming access to few labeled examples and additional human specified semantic knowledge. [14] tackled a classification problem where predictions are made jointly across all test examples in a transductive [9] setting. The method essentially enforces the notion that the true labels vary smoothly with respect to the input data. We extend this notion to jointly infer the labels of unsupervised target data points in a k-NN graph.

To summarize, our main contribution is to formulate an end-to-end deep learning framework where we learn the optimal feature representation, infer target labels via discrete energy minimization (*transduction*), and learn the transformation (*adaptation*) between source and target examples all jointly. Our experiments on digit classification using MNIST [18] and SVHN[20] as well as the object recognition experiments on Office [24] datasets show state of the art results, outperforming all existing methods by a substantial margin.

3 Method

3.1 Problem Definition and Notation

In the unsupervised domain adaptation, one of the domains (source) is supervised $\{\hat{\mathbf{x}}_i, \hat{y}_i\}_{i\in[N^s]}$ with N^s data points $\hat{\mathbf{x}}_i$ and the corresponding labels $\hat{y}_i$ from a discrete set $\hat{y}_i \in \mathcal{Y} = \{1, \dots, Y\}$. The other domain (target), on the other hand is unsupervised and has N^u data points $\{\mathbf{x}_i\}_{i\in[N^u]}$.

We further assume that two domains have different distributions $\hat{\mathbf{x}}_i \sim p_s$ and $\mathbf{x}_i \sim p_t$ defined on the same space as $\hat{\mathbf{x}}_i, \mathbf{x}_i \in \mathcal{X}$. We consider a case in which there are two feature functions $\Phi_s, \Phi_t : \mathcal{X} \to \mathcal{R}^d$ applicable to source and target separately. These feature functions extract the information both shared among domains and explicit to the individual ones. The way we model common features is by sharing a subset of parameters between feature functions as $\Phi_s = \Phi_{\theta_c,\theta_s}$ and $\Phi_t = \Phi_{\theta_c,\theta_t}$. We use deep neural networks to implement these functions. In our implementation, θ_c corresponds to the parameters in the first few layers of the networks and θ_s, θ_t correspond to the respective final layers. In general, our model is applicable to any hierarchical and differentiable feature function which can be expressed as a composite function $\Phi_s = f_{\theta_s}(g_{\theta_c}(\cdot))$ for the source and similarly for the target.

3.2 Consistent Structured Transduction

Our method is based on jointly learning the transferable domain specific representations for source and target points as well as estimating the labels of the unsupervised data-points. We denote these two main components of our method as transduction and adaptation. The transduction is the sub-problem of labelling unsupervised data points and the adaptation is the sub-problem of solving for the domain shift. In order to solve this joint problem tractably, we exploit two heuristics: cyclic consistency for adaptation and structured consistency for transduction.

Cyclic consistency: One desired property of Φ_s and Φ_t is consistency. If we estimate the labels of the unsupervised data points and then use these points with their estimated labels to estimate the labels of supervised data-points, we want the predicted labels of the supervised data-points to be consistent with the ground truth labels. Using the inner product as an asymmetric similarity metric -$s(\hat{\mathbf{x}}_i, \mathbf{x}_j) = \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_j)$- this consistency can be represented with the following diagram.

$$\begin{tikzcd}
(\hat{\mathbf{x}}_i, \hat{y}_i) \arrow[r, "\text{Transduction}"] & (\mathbf{x}_j, y_j) \arrow[r, "\text{Transduction}"] & (\hat{\mathbf{x}}_i, \hat{y}_i^{pred}) \arrow[ll, bend left, "\text{Cyclic Consistency: } \hat{y}_i = \hat{y}_i^{pred}"]
\end{tikzcd}$$

It can be shown that if the transduction from target to source follows a nearest neighbor rule, cyclic consistency can be enforced without explicitly computing $\hat{y}_i^{pred}$ using the large-margin nearest neighbor (LMNN)[34] rule. For each source point, we enforce a margin such that the similarity between the source point and the nearest neighbor from the target with the same label is greater than the similarity between the source point and the nearest neighbor from the target with a different label. Formally; $\Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_{i^+}) > \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_{i^-}) + \alpha$ where $\mathbf{x}_{i^+}$ is the nearest target having the same class label as $\hat{\mathbf{x}}_i$ and $\mathbf{x}_{i^-}$ is the nearest target having a different class label.

Structured consistency: We enforce a structured consistency when we label the target data-points during the transduction. In other words, if two target data-points are similar to each other, they are more likely to have the same label. To enforce this consistency, we create a k-NN graph of target data points using a similarity metric $\Phi_t(\mathbf{x}_i)^\intercal \Phi_t(\mathbf{x}_j)$. We denote the neighbors of the point $\hat{\mathbf{x}}_i$ as $\mathcal{N}(\hat{\mathbf{x}}_i)$. We enforce structured consistency by penalizing neighboring points of different labels proportional to their similarity score.

Our model leads to the following optimization problem, over the target labels y_i and the feature function parameters $\theta_c, \theta_s, \theta_t$, jointly solving transduction and adaptation.

$$\min_{\substack{\theta_c,\theta_s,\theta_t,\\ y_1,\dots y_{N^u}}} \underbrace{\sum_{i\in[N^s]} [\Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_{i^-}) - \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_{i^+}) + \alpha]_+}_{\text{Cyclic Consistency}} + \lambda \underbrace{\sum_{i\in[N^u]} \sum_{\mathbf{x}_j\in\mathcal{N}(\mathbf{x}_i)} \Phi_t(\mathbf{x}_i)^\intercal \Phi_t(\mathbf{x}_j) \mathbb{1}(y_i \neq y_j)}_{\text{Structured Consistency}}$$

$$s.t. \quad i^+ = \arg\max_{j|y_j=\hat{y}_i} \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_j) \qquad \text{and} \qquad i^- = \arg\max_{j|y_j\neq\hat{y}_i} \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_j) \tag{1}$$

where $\mathbb{1}(a)$ is an indicator function which is 1 if a is true and 0 otherwise. $[a]_+$ is a rectifier function which is equal to $\max(0, a)$.

We solve this optimization problem via alternating minimization through iterating over solving for unsupervised labels y_i(transduction) and learning the similarity metric $\theta_c, \theta_s, \theta_t$ (adaptation). We explain these two steps in detail in the following sections.

3.3 Transduction: Labeling Target Domain

In order to label the unsupervised points, we base our model on the k-nearest-neighbor rule. We simply compute the k-NN supervised data point for each unsupervised data point using the learned metric and transfer the corresponding majority label. Formally, given a similarity metric $\theta_c, \theta_s, \theta_t$, the k-NN rule is $(y_i)^{pred} = \arg\max_y \frac{k_y(\mathbf{x}_i)}{k}$ where $k_y(\mathbf{x}_i)$ is the number of samples having label y in the k nearest neighbors of $\mathbf{x}_i$ from the source domain. One major issue with this approach is the inaccuracy of transduction during the initial stage of the algorithm. Since the learned metric will not be accurate, we expect to see some noisy k-NN sets. Hence, we propose two solutions to solve this problem.

Structured Consistency: Similar to existing graph transduction algorithms [3, 36], we create a k-nearest neighbor (k-NN) graph over the unsupervised data points and penalize disagreements of labels between neighbors.

Reject option: In the initial stage of the algorithm, we let the transduction step use the reject R as an additional label (besides the class labels) to label the unsupervised target points. In other words, our transduction algorithm can decide to not label (reject) some of the points so that they will not be used for adaptation. As the learned metric gets more accurate in the future iterations, transduction algorithm can change the label from R to other class labels.

Using aforementioned heuristics, we define our transduction sub-problem as:

$$\min_{y_1,\dots y_{N^u} \in \mathcal{Y} \cup R} \sum_{i \in [N^u]} l(\mathbf{x}_i, y_i) + \lambda \sum_{i \in [N^u]} \sum_{\mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i)} \Phi_t(\mathbf{x}_i)^\intercal \Phi_t(\mathbf{x}_j) \mathbb{1}(y_i \neq y_j) \tag{2}$$

where $l(\mathbf{x}_i, y) = \begin{cases} 1 - \frac{k_y(\mathbf{x}_i)}{k} & y \in \mathcal{Y} \\ \gamma \max_{y' \in \mathcal{Y}} \frac{k'_y(\mathbf{x}_i)}{k} & y = R \end{cases}$ and γ is relative cost of the reject option.

The $l(\mathbf{x}_i, R)$ is smaller if none of the class has a majority, promoting the reject option for undecided cases. We also modulate the γ during learning to decrease number of reject options in the later stage of the adaptation. This problem can approximately be solved using many existing methods. We use the α-β swapping algorithm from [4] since it is experimentally shown to be efficient and accurate.

3.4 Adaptation: Learning the Metric

Given the predicted labels y_i for unsupervised data points $\mathbf{x}_i$, we can then learn a metric in order to minimize the loss function defined in (1). Following the cyclic consistency construction, the LMNN rule can be represented using the triplet loss defined between the supervised source data points and their nearest positive and negative neighbors among the unsupervised target points. We do not include the target-data points with reject labels during this construction. Formally, we can define the adaptation problem given unsupervised labels as;

$$\min_{\theta_c, \theta_s, \theta_t} \sum_{i \in [N^s]} [\Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_{i^-}) - \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_{i^+}) + \alpha]_+ \tag{3}$$

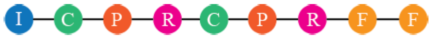
where

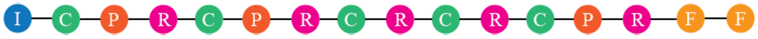
$$i^+ = \arg\max_{j | y_j = \hat{y}_i} \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_j) \quad \text{and} \quad i^- = \arg\max_{j | y_j \neq \hat{y}_i, y_j \neq R} \Phi_s(\hat{\mathbf{x}}_i)^\intercal \Phi_t(\mathbf{x}_j) \tag{4}$$

This minimization is convex if feature functions are convex. This is typically not the case with convolutional neural networks; however, the gradient descent type algorithms are still applicable and work well in practice. Hence, we optimize this function via stochastic gradient descent using the sub-gradients $\frac{\partial loss}{\partial \theta_s}$, $\frac{\partial loss}{\partial \theta_t}$ and $\frac{\partial loss}{\partial \theta_c}$. These sub-gradients can be efficiently computed with back-propagation *(see supplementary material for details).*

3.5 Implementation Details

We use Alexnet [16] and LeNet [17] architectures with small modifications. We remove their final softmax layer and change the size of the final fully connected layer according to the desired feature dimension. We consider the last fully connected layer as domain specific features (θ_s, θ_t) and the rest as common network θ_c. Common network weights are tied between domains, and the final layers are learned separately. In order to have a fair comparison with existing algorithms, we follow the same architecture used by [10] only modifying the final feature dimensionality (embedding size). Explicitly, we use the following architectures for domains:

MNIST and **SVHN:** LeNet[17] as; 

Office: AlexNet[16] as; 

where **C** is convolution, **P** is max-pooling, **R** is ReLU and **F** is fully connected layer.

Since the office dataset is quite small, we do not learn the network from scratch for office experiments and instead we initialize with the weights pre-trained on ImageNet. In all of our experiments, we set the feature dimension as 128. We use stochastic gradient descent to learn the feature function with AdaGrad[7]. We initialize variables with truncated normals having unit variance and use a learning rate of 2.5×10^{-4} and a batch size of 512. We start the rejection penalty with $\gamma = 0.1$ and linearly increase with each epoch as $\gamma = \frac{\#epoch-1}{M} + 0.1$. In our experiments we use $M = 20$ and discuss the sensitivity of our algorithm to M in supplementary material.

Algorithm 1 Transduction with Domain Shift

```
Input: Source $\hat{\mathbf{x}}_{1\cdots N^s}, \hat{y}_{1,\cdots N^s}$, Target $\mathbf{x}_{1,\cdots,N^u}$, Batch Size $2 \times B$
for $t = 0$ to max_iter do
  Sample $\{\hat{\mathbf{x}}_{1,\dots,B}, \hat{y}_{1,\dots,B}\}$, $\{\mathbf{x}_{1,\dots,B}\}$
  Solve (2) for $\{y_{1\cdots B}\}$
  $\alpha \leftarrow 0$
  for $i = 1$ to $B$ do
    if $\hat{y}_i \in y_{1\cdots B}$ and $\exists k \; y_k \in \mathcal{Y}/\hat{y}_i$ then
      Compute $(i^+, i^-)$ using $\{y_{1\cdots B}\}$ in (4)
      Update $\frac{\partial loss}{\partial \theta_\mathbf{c}}, \frac{\partial loss}{\partial \theta_\mathbf{s}}, \frac{\partial loss}{\partial \theta_\mathbf{t}}$
      $\alpha \leftarrow \alpha + 1$
    end if
  end for
  if $\alpha > 0$ then
    $\eta(t) \leftarrow$ Adagrad Rule [7]
    $\theta_\mathbf{c} \leftarrow \theta_\mathbf{c} + \eta(t)\frac{1}{\alpha}\frac{\partial loss}{\partial \theta_\mathbf{c}}$, $\theta_\mathbf{s} \leftarrow \theta_\mathbf{s} + \eta(t)\frac{1}{\alpha}\frac{\partial loss}{\partial \theta_\mathbf{s}}$,
    $\theta_\mathbf{t} \leftarrow \theta_\mathbf{t} + \eta(t)\frac{1}{\alpha}\frac{\partial loss}{\partial \theta_\mathbf{t}}$
  end if
end for
```

4 Experimental Results

We evaluate our algorithm on various unsupervised domain adaptation tasks while focusing on two different problems: hand-written digit classification and object recognition.

Datasets: We use MNIST [18], Street View House Number [20] and the artificially generated version of MNIST -MNIST-M- [10] to experiment our algorithm on the digit classification task. MNIST-M is simply a blend of the digit images of the original MNIST dataset and the color images of BSDS500 [1] following the method explained in [10]. Since the dataset is not distributed directly by the authors, we generated the dataset using the same procedure and further confirmed that the performance is the same as the one reported in [10]. Street View House Numbers is a collection of house numbers collected from Google street view images. Each of these three domains are quite different from each other. Among many important differences, the most significant ones are MNIST being grayscale whilw the others are colored, and SVHN images having extra confusing digits around the centered digit of interest. Moreover, all domains are large-scale, having at least 60k examples over 10 classes.

In addition, we use the Office [24] dataset to evaluate our algorithm on the object recognition task. The office dataset includes images of the objects taken from Amazon, captured with a webcam and captured with a D-SLR. Differences between domains include the white background of Amazon images versus realistic webcam images, and the resolution differences. The Office dataset has fewer images, with a maximum of 2478 per domain over 31 classes.

Baselines: We compare our method against a variety of methods with and without feature learning. **SA***[8] is the dominant state-of-the-art approach not employing any feature learning, and **Backprop(BP)**[10] is the dominant state-of-the-art employing feature learning. We use the available source code of [10] and [8] and following the evaluation procedure in [10], we choose the hyper-parameter of [8] as the highest performing one among various alternatives. We also compare our method with the **source only** baseline which is a convolutional neural network trained only using the source data. This classifier is clearly different from our nearest neighbor classifier; however, we experimentally validated that the CNN always outperformed the nearest neighbor based classifier. Hence, we report the highest performing source only method.

Evaluation: We evaluate all algorithms in a *fully transductive* setup [11]. We feed training images and labels of first domain as the source and training images of the second domain as the target. We evaluate the accuracy on the target domain as the ratio of correctly labeled images to all target images.

4.1 Results

Following the fully transductive evaluation, we summarize the results in Table 1 and Table 2. Table 1 summarizes the results on the object recognition task using office dataset whereas Table 2 summarizes the digit classification task on MNIST and SVHN.

Table 1: Accuracy of our method and the state-of-the-art algorithms on Office dataset.

SOURCE TARGET	AMAZON WEBCAM	D-SLR WEBCAM	WEBCAM D-SLR	WEBCAM AMAZON	AMAZON D-SLR	D-SLR AMAZON
GFK [11]	.398	.791	.746	.371	.379	.379
SA* [8]	.450	.648	.699	.393	.388	.420
DLID [5]	.519	.782	.899	-	-	-
DDC [32]	.618	.950	.985	.522	.644	.521
DAN [19]	.685	.960	.990	.531	.670	.540
BACKPROP [10]	.730	.964	.992	.536	.728	.544
SOURCE ONLY	.642	.961	.978	.452	.668	.476
OUR METHOD (K-NN ONLY)	.727	.952	.915	.575	.791	.521
OUR METHOD (NO REJECT)	.804	.962	.989	.625	.839	.567
OUR METHOD (FULL)	**.814**	**.971**	**.993**	**.663**	**.847**	**.601**

Tables 1&2 show results on object recognition and digit classification tasks covering all adaptation scenarios. Our experiments show that our proposed method outperforms all state-of-the-art algorithms. Moreover, the increase in the accuracy is rather significant when there is a large domain difference such as MNIST↔MNIST-M, MNIST↔SVHN, Amazon↔Webcam and Amazon↔D-SLR. Our hypothesis is that the state-of-the-art algorithms such as [10] are seeking features invariant to the domains whereas we seek an explicit similarity metric explaining both differences and similarities of domains. In other words, instead of seeking an invariance, we seek an equivariance.

Table 2: Accuracy on the digit classification task.

SOURCE TARGET	M-M MNIST	MNIST M-M	SVHN MNIST	MNIST SVHN
SA* [8]	.523	.569	.593	.211
BP [10]	.732	.766	.738	.289
SOURCE ONLY	.483	.522	.549	.162
OUR METHOD(K-NN ONLY)	.805	.795	.713	.158
OUR METHOD(NO REJECT)	.835	.855	.774	.323
OUR METHOD(FULL)	**.839**	**.875**	**.799**	**.413**

Table 2 further suggests that our algorithm is the only one which can successfully perform adaptation from MNIST to SVHN. Clearly the features which are learned from MNIST cannot generalize to SVHN since the SVHN has concepts like color and occlusion which are not available in MNIST. Hence, our algorithm learns SVHN specific features by enforcing accurate transduction in the adaptation.

Another interesting conclusion is the asymmetric results. For example, adapting webcam to Amazon and adapting Amazon to webcam yield very different accuracies. The similar asymmetry exists in MNIST and SVHN as well. This observation validates the importance of an asymmetric modeling.

To evaluate the importance of joint labelling and reject option, we compare our method with self baselines. Our self-baselines are versions of our algorithm not using the reject option (**no reject**) and the version using neither reject option nor joint labelling (**k-NN only**). Results on both experiments suggest that joint labelling and the reject option are both crucial for successful transduction. Moreover, the reject option is more important when the domain shift is large (e.g. MNIST→SVHN). This is expected since transduction under a large shift is more likely to fail a situation that can be prevented with reject option.

4.1.1 Qualitative Analysis

To further study the learned representations and the similarity metric, we performed a series of qualitative analysis in the form of nearest neighbor and tSNE[33] plots.

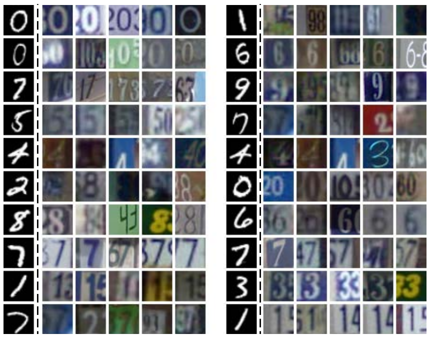


Figure 1: Nearest neighbors for SVHN→MNIST exp. We show an example MNIST image and its 5-NNs.

Figure 1 visualizes example target images from MNIST and their corresponding source images. First of all, our experimental analysis suggests that MNIST and SVHN are the two domains with the largest difference. Hence, we believe MNIST↔SVHN is a very challenging set-up and despite the huge visual differences, our algorithm results in accurate nearest neighbors. On the other hand, Figure 2 visualizes the example target images from webcam and their corresponding nearest source images from Amazon.

The difference between invariance and equivariance is clearer in the tSNE plots of the Office dataset in Figure 3 and the digit classification task in Figure 4. In Figure 3, we plot the distribution of features before and after adaptation for source and target while color coding class labels. We use the learned embeddings as output of Φ_s and Φ_t as an input to tSNE algorithm[33]. As Figure 3 suggests, the source domain is well clustered according to the object classes with and without adaptation. This is expected since the features are specifically fine-tuned to the source domain before the adaptation starts. However, the target domain features have no structure before adaptation. This is also expected since the algorithm did not see any image from the target domain. After the adaptation, target images also get clustered according to the object classes.

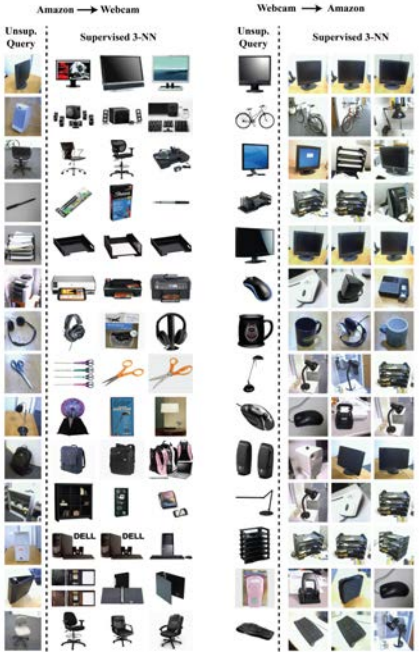


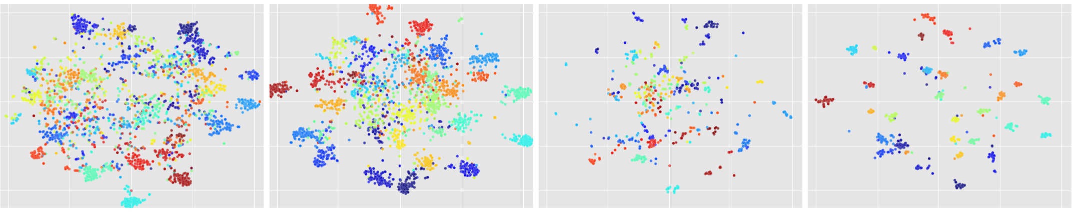
Figure 2: Nearest neighbors for Amazon↔Webcam exp. We show an example Amazon image and its 3-NNs.

In Figure 4, we show the digit images of the source and target after the adaptation. In order to see the effect of common features and domain specific features separately, we compute the low-dimensional embeddings of the output of the shared network (output of the first fully connected layer). We further compute the NN points between the source and target using Φ_s and Φ_t, and draw an edge between NNs. Clearly, the target is well clustered according to the classes and the source is not very well clustered although it has some structure. Since we learn the entire network for digit classification, our networks learn discriminative features in the target domain as our loss depends directly on classification scores in the target domain. Moreover, discriminative features in the target arises because of the transductive modeling. In comparison, state of the art domain invariance based algorithms only try to be invariant to the domains without explicit modeling of discriminative behavior on the target. Hence, our method explicitly models the relationship between the domains and results in an equivarient model while enforcing discriminative behavior in the target.

5 Conclusion

We described an end-to-end deep learning framework for jointly optimizing the optimal deep feature representation, cross domain transformation, and the target label inference for state of the art unsupervised domain adaptation.

Experimental results on digit classification using MNIST[18] and SVHN[20] as well as on object recognition using the Office[24] dataset show state of the art performance with a significant margin. We will make our learned models as well as the source code available immediately upon acceptance.



(a) S. w/o Adaptation (b) S. with Adaptation (c) T w/o Adaptation (d) T with Adaptation

Figure 3: tSNE plots for office dataset Webcam(S)→Amazon(T). Source features were discriminative and stayed discriminative as expected. On the other hand, target features became quite discriminative after the adaptation.

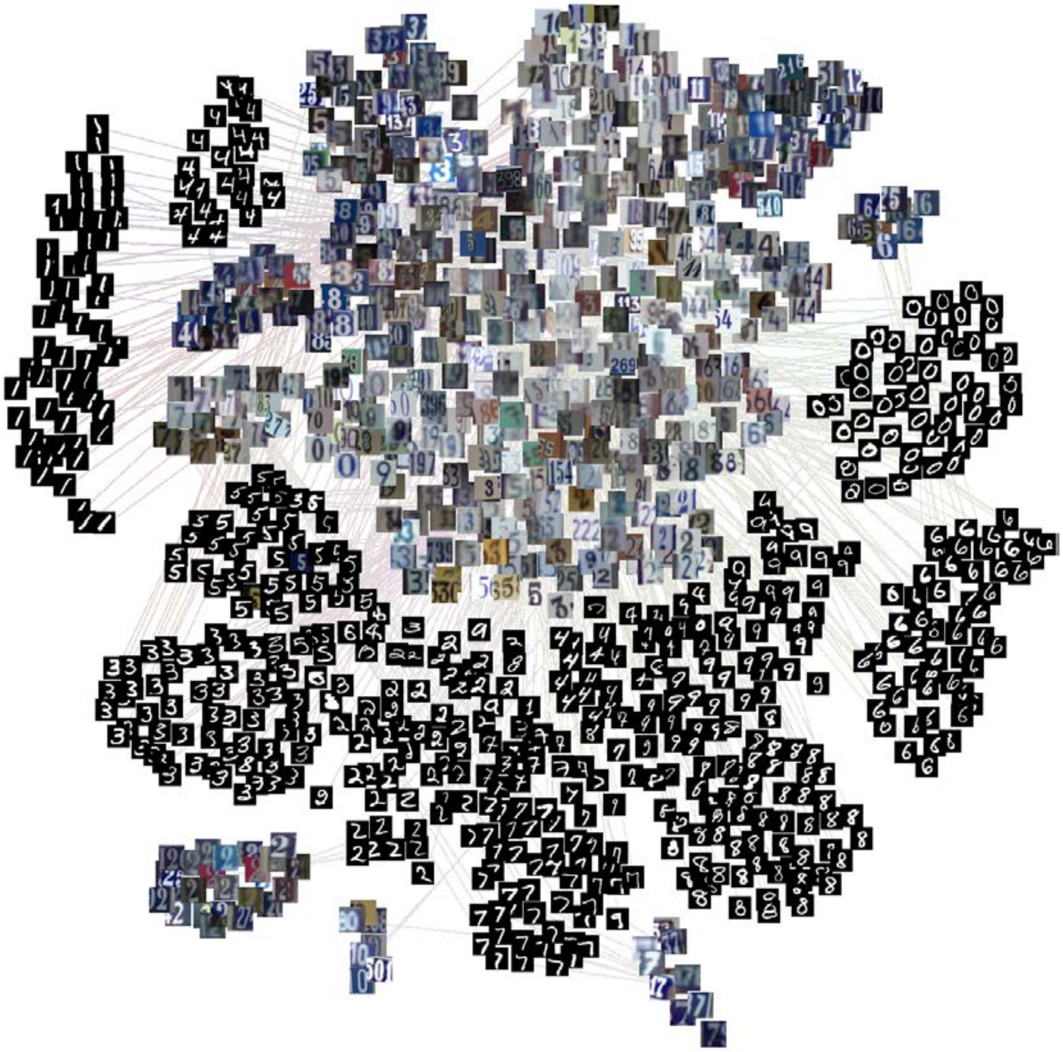


Figure 4: tSNE plot for SVHN→MNIST experiment. Please note that the discriminative behavior only emerges in the unsupervised target instead of the source domain. This explains the motivation behind modeling the problem as transduction. In other words, our algorithm is designed to be accurate and discriminative in the target domain which is the domain we are interested in.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *T-PAMI*, 33:898–916, 2011.

[2] C. Berlind and R. Urner. Active nearest neighbors in changing environments. In *ICML*, 2015.

[3] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *ICML*, 2001.

[4] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *T-PAMI*, 26:1124–1137, 2004.

[5] S. Chopra, S. Balakrishnan, and R. Gopalan. Dlid: Deep learning for domain adaptation by interpolating between domains. In *ICML W*, 2013.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, pages 2121–2159, 2011.

[8] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013.

[9] A. Gammerman, V. Vovk, and V. Vapnik. Learning by transduction. In *UAI*, 1998.

[10] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

[11] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013.

[12] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

[13] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[14] S. Khamis and C. Lampert. Coconut: Co-classification with output space regularization. In *BMVC*, 2014.

[15] E. Kodirov, T. Xiang, Z. Fu, and S. Gong. Domain adaptation for zero-shot learning. In *ICCV*, 2015.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[18] Y. LeCun, C. Cortes, and C. J. Burges. The mnist database of handwritten digits, 1998.

[19] M. Long, C. Yue, J. Wang, and J. Michael. Learning transferable features with deep adaptation networks. *arXiv*, 2015.

[20] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS W*, 2011.

[21] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell. Zero-shot learning with semantic output codes. In *NIPS*, 2009.

[22] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*. ACM, 2007.

[23] M. Rohrbach, S. Ebert, and B. Schiele. Transfer learning in a transductive setting. In *NIPS*, 2013.

[24] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226. Springer, 2010.

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[26] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *ICML*, 2005.

[27] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016.

[28] B. Sun and K. Saenko. Subspace alignment for unsupervised domain adaptation. In *BMVC*, 2015.

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014.

[30] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business, 2012.

[31] T. Tommasi and B. Caputo. Frustratingly easy NBNN domain adaptation. In *ICCV*, 2013.

[32] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*, 2014.

[33] L. van der maaten. Accelerating t-sne using tree-based algorithms. In *JMLR*, 2014.

[34] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, 2006.

[35] X. Zhu. Semi-supervised learning literature survey. 2005.

[36] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. 2002.

[37] X. Zhu, Z. Ghahramani, J. Lafferty, et al. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.