Label Propagation with Augmented Anchors: A Simple Semi-Supervised Learning baseline for Unsupervised Domain Adaptation

Yabin Zhang[123], Bin Deng[12], Kui Jia[12], and Lei Zhang[34]

[1] South China University of Technology, Guangzhou, China
[2] Pazhou Lab, Guangzhou, China
[3] DAMO Academy, Alibaba Group
[4] Dept. of Computing, The Hong Kong Polytechnic University
zhang.yabin@mail.scut.edu.cn, bindeng.scut@gmail.com, kuijia@scut.edu.cn,
cslzhang@comp.polyu.edu.hk

Abstract. Motivated by the problem relatedness between unsupervised domain adaptation (UDA) and semi-supervised learning (SSL), many state-of-the-art UDA methods adopt SSL principles (e.g., the cluster assumption) as their learning ingredients. However, they tend to overlook the very domain-shift nature of UDA. In this work, we take a step further to study the proper extensions of SSL techniques for UDA. Taking the algorithm of label propagation (LP) as an example, we analyze the challenges of adopting LP to UDA and theoretically analyze the conditions of affinity graph/matrix construction in order to achieve better propagation of true labels to unlabeled instances. Our analysis suggests a new algorithm of Label Propagation with Augmented Anchors (A^2LP), which could potentially improve LP via generation of unlabeled virtual instances (i.e., the augmented anchors) with high-confidence label predictions. To make the proposed A^2LP useful for UDA, we propose empirical schemes to generate such virtual instances. The proposed schemes also tackle the domain-shift challenge of UDA by alternating between pseudo labeling via A^2LP and domain-invariant feature learning. Experiments show that such a simple SSL extension improves over representative UDA methods of domain-invariant feature learning, and could empower two state-of-the-art methods on benchmark UDA datasets. Our results show the value of further investigation on SSL techniques for UDA problems.

Keywords: Domain adaptation, semi-supervised learning, label propagation

1 Introduction

As a specific setting of transfer learning [32], unsupervised domain adaptation (UDA) is to predict labels of given instances on a target domain, by learning classification models assisted with labeled data on a source domain that has a different distribution from the target one. Impressive results have been achieved by

learning domain-invariant features [43, 27, 45], especially the recent ones based on adversarial training of deep networks [12, 42, 36, 47]. These methods are primarily motivated by the classical UDA theories [4, 3, 30, 46] that specify the success conditions of domain adaptation, where domain divergences induced by hypothesis space of classifiers are typically involved.

While a main focus of these methods is on designing algorithms to learn domain-invariant features, they largely overlook a UDA nature that shares the same property with the related problem of semi-supervised learning (SSL) — both UDA and SSL argue for a principle that the (unlabeled) instances of interest satisfy basic assumptions (e.g., the cluster assumption [6]), although in SSL, the unlabeled instances follow the same distribution as that of the labeled ones. Given the advantages of SSL methods over models trained with labeled data only [5], it is natural to apply the SSL techniques to domain-invariant features learned by seminal UDA methods [27, 12] so as to boost the performance further. We note that ideal domain alignment can hardly be achieved in practice. Although state-of-the-art results have already been achieved by the combination of vanilla SSL techniques and domain-invariant feature learning [17, 9, 25, 29, 47, 18], they typically neglect the issue that SSL methods are designed for data of the same domain, and their direct use for data with shifted distributions (e.g., in UDA tasks) could result in deteriorated performance.

To this end, we investigate how to extend SSL techniques for UDA problems. Take the SSL method of label propagation (LP) [48] as an example. When there exists such a shift of distributions, edges of an LP graph constructed by affinity relations of data instances could be of low reliability, thus preventing its direct use in UDA problems. To tackle the issue, we analyze in this paper the conditions of the affinity graph (and the corresponding affinity matrix) for better propagation of true labels to unlabeled instances. Our analysis suggests a new algorithm of Label Propagation with Augmented Anchors (A^2LP), which could potentially improve LP via generation of unlabeled virtual instances (i.e., the augmented anchors) with high-confidence label predictions. To make the proposed A^2LP particularly useful for UDA, we generate such virtual instances via a weighted combination of unlabeled target instances, using weights computed by the entropy of their propagated soft cluster assignments, considering that instances of low entropy are more confident in terms of their predicted labels. We iteratively do the steps of (1) using A^2LP to get pseudo labels of target instances, and (2) learning domain-invariant features with the obtained pseudo-labeled target instances and labeled source ones, where the second step, in turn, improves the quality of pseudo labels of target instances. Experiments on benchmark UDA datasets show that our proposed A^2LP significantly improves over the LP algorithm, and alternating steps of A^2LP and domain-invariant feature learning give state-of-the-art results. We finally summarize our main contributions as follows.

– Motivated by the relatedness between SSL and UDA, we study in this paper the technical challenge that prevents the direct use of graph-based SSL methods in UDA problems. We analyze the conditions of the affinity graph/matrix construction for better propagation of true labels to unlabeled instances,

which suggests a new algorithm of A^2LP. A^2LP could potentially improve LP via generation of unlabeled virtual instances (i.e., the augmented anchors) with high-confidence label predictions.

– To make the proposed A^2LP useful for UDA, we generate virtual instances as augmented anchors via a weighted combination of unlabeled target instances, where weights are computed based on the entropy of propagated soft cluster assignments of target instances. Our A^2LP based UDA method alternates in obtaining pseudo labels of target instances via A^2LP, and using the obtained pseudo-labeled target instances, together with the labeled source ones, to learn domain-invariant features. The second step is expected to enhance the quality of pseudo labels of target instances.

– We conduct careful ablation studies to investigate the influence of graph structure on the results of A^2LP. Empirical evidences on benchmark UDA datasets show that our proposed A^2LP significantly improves over the original LP, and the alternating steps of A^2LP and domain-invariant feature learning give state-of-the-art results, confirming the value of further investigating the SSL techniques for UDA problems. The codes are available at https://github.com/YBZh/Label-Propagation-with-Augmented-Anchors .

2   Related works

In this section, we briefly review the UDA methods, especially these [35, 47, 29, 38, 23, 11, 17, 9, 25] involving SSL principles as their learning ingredients, and the recent works [51, 39, 48, 2, 7, 21] on the LP technique.

Unsupervised domain adaptation Motivated by the theoretical bound proposed in [4, 3, 46], the dominant UDA methods target at minimizing the discrepancy between the two domains, which is measured by various statistic distances, such as Maximum Mean Discrepancy (MMD) [27], Jensen-Shannon divergence [12] and Wasserstein distance [37]. They assume that once the domain discrepancy is minimized, the classifier trained on source data only can also perform well on the target ones. Given the advantages of SSL methods over models trained with labeled data only [5], it is natural to apply SSL techniques on domain-invariant features to boost the results further. Recently, state-of-the-art results are achieved by involving the SSL principles in UDA, although they may not have emphasized this point explicitly. Based on the cluster assumption, entropy regularization [13] is adopted in UDA methods [38, 29, 47, 23] to encourage low density separation of category decision boundaries, which is typically used in conjunction with the virtual adversarial training [31] to incorporate the locally-Lipschitz constraint. The vanilla LP method [52] is adopted in [17, 9, 25] together with the learning of domain-invariant features. Based on the mean teacher model of [41], a self-ensembling (SE) algorithm [11] is proposed to penalize the prediction differences between student and teacher networks for the same input target instance. Inspired by the tri-training [50], three task classifiers are asymmetrically used in [35]. However, taking the comparable LP-based methods [17, 9, 25]

as an example, they adopt the vanilla LP algorithm directly with no consideration of the UDA nature of domain shift. By contrast, we analyze the challenges of adopting LP in UDA, theoretically characterize the conditions of potential improvement (cf. **Proposition** 1), and accordingly propose the algorithmic extension of LP for UDA. Such a simple algorithmic extension improves the results dramatically on benchmark UDA datasets.

Label propagation The LP algorithm is based on a graph whose nodes are data instances (labeled and unlabeled), and edges indicate the similarities between instances. The labels of labeled data can propagate through the edges in order to label all nodes. Following the above principle, a series of LP algorithms [51, 39, 48] and the graph regularization methods [2, 7, 21] have been proposed for the SSL problems. Recently, Iscen *et al.* [19] revisit the LP algorithm for SSL problems with the iterative strategy of pseudo labeling and network retraining. Liu *et al.* [26] study the LP algorithm for few-shot learning. Unlike them, we investigate the LP algorithm for UDA problems and alleviate the performance deterioration brought by domain shift via the introduction of virtual instances as well as the domain-invariant feature learning.

3   Semi-supervised learning and unsupervised domain adaptation

Given data sets $X_L = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_l\}$ and $X_U = \{\boldsymbol{x}_{l+1}, ..., \boldsymbol{x}_n\}$ with each $\boldsymbol{x}_i \in \mathcal{X}$, the first l instances have labels $Y_L = \{y_1, ..., y_l\}$ with each $y_i \in \mathcal{Y} = \{1, ..., K\}$ and the remaining $n - l$ instances are unlabeled. We also write them collectively as $X = \{X_L, X_U\}$. The goal of both SSL and UDA is to predict the labels of the unlabeled instances in X_U [5]. In UDA, the labeled data in X_L and unlabeled data in X_U are drawn from two different distributions of the source one $\mathcal{D}_s$ and the target one $\mathcal{D}_t$. Differently, in SSL, the source and target distributions are assumed to be the same, i.e., $\mathcal{D}_s = \mathcal{D}_t$.

3.1   Semi-supervised learning preliminaries

We denote $\psi : \mathcal{X} \to \mathbb{R}^K$ as the mapping function parameterized by $\theta = \{\theta_e, \theta_c\}$, where θ_e indicates the parameters of a feature extractor $\phi : \mathcal{X} \to \mathbb{R}^d$ and θ_c indicates the parameters of a classifier $f : \mathbb{R}^d \to \mathbb{R}^K$. Let $\mathcal{P}$ denote the set of $n \times K$ probability matrices. A matrix $\boldsymbol{P} = [\boldsymbol{p}_1^T; ...; \boldsymbol{p}_n^T] \in \mathcal{P}$ corresponds to a classification on the dataset $X = \{X_L, X_U\}$ by labeling each instance $\boldsymbol{x}_i$ as a label $\hat{y}_i = \arg\max_j \boldsymbol{p}_{ij}$. Each $\boldsymbol{p}_i \in [0, 1]^K$ indicates classification probabilities of the instance $\boldsymbol{x}_i$ to K classes.

The general goal of SSL can be stated as finding $\boldsymbol{P}$ by minimizing the following meta objective:

$$\mathcal{Q}(\boldsymbol{P}) = \mathcal{L}(X_L, Y_L; \boldsymbol{P}) + \lambda\mathcal{R}(X; \boldsymbol{P}), \tag{1}$$

---

[5] We formulate in this paper both the SSL and UDA under the transductive learning setting [5].

where $\mathcal{L}$ represents the supervised loss term that applies only to the labeled data, and $\mathcal{R}$ is the regularizer with λ as a trade-off parameter. The purpose of regularizer $\mathcal{R}$ is to make the learning decision to satisfy the underlying assumptions of SSL, including the smoothness, cluster, and manifold assumptions [5].

For SSL methods based on cluster assumption (e.g., low density separation [6]), their regularizers are concerned with unlabeled data only. As such, they are more amenable to be used in UDA problems, since the domain shift is not an issue to be taken into account. A prominent example of SSL regularizer is the entropy minimization (EM) [13], whose use in UDA can be instantiated as:

$$\mathcal{Q}(\boldsymbol{P}) = \underbrace{\sum_{i=1}^{l} \ell(\boldsymbol{p}_i, y_i)}_{\mathcal{L}(X_L, Y_L; \boldsymbol{P})} + \lambda \underbrace{\sum_{i=l+1}^{n} \sum_{j=1}^{K} -\boldsymbol{p}_{ij} \log \boldsymbol{p}_{ij}}_{\mathcal{R}(X; \boldsymbol{P})}, \tag{2}$$

where ℓ represents a typical loss function (e.g., cross-entropy loss). Objectives similar to (2) are widely used in UDA methods, together with other useful ingredients such as adversarial learning of aligned features [29, 47, 15, 38].

3.2 From graph-based semi-supervised learning to unsupervised domain adaptation

Different from the above EM like methods, the graph-based SSL methods that are based on local (and global) smoothness rely on the geometry of the data, and thus their regularizers are concerned with both labeled and unlabeled instances. The key of graph-based methods is to build a graph whose nodes are data instances (labeled and unlabeled), and edges represent similarities between instances. Such a graph is represented by the affinity matrix $\boldsymbol{A} \in \mathbb{R}_+^{n \times n}$, whose elements $\boldsymbol{a}_{ij}$ are non-negative pairwise similarities between instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Here, we choose the LP algorithm [48] as an instance for exploiting the advantages of graph-based SSL methods. Denote $\boldsymbol{Y} = [\boldsymbol{y}_1^T; ...; \boldsymbol{y}_n^T] \in \mathcal{P}$ as the label matrix with $\boldsymbol{y}_{ij} = 1$ if $\boldsymbol{x}_i$ is labeled as $y_i = j$ and $\boldsymbol{y}_{ij} = 0$ otherwise. The goal of LP is to find a $\boldsymbol{F} = [\boldsymbol{f}_1^T; ...; \boldsymbol{f}_n^T] \in \mathbb{R}_+^{n \times K}$ by minimizing

$$\mathcal{Q}(\boldsymbol{F}) = \underbrace{\sum_{i=1}^{n} \|\boldsymbol{f}_i - \boldsymbol{y}_i\|^2}_{\mathcal{L}(X_L, Y_L; \boldsymbol{F})} + \lambda \underbrace{\sum_{i,j}^{n} \boldsymbol{a}_{ij} \|\frac{\boldsymbol{f}_i}{\sqrt{\boldsymbol{d}_{ii}}} - \frac{\boldsymbol{f}_j}{\sqrt{\boldsymbol{d}_{jj}}}\|^2}_{\mathcal{R}(X; \boldsymbol{F})}, \tag{3}$$

and then the resulting probability matrix $\boldsymbol{P}$ is given by $\boldsymbol{p}_{ij} = \boldsymbol{f}_{ij} / \sum_j \boldsymbol{f}_{ij}$, where $\boldsymbol{D} \in \mathbb{R}_+^{n \times n}$ is a diagonal matrix with its (i,i)-element $\boldsymbol{d}_{ii}$ equal to the sum of the i-th row of $\boldsymbol{A}$. From the above optimization objective, we can easily see that a good affinity matrix is the key success factor of the LP algorithm. So, the straightforward question is that what is the good affinity matrix? As an analysis, we assume the true label of each data instance $\boldsymbol{x}_i$ is y_i, then the regularizer of

the objective (3) can be decomposed as:

$$\mathcal{R}(X; \boldsymbol{F}) = \sum_{y_i = y_j} \boldsymbol{a}_{ij} \| \frac{\boldsymbol{f}_i}{\sqrt{\boldsymbol{d}_{ii}}} - \frac{\boldsymbol{f}_j}{\sqrt{\boldsymbol{d}_{jj}}} \|^2$$
$$+ \sum_{y_i \neq y_j} \boldsymbol{a}_{ij} \| \frac{\boldsymbol{f}_i}{\sqrt{\boldsymbol{d}_{ii}}} - \frac{\boldsymbol{f}_j}{\sqrt{\boldsymbol{d}_{jj}}} \|^2. \quad (4)$$

Obviously, a good affinity matrix should make its element $\boldsymbol{a}_{ij}$ as large as possible if instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are in the same class, and at the same time make those $\boldsymbol{a}_{ij}$ as small as possible otherwise. Therefore, it is rather easy to construct such a good affinity matrix in the SSL setting where the all data are drawn from the same underlying distribution. However, in UDA, due to the domain shift between labeled and unlabeled data, those values of elements $\boldsymbol{a}_{ij}$ of the same class pairs between labeled and unlabeled instances would be significantly reduced, which would prevent its use in the UDA problems as illustrated in Figure 1.

(a) SSL (93.5%)   (b) UDA (64.8%)   (c) UDA with Anchors (79.5%)
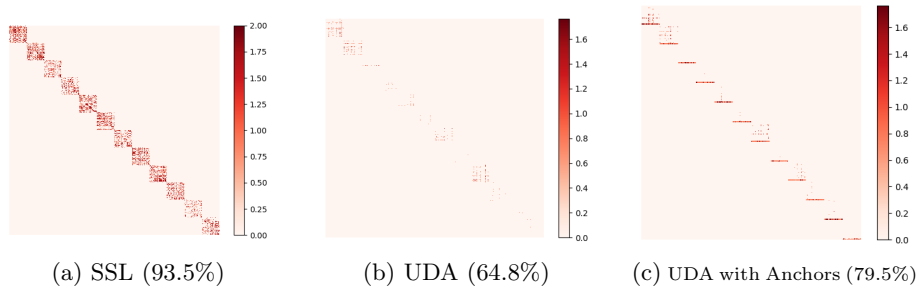
Fig. 1: Visualization of sub-affinity matrices for the settings of (a) SSL, (b) UDA, and (c) UDA with augmented anchors, and their corresponding classification results via the LP. The row-wise and column-wise elements are the unlabeled and labeled instances, respectively. For illustration purposes, we keep elements connecting instances of the same class unchanged, set the others to zero, and sort all instances in the category order using the ground truth category of all data. As we can see, the augmented anchors present better connections with unlabeled target instances compared to the labeled source instances in UDA.

4   Label propagation with augmented anchors

In this section, we first analyze conditions of the corresponding affinity matrix for better propagation of true labels to unlabeled instances, which motivate our proposed A^2LP algorithm. Let Acc be the classification accuracy in X_U by the solution of the LP (Equ. (3)), i.e.,

$$Acc := \frac{|\{\boldsymbol{x}_i \in X_U : \hat{y}_i = y_i\}|}{|X_U|}, \quad (5)$$
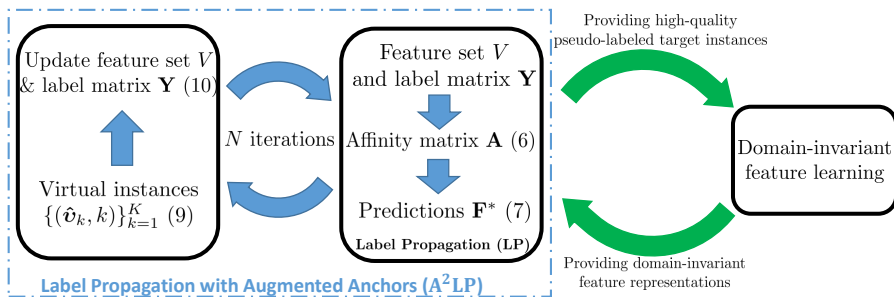
Fig. 2: An illustration of the overall framework of alternating steps of pseudo labeling via A^{2}LP and domain-invariant feature learning. The dashed line rectangle illustrates the algorithm of A^{2}LP, where we iteratively do the steps of (1) augmenting the feature set V and label matrix $\boldsymbol{Y}$ with the generated virtual instances and (2) generating virtual instances by the LP algorithm based on the updated feature set V and label matrix $\boldsymbol{Y}$.

where $\hat{y}_i = \arg\max_j \boldsymbol{f}_{ij}^*$ with $\boldsymbol{F}^* = [\boldsymbol{f}_1^{*T}; ...; \boldsymbol{f}_n^{*T}]$ the solution of Equ. (3).

Proposition 1. *Assume the data satisfy the ideal cluster assumption, i.e., $\boldsymbol{a}_{ij} = 0$ for all $y_i \neq y_j$. Enhancing one zero-valued element $\boldsymbol{a}_{mn}$ between a data instance $\boldsymbol{x}_m$ (labeled or unlabeled) and a labeled instance $\boldsymbol{x}_n \in X_L$ to a positive number, where $y_m = y_n$, the Acc (5) non-decreases, and increases under the condition when originally $\hat{y}_m \neq y_m$.*

The proof of Proposition 1 can be found in the appendices.

Remark 1. Under the assumption of Proposition 1, if we can augment the labeled set X_L with one virtual instance with the true label, whose neighbors are exactly the instances with the same label, then based on Proposition 1, the LP algorithm can get increasing (non-decreasing) *Acc* (Equ. (5)) in X_U.

4.1 The proposed algorithms

Based on the above analysis, we propose the algorithm of Label Propagation with Augmented Anchors (A^{2}LP), as illustrated in Fig. 2. We detail the A^{2}LP method as follows.

Nearest neighbor graph. We construct the feature set $V = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_l, \boldsymbol{v}_{l+1}, \cdots, \boldsymbol{v}_n\}$, where $\boldsymbol{v}_i := \phi_{\theta_e}(\boldsymbol{x}_i)$. The affinity matrix $\boldsymbol{A} \in \mathbb{R}_+^{n \times n}$ is constructed with elements:

$$\boldsymbol{a}_{ij} := \begin{cases} \varepsilon(\boldsymbol{v}_i, \boldsymbol{v}_j), \text{ if } & i \neq j \wedge \boldsymbol{v}_i \in \text{NN}_k(\boldsymbol{v}_j) \\ 0, & otherwise \end{cases} \tag{6}$$

where $\varepsilon(\boldsymbol{v}_i, \boldsymbol{v}_j)$ measures the non-negative similarity between $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$, and NN_k denotes the set of k nearest neighbors in X. Then, we adopt $\boldsymbol{A} = \boldsymbol{A} + \boldsymbol{A}^T$ to make $\boldsymbol{A}$ a symmetric non-negative adjacency matrix with zero diagonal.

Label propagation. The closed-form solution of the objective (Equ. (3)) of the LP algorithm is given by [48] as

$$\boldsymbol{F}^* = (\boldsymbol{I} - \alpha \boldsymbol{S})^{-1}\boldsymbol{Y}, \tag{7}$$

where $\alpha = \frac{2\lambda}{2\lambda+1}$, $\boldsymbol{I}$ is an identity matrix and $\boldsymbol{S} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$.

LP with augmented anchors. Suggested by the Remark 1, we generate virtual instances via a weighted combination of unlabeled target instances, using weights computed by the entropy of their propagated soft cluster assignments, considering that instances of low entropy are more confident in terms of their predicted labels. In particular, we first obtain the pseudo labels of unlabeled target instances by solving Equ. (7), and then we assign the weight w_i to each unlabeled instance $\boldsymbol{x}_i$ by

$$w_i := 1 - \frac{H(\boldsymbol{p}_i^*)}{\log(K)}, \tag{8}$$

where $H(\cdot)$ is the entropy function and $\boldsymbol{p}_{ij}^* = \boldsymbol{f}_{ij}^* / \sum_j \boldsymbol{f}_{ij}^*$. We have $w_i \in [0,1]$ since $0 \le H(\boldsymbol{p}_i^*) \le \log K$. The virtual instances $\{(\hat{\boldsymbol{v}}_{n+k}, k)\}_{k=1}^K$ can then be calculated as:

$$(\hat{\boldsymbol{v}}_{n+k}, k) = \left(\sum_{\boldsymbol{x}_i \in X_U} \frac{\mathbb{1}(k = \hat{y}_i) w_i \phi_{\theta_e}(\boldsymbol{x}_i)}{\sum_{\boldsymbol{x}_j \in X_U} \mathbb{1}(k = \hat{y}_j) w_j}, k \right), \tag{9}$$

where $\mathbb{1}(\cdot)$ is the indicator function. The virtual instances generated by Equ. (9) are relatively robust to the label noise and their neighbors are probably the instances of the same label due to the underlying cluster assumption.

Then, we iteratively do the steps of (1) augmenting the feature set V and label matrix $\boldsymbol{Y}$ with the generated virtual instances and (2) generating virtual instances by the LP algorithm based on updated feature set V and label matrix $\boldsymbol{Y}$. The updating strategies of feature set V and label matrix $\boldsymbol{Y}$ are as follows:

$$V = V \cup \{\hat{\boldsymbol{v}}_{n+1}, \cdots, \hat{\boldsymbol{v}}_{n+K}\}, \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y} \\ \boldsymbol{I} \end{bmatrix}, n = n + K. \tag{10}$$

The iterative steps empirically converge in less than 10 iterations, as illustrated in Sec. 5.1. The implementation of our A^2LP is summarized in Algorithm 1 (line 2 *to* 10).

Alternating steps of pseudo labeling and domain-invariant feature learning. Although our proposed A^2LP can largely alleviate the performance degradation of applying LP to UDA tasks via the introduction of virtual instances, learning domain-invariant features across labeled source data and unlabeled target data is fundamentally important, especially when the domain shift is unexpectedly large. To illustrate the advantage of our proposed A^2LP on generating high-quality pseudo labels of unlabeled data, and to justify the efficacy of the alternating steps of pseudo labeling via SSL methods and domain-invariant feature learning, we empower state-of-the-art UDA methods [44, 22] by replacing their pseudo label generators with our A^2LP, and keep other settings unchanged. Empirical results in Sec. 5.2 testify the efficacy of our A^2LP.

Algorithm 1 Alternating steps of pseudo labeling via A^{2}LP and domain-invariant feature learning.

Input:
Labeled data: $\{X_L, Y_L\} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_l, y_l)\}$
Unlabeled data: $X_U = \{\boldsymbol{x}_{l+1}, ..., \boldsymbol{x}_n\}$
Model parameters: $\theta = [\theta_e, \theta_c]$
Procedure:
 1: **while** Not Converge **do**
 2:      Construct feature set V and label matrix $\boldsymbol{Y}$;
 3:      **for** iter $= 1$ to N **do**   ▷ Pseudo labeling via A^{2}LP
 4:          Compute affinity matrix $\boldsymbol{A}$ by Equ. (6);
 5:          $\boldsymbol{A} \leftarrow \boldsymbol{A} + \boldsymbol{A}^T$;
 6:          $\boldsymbol{S} \leftarrow \boldsymbol{D}^{-1/2} \boldsymbol{A} \boldsymbol{D}^{-1/2}$;
 7:          Get predictions $\boldsymbol{F}^*$ by Equ. (7);
 8:          Calculate the virtual instances by Equ. (9);
 9:          Update V and $\boldsymbol{Y}$ with virtual instances by Equ. (10);
10:      **end for**
11:      Remove added virtual instances, and $n = n - NK$;
12:      **for** iter $= 1$ to M **do**   ▷ Domain-invariant feature learning
13:          Update parameters θ by domain-invariant feature learning (e.g., [44, 22]);
14:      **end for**
15: **end while**

Time Complexity of A^2LP. Computation of our proposed algorithm is dominated by constructing the affinity matrix (6) via k-nearest neighbor graph and solving the closed-form solution (7). Brute-force implementations of them are computationally expensive for datasets with large numbers of instances. Fortunately, the $O(n^2)$ complexity of full affinity matrix construction of the k-nearest neighbor graph can be largely improved via NN-Descent [10], giving rise to an almost linear empirical complexity of $O(n^{1.1})$. Given that the matrix $(\boldsymbol{I} - \alpha\boldsymbol{S})$ is positive-definite, the label predictions $\boldsymbol{F}^*$ (7) can be achieved by solving the following linear system with the conjugate gradient (CG) [16, 53]:

$$(\boldsymbol{I} - \alpha\boldsymbol{S})\boldsymbol{F}^* = \boldsymbol{Y}, \tag{11}$$

which is known to be faster than the closed-form solution (7). Empirical results in the appendices show that such accelerating strategies significantly reduce the time consumption and hardly suffer performance penalties.

5   Experiments

Office-31 [34] is a standard UDA dataset including three diverse domains: Amazon (**A**) from Amazon website, Webcam (**W**) by web camera, and DSLR (**D**) by digital SLR camera. There are $4,110$ images of 31 categories shared across three domains. **ImageCLEF-DA** [1] is a balanced dataset containing three domains: Caltech-256 (**C**), ImageNet ILSVRC 2012 (**I**), and Pascal VOC 2012 (**P**). There

are 12 categories and 600 images in each domain. **VisDA-2017** [33] is a dataset with large domain shift from the synthetic data (**Syn.**) to real images (**Real**). There are about 280K images across 12 categories.

We implement our A^{2}LP based on PyTorch. We adopt the ResNet [14] pre-trained on the ImageNet dataset [8] excluding the last fully connected (FC) layer as the feature extractor ϕ. In the alternating training step, we fine-tune the feature extractor ϕ and train a classifier f of one FC layer from scratch. We update all parameters by stochastic gradient descent with momentum of 0.9, and the learning rate of the classifier f is 10 times that of the feature extractor ϕ. We employ the annealing strategy of learning rate [12] following $\eta_p = \frac{\eta_0}{(1+\mu p)^\beta}$, where p is the process of training iterations linearly changing from 0 to 1, $\eta_0 = 0.01$ and $\mu = 10$. Following [22], we set $\beta = 0.75$ for datasets of Office-31 [34] and ImageCLEF-DA [1], while for VisDA-2017 dataset, $\beta = 2.25$. We adopt the cosine similarity, i.e., $\varepsilon(\boldsymbol{v}_i, \boldsymbol{v}_j) = \frac{<\boldsymbol{v}_i, \boldsymbol{v}_j>}{\|\boldsymbol{v}_i\|\|\boldsymbol{v}_j\|}$, to construct the affinity matrix (6) and compare it with other two alternatives in Sec. 5.1. We empirically set α as 0.75 and 0.5 for the VisDA-2017 dataset and datasets of Office-31 and ImageCLEF-DA, respectively. We use all labeled source data and all unlabeled target data in the training process, following the standard protocols for UDA [12, 27]. For each adaptation task, we report the average classification accuracy and the standard error on three random experiments.

5.1   Analysis

Various Similarity Metrics In this section, we conduct ablative experiments on the **C** $\rightarrow$ **P** task of the ImageCLEF-DA dataset to analyze the influences of graph structures to results of A^{2}LP. To study the impact of similarity measurements, we construct the affinity matrix with other two alternative similarity measurements, namely the Euclidean distance-based similarity $\varepsilon(\boldsymbol{v}_i, \boldsymbol{v}_j) = \exp(-\|\boldsymbol{v}_i - \boldsymbol{v}_j\|^2/2)$ introduced in [49] and the scalar product-based similarity $\varepsilon(\boldsymbol{v}_i, \boldsymbol{v}_j) = \max(\boldsymbol{v}_i^T \boldsymbol{v}_j, 0)^3$ adopted in [20]. We also set k to different values to investigate its influence. Results are illustrated in Figure 3. We empirically observe that results with cosine similarity are consistently better than those with the other two alternatives. We attribute the advantages of cosine similarity to the adopted FC layer-based classifier, where the cosine similarities between features and weights of the classifier dominate category predictions. The results of A^{2}LP with affinity matrix constructed by the cosine similarity are stable under a wide range of k (i.e., $5 \sim 30$). Results with the full affinity matrix (i.e., $k = n$) are generally lower than that with the nearest neighbor graph. We empirically set $k = 20$ in the experiments for the Office-31 and ImageCLEF-DA datasets, and set $k = 100$ for the VisDA-2017 dataset, where the number of instances is considerably large.

A^{2}LP on UDA and SSL In this section, we observe the behaviors of the A^{2}LP on UDA and SSL tasks. The goal of the experiment is to observe the results with augmented virtual instances in LP. For the labeled data, we randomly
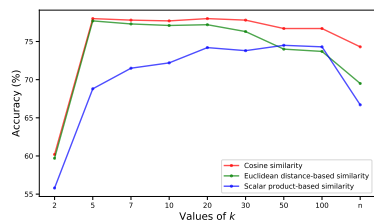
Fig. 3: An illustration of the accuracy (%) of A^{2}LP with affinity matrix constructed with different similarity measurements and different values of k. Results are reported on the $\mathbf{C} \rightarrow \mathbf{P}$ task of the ImageCLEF-DA dataset based on a 50-layer ResNet. When $k = n$, we construct the full affinity matrix as in [49]. Please refer to Sec. 5.1 for the definitions of similarities.
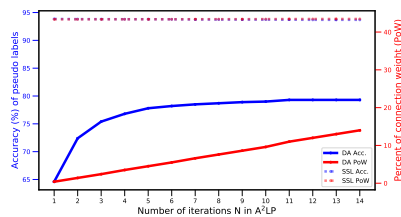

Fig. 4: An illustration of the accuracy (%) of pseudo labels of unlabeled instances (left y-axis) and the percent (%) of connection weight (PoW) of the same class pairs between labeled and unlabeled data (right y-axis) of our proposed A^{2}LP on the tasks of SSL and UDA. The A^{2}LP degenerates to LP [48] when the number of iteration N is set to 1. Please refer to Section 5.1 for the detailed settings.

sample 1000 instances per class in the synthetic image set of the VisDA-2017 dataset. For the SSL task, we randomly sample another 1000 instances per class in the synthetic image set of the VisDA-2017 dataset as the unlabeled data, whereas 1000 instances are sampled randomly in each class of the real image set to construct unlabeled data in the UDA task. We denote the constructed UDA task as VisDA-2017-Small for ease of use. The mean prediction accuracy of all unlabeled instances is reported. To give insights of the different results, we illustrate the percent of connection weight (PoW) of the same class pairs between labeled and unlabeled data in the constructed k-nearest neighbor graph using ground-truth labels of unlabeled data, which is calculated as: $PoW = \frac{W_{lu}}{W_{all}}$, where W_{lu} is the similarities sum of connections of the same class pairs between labeled and unlabeled data in the affinity matrix $\mathbf{A}$ and $W_{all} = \sum_{i,j} \boldsymbol{a}_{ij}$.

The results are illustrated in Fig. 4. In the UDA task, the initial PoW (i.e., N=1) is too low to enable the labels of labeled data propagate to all the unlabeled target data. As the A^{2}LP proceeds, the labeled data are augmented with virtual instances with true labels, whose neighbors involve unlabeled instances sharing the same label. Thus the PoW increases, leading to more accurate predictions of unlabeled data. In the SSL task, cluster centers of labeled and unlabeled data are positioned to be close and (statistically) in relatively dense population areas, since all data follow the same distribution. Instances close to cluster centers, including the k nearest neighbors of virtual instances, are expected to be classified correctly by the LP algorithm, leading to unchanged results as the A^{2}LP proceeds. These observations corroborate the **Proposition** 1 and verify the efficacy of our proposed virtual instances generation strategy for UDA.

Robustness To Noise We investigate the influence of the noise level of label predictions on the results of A²LP. As illustrated in Table 1, the A²LP is robust to the label noise. Specifically, as the noise level increases, results of A2LP degrade and are worse than that of the vanilla LP when the noise level is larger than ∼60%.

Noise level (%)	0	10	30	50	60	70	80	100	Vanilla LP
Acc. (%) of A²LP	92.8	92.8	92.3	91.8	91.0	90.7	90.3	90.0	91.2

Table 1: Results of A²LP with different noise levels of initial label predictions on the $\mathbf{P} \rightarrow \mathbf{C}$ task of the ImageCLEF-DA dataset. We replace the initial label predictions from the LP (i.e., the Line 7 of the Algorithm 1) with a manually defined setting, where the noise level of L% indicates that the virtual instances (i.e., Equ. (9)) are calculated with unlabeled target data, L% of which are assigned with random and wrong pseudo labels. Note that we set N=2 (cf. Line 3 of Algorithm 1) here.

A²LP variant We propose a degenerated variant of A²LP by representing the entire labeled source data with several representative surrogate instances in the A²LP process, which can largely alleviate the computation cost of the LP algorithm. More specifically, we replace the features of source data $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_l\}$ with K source category centers $\{\boldsymbol{v}_k = \sum_{i=1}^{l} \frac{\mathbb{1}(k=y_i)\boldsymbol{v}_i}{\sum_{j=1}^{l} \mathbb{1}(k=y_j)}\}_{k=1}^{K}$ with category labels $\{1, \cdots, K\}$ (only the Line 2 of the Algorithm 1 is updated accordingly). As illustrated in Table 2, the result of A²LP variant is slightly lower than that of the A²LP on the VisDA-2017-Small task. Note that we only adopt the A²LP variant in tasks involving the entire VisDA-2017 dataset unless otherwise specified.

Table 2: Comparison between the A²LP and its degenerated variant on the VisDA-2017-Small task based on a 50-layer ResNet.

Methods	A²LP	A²LP variant
Acc. (%)	79.3	77.9

Table 3: Illustration of effects of the entropy-based instance weights (9) in A²LP based on a 50-layer ResNet.

Methods	A→W	W→A
A²LP	87.7	75.9
A²LP (w_i=1, $\forall i$)	87.4	75.4

Effects of instance weighting in A²LP We investigate the effects of entropy-based instance weights in reliable virtual instances generation (9) of A²LP in this section. As illustrated in Table 3, A²LP improves over A²LP (w_i=1, $\forall i$), where all unlabeled instances are weighted equally, supporting that instances of low entropy are more confident in terms of their predicted labels.

Table 4: Results on the Office31 dataset [34] (ResNet-50).

Methods	A → W	D → W	W → D	A → D	D → A	W → A	Avg.
Source Only	68.4±0.2	96.7±0.1	99.3±0.1	68.9±0.2	62.5±0.3	60.7±0.3	76.1
DAN [27]	80.5±0.4	97.1±0.2	99.6±0.1	78.6±0.2	63.6±0.3	62.8±0.2	80.4
DANN [12]	82.0±0.4	96.9±0.2	99.1±0.1	79.7±0.4	68.2±0.4	67.4±0.5	82.2
CDAN+E [28]	94.1±0.1	98.6±0.1	**100.0**±.0	92.9±0.2	71.0±0.3	69.3±0.3	87.7
SymNets [47]	90.8±0.1	98.8±0.3	**100.0**±.0	93.9±0.5	74.6±0.6	72.5±0.5	88.4
DADA [40]	92.3±0.1	99.2±0.1	100.0±0.0	93.9 ±0.2	74.4±0.1	74.2±0.1	89.0
CAN [22]	**94.5**±0.3	**99.1**±0.2	99.8±0.2	95.0±0.3	78.0±0.3	77.0±0.3	90.6
LP	81.1	96.8	99.0	82.3	71.6	73.1	84.0
A^{2}LP (ours)	87.7	98.1	99.0	87.8	75.8	75.9	87.4
MSTN (reproduced)	92.7±0.5	98.5±0.2	99.8±0.2	89.9±0.3	74.6±0.3	75.2±0.5	88.5
empowered by A^{2}LP	93.1±0.2	98.5±0.1	99.8±0.2	94.0±0.2	76.5±0.3	76.7±0.3	89.8
CAN (reproduced)	94.0±0.5	98.5±0.1	99.7±0.1	94.8±0.4	**78.1**±0.2	76.7±0.3	90.3
empowered by A^{2}LP	93.4±0.3	98.8±0.1	**100.0**±.0	**96.1**±0.1	**78.1**±0.1	**77.6**±0.1	**90.7**

Table 5: Results on the ImageCLEF-DA dataset [1] (ResNet-50).

Methods	I → P	P → I	I → C	C → I	C → P	P → C	Avg.
Source Only	74.8±0.3	83.9±0.1	91.5±0.3	78.0±0.2	65.5±0.3	91.2±0.3	80.7
DAN [27]	74.5±0.4	82.2±0.2	92.8±0.2	86.3±0.4	69.2±0.4	89.8±0.4	82.5
DANN [12]	75.0±0.6	86.0±0.3	96.2±0.4	87.0±0.5	74.3±0.5	91.5±0.6	85.0
CDAN+E [28]	77.7±0.3	90.7±0.2	**97.7**±0.3	91.3±0.3	74.2±0.2	94.3±0.3	87.7
SymNets [47]	**80.2**±0.3	93.6±0.2	97.0±0.3	**93.4**±0.3	78.7±0.3	96.4±0.1	89.9
LP	77.1	89.2	93.0	87.5	69.8	91.2	84.6
A^{2}LP (ours)	79.3	91.8	96.3	91.7	78.1	96.0	88.9
MSTN (reproduced)	78.3±0.2	92.5±0.3	96.5±0.2	91.1±0.1	76.3±0.3	94.6±0.4	88.2
empowered by A^{2}LP	79.6±0.3	92.7±0.3	96.7±0.1	92.5±0.2	78.9±0.2	96.0±0.1	89.4
CAN (reproduced)	78.5±0.3	93.0±0.3	97.3±0.2	91.0±0.3	77.2±0.2	**97.0**±0.2	89.0
empowered by A^{2}LP	79.8±0.2	**94.3**±0.3	**97.7**±0.2	93.0±0.3	**79.9**±0.1	96.9±0.2	**90.3**

5.2   Results

We report the classification results on the Office-31 [34], ImageCLEF-DA [1], and VisDA-2017 [33] datasets in Table 4, Table 5, and Table 6, respectively. Results of other methods are either directly reported from their original papers if available or quoted from [28, 24]. Compared to classical methods [12, 27] aiming at domain-invariant feature learning, the vanilla LP generally achieves better results via the graph-based SSL principle, certifying the efficacy of the SSL principles in UDA tasks. Our A^{2}LP improves over the LP significantly on all three UDA benchmarks, justifying the efficacy of the introduction of virtual instances for UDA. Additionally, we reproduce the state-of-the-art UDA methods of Moving Semantic Transfer Network (MSTN) [44] and Contrastive Adaptation Network (CAN) [22] with the released codes [6]; by replacing the pseudo label generators

---

[6] https://github.com/Mid-Push/Moving-Semantic-Transfer-Network
  https://github.com/kgl-prml

of MSTN and CAN with our A^2LP, we improve their results noticeably and achieve the new state of the art, testifying the effectiveness of the combination of A^2LP and domain-invariant feature learning.

Table 6: Results on the VisDA-2017 dataset. The A^2LP reported is the degenerated variant detailed in Sec. 5.1. Full results are presented in the appendices.

Methods	Acc. Based on a ResNet50	Acc. Based on a ResNet101
Source Only	45.6	50.8
DAN [27]	53.0	61.1
DANN [12]	55.0	57.4
MCD [36]	–	71.9
CDAN+E [28]	70.0	–
LPJT [25]	–	74.0
DADA [40]	–	79.8
Lee *et al.* [24]	76.2	81.5
CAN [22]	–	87.2
LP	69.8	73.9
A^2LP (ours)	78.7	82.7
MSTN (reproduced)	71.9	75.2
empowered by A^2LP	81.5	83.7
CAN (reproduced)	85.6	87.2
empowered by A^2LP	**86.5**	**87.6**

6   Conclusion

Motivated by the relatedness of problem definitions between UDA and SSL, we study the use of SSL principles in UDA, especially the graph-based LP algorithm. We analyze the conditions of affinity graph/matrix to achieve better propagation of true labels to unlabeled instances, and accordingly propose a new algorithm of A^2LP, which potentially improves LP via generation of unlabeled virtual instances. An empirical scheme of virtual instance generation is particularly proposed for UDA via a weighted combination of unlabeled target instances. By iteratively using A^2LP to get high-quality pseudo labels of target instances and learning domain-invariant features involving the obtained pseudo-labeled target instances, new state of the art is achieved on three datasets, confirming the value of further investigating SSL techniques for UDA problems.

References

1. Imageclef-da dataset. http://imageclef.org/2014/adaptation/
2. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: International Conference on Computational Learning Theory. pp. 624–638. Springer (2004)
3. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Machine learning **79**(1-2), 151–175 (2010)
4. Ben-David, S., Blitzer, J., Crammer, K., Pereira, F.: Analysis of representations for domain adaptation. In: Advances in neural information processing systems. pp. 137–144 (2007)
5. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. The MIT Press (2006). https://doi.org/10.7551/mitpress/9780262033589.001.0001, https://doi.org/10.7551/mitpress/9780262033589.001.0001
6. Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: AISTATS. vol. 2005, pp. 57–64. Citeseer (2005)
7. Delalleau, O., Bengio, Y., Roux, N.L.: Efficient non-parametric function induction in semi-supervised learning. In: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (2005), http://www.gatsby.ucl.ac.uk/aistats/fullpapers/204.pdf
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
9. Ding, Z., Li, S., Shao, M., Fu, Y.: Graph adaptive knowledge transfer for unsupervised domain adaptation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 37–52 (2018)
10. Dong, W., Moses, C., Li, K.: Efficient k-nearest neighbor graph construction for generic similarity measures. In: Proceedings of the 20th international conference on World wide web. pp. 577–586 (2011)
11. French, G., Mackiewicz, M., Fisher, M.: Self-ensembling for visual domain adaptation. In: International Conference on Learning Representations (2018)
12. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. The Journal of Machine Learning Research **17**(1), 2096–2030 (2016)
13. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: Advances in Neural Information Processing Systems 17. pp. 529–536 (2005)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
15. He, R., Lee, W.S., Ng, H.T., Dahlmeier, D.: Adaptive semi-supervised learning for cross-domain sentiment classification. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 3467–3476 (2018)
16. Hestenes, M.R., Stiefel, E., et al.: Methods of conjugate gradients for solving linear systems. Journal of research of the National Bureau of Standards **49**(6), 409–436 (1952)
17. Hou, C.A., Tsai, Y.H.H., Yeh, Y.R., Wang, Y.C.F.: Unsupervised domain adaptation with label and structural consistency. IEEE Transactions on Image Processing **25**(12), 5552–5562 (2016)

18. Hui Tang, K.C., Jia, K.: Unsupervised domain adaptation via structurally regularized deep clustering. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
19. Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Label propagation for deep semi-supervised learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5070–5079 (2019)
20. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2077–2086 (2017)
21. Joachims, T.: Transductive learning via spectral graph partitioning. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03). pp. 290–297 (2003)
22. Kang, G., Jiang, L., Yang, Y., Hauptmann, A.G.: Contrastive adaptation network for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4893–4902 (2019)
23. Kumar, A., Sattigeri, P., Wadhawan, K., Karlinsky, L., Feris, R., Freeman, B., Wornell, G.: Co-regularized alignment for unsupervised domain adaptation. In: Advances in Neural Information Processing Systems. pp. 9345–9356 (2018)
24. Lee, S., Kim, D., Kim, N., Jeong, S.G.: Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 91–100 (2019)
25. Li, J., Jing, M., Lu, K., Zhu, L., Shen, H.T.: Locality preserving joint transfer for domain adaptation. IEEE Transactions on Image Processing **28**(12), 6103–6115 (2019)
26. Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S.J., Yang, Y.: Learning to propagate labels: Transductive propagation network for few-shot learning. arXiv preprint arXiv:1805.10002 (2018)
27. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37. pp. 97–105. ICML'15, JMLR.org (2015), http://dl.acm.org/citation.cfm?id=3045118.3045130
28. Long, M., Cao, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: Advances in Neural Information Processing Systems. pp. 1640–1650 (2018)
29. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Unsupervised domain adaptation with residual transfer networks. In: Advances in Neural Information Processing Systems. pp. 136–144 (2016)
30. Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: Learning bounds and algorithms. In: 22nd Conference on Learning Theory, COLT 2009 (2009)
31. Miyato, T., Maeda, S.i., Koyama, M., Ishii, S.: Virtual adversarial training: a regularization method for supervised and semi-supervised learning. IEEE transactions on pattern analysis and machine intelligence **41**(8), 1979–1993 (2018)
32. Pan, S.J., Yang, Q., et al.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering **22**(10), 1345–1359 (2010)
33. Peng, X., Usman, B., Kaushik, N., Hoffman, J., Wang, D., Saenko, K.: Visda: The visual domain adaptation challenge. arXiv preprint arXiv:1710.06924 (2017)
34. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: European conference on computer vision. pp. 213–226. Springer (2010)

35. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. arXiv preprint arXiv:1702.08400 (2017)
36. Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3723–3732 (2018)
37. Shen, J., Qu, Y., Zhang, W., Yu, Y.: Wasserstein distance guided representation learning for domain adaptation. In: AAAI. pp. 4058–4065 (2018)
38. Shu, R., Bui, H.H., Narui, H., Ermon, S.: A dirt-t approach to unsupervised domain adaptation. arXiv preprint arXiv:1802.08735 (2018)
39. Szummer, M., Jaakkola, T.: Partially labeled classification with markov random walks. In: Advances in neural information processing systems. pp. 945–952 (2002)
40. Tang, H., Jia, K.: Discriminative adversarial domain adaptation. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020. pp. 5940–5947. AAAI Press (2020)
41. Tarvainen, A., Valpola, H.: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In: Advances in neural information processing systems. pp. 1195–1204 (2017)
42. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4068–4076 (2015)
43. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014)
44. Xie, S., Zheng, Z., Chen, L., Chen, C.: Learning semantic representations for unsupervised domain adaptation. In: International Conference on Machine Learning. pp. 5419–5428 (2018)
45. Yan, H., Ding, Y., Li, P., Wang, Q., Xu, Y., Zuo, W.: Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). vol. 3 (2017)
46. Zhang, Y., Deng, B., Tang, H., Zhang, L., Jia, K.: Unsupervised multi-class domain adaptation: Theory, algorithms, and practice. CoRR **abs/2002.08681** (2020)
47. Zhang, Y., Tang, H., Jia, K., Tan, M.: Domain-symmetric networks for adversarial domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5031–5040 (2019)
48. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Advances in neural information processing systems. pp. 321–328 (2004)
49. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Thrun, S., Saul, L.K., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems 16. pp. 321–328 (2004)
50. Zhou, Z.H., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. IEEE Transactions on knowledge and Data Engineering **17**(11), 1529–1541 (2005)
51. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Tech. rep., School of Computer Science, Carnegie Mellon University (2002)
52. Zhu, X., Ghahramani, Z., Lafferty, J.D.: Semi-supervised learning using gaussian fields and harmonic functions. In: Proceedings of the 20th International conference on Machine learning (ICML-03). pp. 912–919 (2003)
53. Zhu, X., Lafferty, J., Rosenfeld, R.: Semi-supervised learning with graphs. Ph.D. thesis, Carnegie Mellon University, language technologies institute, school of . . . (2005)