Pattern Recognition 112 (2021) 107763

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Compact class-conditional domain invariant learning for multi-class domain adaptation

Woojin Lee, Hoki Kim, Jaewook Lee*

Department of Industrial Engineering Seoul National University, Seoul National University, San 56-1, Sillim-dong, Gwanak-gu, Seoul 151-742, Republic of Korea

ARTICLE INFO

Article history:
Received 20 August 2019
Revised 26 September 2020
Accepted 25 November 2020
Available online 6 December 2020

Keywords:
Domain adaptation
Generalization bound
Class-conditional domain invariant learning
PAC learning complexity
Transfer Learning

ABSTRACT

Neural network-based models have recently shown excellent performance in various kinds of tasks. However, a large amount of labeled data is required to train deep networks, and the cost of gathering labeled training data for every kind of domain is prohibitively expensive. Domain adaptation tries to solve this problem by transferring knowledge from labeled source domain data to unlabeled target domain data. Previous research tried to learn domain-invariant features of source and target domains to address this problem, and this approach has been used as a key concept in various methods. However, domain-invariant features do not mean that a classifier trained on source data can be directly applied to target data because it does not guarantee that data distribution of the same classes will be aligned across two domains. In this paper, we present novel generalization upper bounds for domain adaptation that motivates the need for class-conditional domain invariant learning. Based on this theoretical framework, we then propose a class-conditional domain invariant learning method that can learn a feature space in which features in the same class are expected to be mapped nearby. We empirically experimented that our model showed state-of-the-art performance on standard datasets and showed effectiveness by visualization of latent space.

1. Introduction

Recently deep learning-based artificial intelligence has demonstrated excellent performance in computer vision, natural language processing, and a variety of other applications. State-of-the-art models in classification, regression, unsupervised learning, and reinforcement learning sometimes show performance even better than that of humans. However, deep learning-based artificial intelligence models require large-scale datasets for training the model, and the cost of gathering labeled data for training is an obstacle for the real-world application of deep learning.

Machine learning assumes that training data and test data are from equal distributions so that we can train a model on training data and implement a real-world application on unlabeled test data. In the real world, however, it is difficult to generate enough labeled data similar to unlabeled real-world test data.

In contrast to machine learning models, humans can transfer knowledge across different tasks, generalize their knowledge among different domains, and learn from a few examples. Inspired by human abilities, transfer learning uses knowledge of labeled training data to improve performance on other data [1].

Domain adaptation is an aspect of transfer learning which attempts to learn a model by using massively available labeled source domain which performs well on target data. Domain adaptation assumes that the two domains are defined for the same task and that the source and target domains are similar but in different distributions.

There have been various approaches related to domain adaptation [2–10]. There was a theoretical analysis that the risk of the target domain can be bounded by the risk of source data and the discrepancy between two domains [2]. Based on this theory, some attempted to learn a transferable representation by reducing the Maximum Mean Discrepancy (MMD) between the source and target domains [4–7]. Another approach assigned an importance weight to each instance of target data [3,9].

Recent trends in domain adaptation, inspired by the generative adversarial network (GAN), are attempting to learn transferable representations that are indistinguishable from the features of the source and target domains using minimax games [11–13]. By using this approach, we can learn a representation that cannot be

* Corresponding author. Department of Industrial Engineering, Seoul National University, San 56-1, Sillim-dong, Gwanak-gu, Seoul 151-742, Republic of Korea.

E-mail addresses: wj926@snu.ac.kr (W. Lee), ghrl9613@snu.ac.kr (H. Kim), jaewook@snu.ac.kr (J. Lee).

https://doi.org/10.1016/j.patcog.2020.107763
0031-3203/© 2020 Elsevier Ltd. All rights reserved.

distinguished by the domain it came from. Then, by using a classifier trained on source data features, we can apply the classifier directly on a hidden representation of target data.

However, a domain invariant representation-based approach does not take into account the class label information of the source and target domains, focusing only on their indistinguishable global distribution. Therefore, there is no guarantee that embedded features of source and target domains with the same label, will be distributed similarly in an indistinguishable representation. If so, there will be large discrepancies in the decision boundaries of the ground truth classifiers of the source and target domains, so a classifier optimized for the source domain sample may not fit well with the target sample. Therefore, if we can learn a class-conditional representation that true classifiers (ground-truth classifiers) of the source and target samples can be similar to each other, the source domain-based classifier will perform better in the target domain.

Some of the existing methods focused on this issue of learning class-conditional distribution, i.e. joint alignment. Most of these methods [4,14–18] use pseudo label induced by label classifier and minimized the difference between data means between source and target data. This means the matching based approach showed prominent performance in class conditional domain adaptation problems.

Fig. 1 motivates the need for compact class-conditional domain invariant representations. An example of a hidden representation when a non-domain adaptation method is applied directly to target data is shown in Fig. 1a. Because the two domains are embedded separately in the latent space, the model degrades in the target domain. Comparatively, domain invariant representation-based methods [10–13] make the classifier work better in the target domain, as shown in Fig. 1b. However, we still can see that domain invariant learning is not enough to tailor the classifier to the target, because it does not take into account the class conditional distribution of the source and target domains. Fig. 1c shows that learning a compact class-conditional distribution on both source and target domains can help an optimized classifier in the source domain to generalize well in the target domains.

However, previous pseudo label induced mean matching methods heavily depends on label classifier to learn class conditional distribution. If the label classifier does not provide an appropriate decision boundary for target samples, it may have the problem of false pseudo labels and lead to incorrect class conditional learning. In addition, there was not much theoretical background on the need for class conditional learning.

To this end, this paper presents a novel generalization bound for domain adaptation that supports the need for the class-conditional compact representation. Based on this theoretical framework, we develop a novel domain adaptation algorithm to learn a compact distribution of class-conditional domain invariant representation. The method introduces a new loss function that makes embedded features of source and target domains with the same label be similarly distributed in an indistinguishable representation and avoids the difficulty of using the class-conditional distribution information of the target domain. The soft assignment of each target sample in latent space is calculated using similarity with the center of the labeled source domain and it is interpreted as the probability of belonging to each label. We then optimize the encoder to enhance prediction, so that each sample can be included more compactly in the closest centroid as shown in Fig. 1c.

Compared to previous class conditional learning methods, our method has two advantages. First, our method does not rely on label classifier, since it only considers the distribution of each sample in latent space. Second, while previous methods focus on matching the mean of the source samples and the pseudo labeled target samples, our method focuses on every individual sample to learn class-conditional distribution.

We can find the advantage of our method over the previous pseudo labeling base method in Fig. 2. In this example, the goal of class conditional learning is to distribute the red sample near the red center and the blue sample to the blue center. The central part of each figure highlights the advantage of our model. In the center of the left figure, in the case when the classifier misclassifies target samples (blue samples in the red area), the previous methods make the examples with false pseudo label moves to the opposite center which is a completely wrong direction. However, in the right figure, since our method relies on the distribution of each sample, we can find that samples in the area in the correct direction. Moreover, while the previous method assigns equal gradients to the samples that are in the same decision area, our method treated each sample differently considering their distributions. Based on this advantage, our suggested method can successfully learn a class conditional distribution and achieve superior results. Detailed settings of these examples and analysis are in Section 5.1.

To evaluate the proposed model, we first perform various image classification tasks for digit recognition (MNIST, MNIST-M, USPS, and SVHN. Then we use Amazon review data for sentimental classification. In classical domain adaptation experiments, our proposed model showed superior performance compared to the previous domain invariant models. The visualization of the hidden representation also showed that the model actually learns a compact representation that the source domain-based classifier can be better applied to the target domain. Experiments with large data sets such as Office, Office-home, Office-Caltech, and VISDA show that our method outperforms any previous state-of-the-art model.

In summary, we list the key contributions of this paper as follows:

1. We propose a novel domain adaptation method called compact class-conditional domain invariant learning. Compared to the previous domain-invariant methods, our method aligns class-conditional distribution across two domains. Our method is simple and easy to apply to previous domain adaptation models to enhance class conditional alignment.
2. We provide a theoretical framework to support class-conditional invariant learning by presenting a new generalization upper bound for domain adaptation.
3. Experimental results on classical and large dataset show that the proposed method performs state-of-the-art classification results. In addition, feature visualization shows that our method forces compact conditional distribution alignment between two distributions.

2. Related work

There are two main approaches to solving the problems of domain adaptation: the instance based approach and the feature based approach. The first approach focuses on training a classifier that can consider the difference between the two domains. This approach tries to revise the training of the classifier by domain adaptation terms in the loss function of the classifier. Sugiyama et al. [3] used importance weighing $p_i = x_i^t / x_i^s$ considering the distribution of source and target domains. However, since recent studies in machine learning are mainly based on representation learning and it handles unstructured data such as images or text data, recent researches focus on feature based domain adaptation methods.

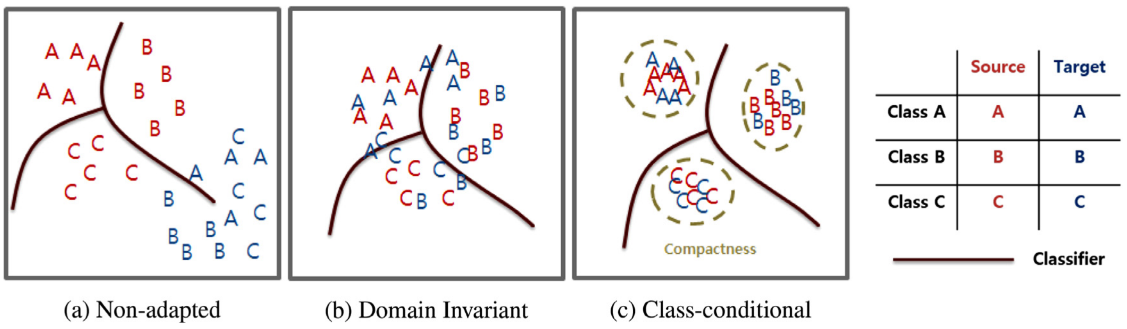


Fig. 1. Comparison of previous methods and proposed domain adaptation method in toy example where alphabet A,B, and C refer to each labels and colors represent domains. Fig. 1a: Distribution of hidden representation when there is with no domain adaptation methods. Fig. 1b: Previous domain invariant methods. Fig. 1c: Hidden feature of our proposed method that attempts to learn class-conditional distribution. It has compact distribution conditional to the labels.

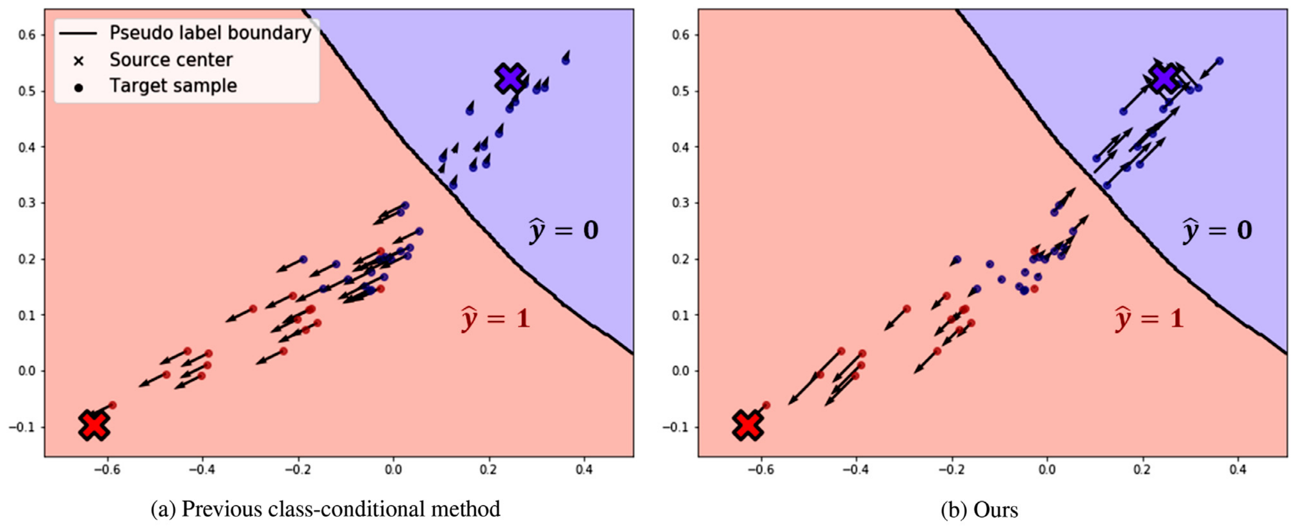


Fig. 2. Comparison previous mean matching methods and our methods in toy example. $\times$ refers to center (μ) of source samples and $\cdot$ means each target samples (x^t), while red and blue colors denote their true labels (y). Black line represents classification boundary, so the red and blue areas denote the pseudo labels, $\hat{y} = 0$ and $\hat{y} = 1$, respectively. Black arrow represents gradient of each samples induced by class conditional loss function. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Feature based domain adaptation tries to find a transferable feature in source and target domains. A transferable feature makes the data from the source and target domains transferable in feature space $\mathcal{Z}$ so that a task specific classifier can be applied to both source and target domains. Prior approaches tried to match the distribution means in the reproducing kernel Hilbert space [4]. In sentimental applications of domain adaptation, Blitzer et al. [19] suggested structural correspondence learning which considers mutual information of words for domain adaptation. There was an approach that used batch normalization for matching the features [20] and group sparse representation [21].

Recent trends of domain adaptation use deep learning based approaches for transferable feature extraction. Glorot et al. [22] trained a deep auto-encoder to extract common features in both source and target domains. Then trained a classifier with the features on the labeled source domain and applied the classifier on the unlabeled target data. Chen et al. [23] extended this approach to marginalized denoising auto-encoders. Recently, there have been approaches that used the idea of memory network [24] and graph convolutional network [25] on domain adaptation problems.

Based on the idea of GAN, Ganin et al. [11] suggested Domain Adversarial Neural Networks (DANN). This model tries to extract a common feature that minimizes $\mathcal{H}$-divergence derived by Ben-David et al. [2]. Minimizing $\mathcal{H}$-divergence makes data points of source and target domain in feature space $\mathcal{Z}$ indistinguishable. DANN uses a gradient reversal layer in a neural network to build an indistinguishable representation function.

Recently many researches have focused on finding an indistinguishable feature by using neural networks. Bousmalis et al. [12] suggested Domain Separation Networks (DSN) which extract domain invariant feature as well as domain specific feature. DSN used a private auto-encoder and a shared auto-encoder to source and target domains, and used similarity and difference losses to extract common features between two domains. Tzeng et al. [13] proposed Adversarial Discriminative Domain Adaptation (ADDA) which uses different mapping functions for source and target domains and GAN loss for learning indistinguishable features. Recently, Wu et al. [25] proposed a new model with a graph convolutional network (GCN) to reveal the geometric information of two different domains. Through GCN, they extracted the source and target feature and reduced domain distribution matching loss with the kernel method.

However, unsupervised learning for extracting transferable features [11–13,25] only enforces global domain statistics without using the information of conditional distribution on class labels. Recently, there was a research that theoretically analyzed why learning invariant representation is not sufficient in domain adaptation problems [26]. This paper showed a counterexample where domain invariant features lead to increasing error on the source and target domains. Moreover suggested a new information-theoretic lower bound that shows there exists a trade-off between invariant representation and small joint error.

Recent approaches to domain adaptation have focused mostly on aligning the joint distributions of source and target domains. Long et al. [4] tries to deal with this by aligning the class conditional representation by the pseudo label in the target data by using MMD and pseudo labels Zhang et al. [15] also used a similar approach with [4], by learning two coupled projections with source and target domains. There were also researches that tried to matching both marginal and conditional distributions [14] and balancing between those two terms [16,18]. These pseudo labels and MMD based approaches have shown great success in learning class conditional distributions. However, since these methods need to solve the eigenvalue decomposition problem, it requires a high computational cost. In addition, it has the problem of false pseudo-labels, which can negatively impact performance. PACET [27] proposes using a local triplet-wise instance-to-center margin to align the joint distribution of both domains. Concurrent work by Wang and Breckon [28] used iterative learning with selective pseudo labeling to deviate from the problem of false pseudo labels during the beginning of the training. This is similar to our work in the sense that it assigns a pseudo label by using the distance between target samples and the class prototypes. However, while our methods use the probability of assignment, [28] takes only the maximum value and assigns a pseudo label for each target sample. In this respect, [28] did not consider the distribution of each sample in the latent space. In addition, similar to other MMD based methods [4,14,15], Wang and Breckon [28] require to solve the eigenvalue decomposition problem These approaches are not a deep learning-based model, making it difficult to construct an end-to-end model on large data sets and requires an add-on feature extractors such as SURF of DeCAF6 [29].

There are also deep learning based methods for aligning the joint distribution of the two domains. JAN(Joint Adaptation Network) tried to reduce moment matching between the feature of source and target data by using kernel and tensor Product [30]. In particular, they consider the joint distribution of data by merging the output of the middle layers with the features of the last layer. However, they did not use the label information to calculate the JMMD, which can be considered a lack of alignment. Unlike other methods that focus on additional constraints on adversarial training, Long et al. [31] proposed CDAN (Conditional Domain Adaptation Network) that uses multiplicative interactions between features and predictions in adversarial training and changes the adversarial training by adding prediction information. Saito et al. [32] proposed to consider task-specific decision boundaries for domain adaptation issues. They attempted to adjust decision boundaries by maximizing and minimizing discrepancies between two separate classifiers. However, this system requires additional classifiers, and optimization requires three steps to learn the appropriate decision boundaries. Xie et al. [17] used pseudo labels and aligned the centroid of each category between the two domains. To overcome the problem of false pseudo-labels, they suggested moving average semantic transfer loss (MSTN). Similarly, to alleviate the false pseudo-labels problem, Chen et al. [33] proposed a method that sequentially trains a model from easy target images, which are similar to the source images, to hard target images, which are different from the source images, considering intra-class variation in the target domain. Another approaches to produce a similar representation encoder with the joint distribution is considering joint labels. Even if the inputs have the same label, if they belong to different domains, they are considered as different labels. For instance, Cicek and Soatto [34] constructed a joint classifier which outputs domain-class label, e.g., source dog, source cat, target dog, and target cat, to extract domain invariant features.

To push further along these unsupervised domain adaptations via joint alignment methods [17,31,32,35], we suggest a novel design of class-conditional invariant learning. While [31,32,35] require specific network structures, such as multiplicative interactions or two separate classifiers or two feature translators, our method does not require any additional networks or specific structure so that it can be applied to any previous domain invariant learning structure. Similar to Xie et al. [17], our method attempts to align the target sample to the centroid of the source domain, taking into account class conditional distribution. However, unlike [17], which matches the centers between the two domains, our method aligns each sample to the source centroid so that it has a much compact distribution near each centroid.

To extract label information for unlabeled target domains, previous studies [17,31,35] used classifier prediction. However, using a classifier prediction as a pseudo label may not be accurate enough because it does not guarantee that the classifier in early epochs works well with the target data. They used entropy condition, moving average, and truly domain invariant features to solve this problem. We suggest using soft assignment of each sample directly in the latent space, instead of using classifier prediction for the pseudo label. In this way, we can focus on learning the proper representation of the latent space without resorting to classifiers.

3. Theoretical framework

3.1. Domain adaptation setting

We introduce an unsupervised learning algorithm that transfers information from a large labeled source domain to an unlabeled target domain. We consider classification tasks where $\mathcal{X}$ is an input space and $\mathcal{Y}$ is an output space. In the domain adaptation settings, we have two distinct distributions over $\mathcal{X}$ called source domain $\mathcal{D}^s$ and target domain $\mathcal{D}^t$ where $S = \{(\mathbf{x}_i^s, y_i^s) \in \mathcal{D}^s \times \mathcal{Y} : i = 1, \ldots, N_s\}$ represents a labeled source data of N_s samples, distributed according to a source density $p_s(\mathbf{x}, y)$, and $\{\mathbf{x}_j^t \in \mathcal{D}^t,\ j = 1, \ldots, N_t\}$ indicates an unlabeled target data of N_t samples, distributed according to a target density $p_t(\mathbf{x})$ with unobserved labels distributed according to $p_t(y|\mathbf{x})$. As usual, we assume that the source and the target domains satisfy the *covariate shift* condition, i.e. $p_s(y|\mathbf{x}) = p_t(y|\mathbf{x})$.

Following the notations of Ben-David et al. [36], given a true target concept or labeling function $c : \mathcal{X} \to [0, 1]$, and given a hypothesis set $\mathcal{H} = \{h : \mathcal{X} \to \{0, 1\}\}$, the generalization error (or risk) of a hypothesis $h \in \mathcal{H}$ in a domain $\mathcal{D}^s$ are defined by

$$\epsilon_{\mathcal{D}^s}(h, c) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^s}[|h(\mathbf{x}) - c(\mathbf{x})|] = \Pr_{\mathbf{x}\sim\mathcal{D}^s}[h(\mathbf{x}) \neq c(\mathbf{x})]$$

For more simplified notation, we use $\epsilon_{\mathcal{D}^s}(h) = \epsilon_{\mathcal{D}^s}(h, c^s)$ and $\epsilon_{\mathcal{D}^t}(h) = \epsilon_{\mathcal{D}^t}(h, c^t)$ where c^t and c^s denote true target functions of source and target domain, respectively. If we let $\hat{\mathcal{D}}_m^s$ and $\hat{\mathcal{D}}_m^t$ are samples of size m each independently drawn according to the distribution $\mathcal{D}^s$ and $\mathcal{D}^t$, respectively, then the empirical error of $h \in \mathcal{H}$ in a domain $\mathcal{D}^s$ is defined by

$$\hat{\epsilon}_{\hat{\mathcal{D}}_m^s}(h, c) = \frac{1}{m} \sum_{\mathbf{x}_i \in \hat{\mathcal{D}}_m^s} 1_{h(\mathbf{x}_i) \neq c(\mathbf{x}_i)}$$

To give a rigorous model of domain adaptation, Ben-David et al. [2] defined the set of disagreement between two hypothesis in $\mathcal{H}$ by the symmetric difference hypothesis space $\mathcal{H}\Delta\mathcal{H} = \{\tilde{h} : \mathcal{X} \to \{0, 1\} | \tilde{h}(\mathbf{x}) = h(\mathbf{x}) \oplus h'(\mathbf{x}),\ h, h' \in \mathcal{H}\}$ where $\oplus$ is the XOR function. The $\mathcal{H}\Delta\mathcal{H}$ divergence measures discrepancy between two different domains and is defined by

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}^s, \mathcal{D}^t) = 2 \sup_{h, h' \in \mathcal{H}} |\Pr_{\mathbf{x}\sim\mathcal{D}^s}[h(\mathbf{x}) \neq h'(\mathbf{x})] - \Pr_{\mathbf{x}\sim\mathcal{D}^t}[h(\mathbf{x}) \neq h'(\mathbf{x})]| \tag{1}$$

$$= 2 \sup_{h,h' \in \mathcal{H}} |\epsilon_{\mathcal{D}^s}(h,h') - \epsilon_{\mathcal{D}^t}(h,h')| \tag{2}$$

They suggested using proxy distance to easily approximate $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}^s, \mathcal{D}^t)$ by the following empirical $\mathcal{H}\Delta\mathcal{H}$ divergence between samples of size m each given by

$$d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{D}}^s_m, \hat{\mathcal{D}}^t_m) = 2\left(1 - \min_{\tilde{h} \in \mathcal{H}\Delta\mathcal{H}} \left[\frac{1}{m}\sum_{\mathbf{x}_i \in \hat{\mathcal{D}}^s_m} 1_{\tilde{h}(\mathbf{x}_i)=0} + \frac{1}{m}\sum_{\mathbf{x}_i \in \hat{\mathcal{D}}^t_m} 1_{\tilde{h}(\mathbf{x}_i)=1}\right]\right), \tag{3}$$

which train a classifier $\tilde{h}(\mathbf{x}) = h(\mathbf{x}) \oplus h'(\mathbf{x})$ that discriminates points between source and target domains where $\tilde{h}$ is labelled with $\tilde{h}(\mathbf{x}_i) = 0$ if $\mathbf{x}_i$ is from a source domain and 1 if $\mathbf{x}_i$ is from a target domain. It is easy to show that if we let $\text{err}(\tilde{h})$ denotes empirical error of classifier $\tilde{h}$ on the task of discriminating points between source and target domain, then the empirical $\mathcal{H}\Delta\mathcal{H}$ divergence becomes

$$d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{D}}^s_m, \hat{\mathcal{D}}^t_m) = 2\left(1 - 2\min_{g \in \mathcal{H}\Delta\mathcal{H}} \text{err}(g)\right) \tag{4}$$

Utilizing these concepts and the PAC learning framework in Mohri et al. [37], Ben-David et al. [2] derived the following generalization error bound on the target risk as: For any $\delta > 0$ with probability at least $1 - \delta$, for every $h \in \mathcal{H}$,

$$\epsilon_{\mathcal{D}^t}(h) \le \epsilon_{\mathcal{D}^s}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{D}}^s_m, \hat{\mathcal{D}}^t_m) + \lambda^* + 4\sqrt{\frac{2\text{VCdim}(\mathcal{H})\log(2m) + \log\frac{2}{\delta}}{m}} \tag{5}$$

where $\text{VCdim}(\mathcal{H})$ is the VC dimension of $\mathcal{H}$ and

$$\lambda^* = \inf_{h \in \mathcal{H}}[\epsilon_{\mathcal{D}^s}(h) + \epsilon_{\mathcal{D}^t}(h)] \tag{6}$$

denotes the combined error of the ideal joint hypothesis h^* that minimizes the combined source and target errors. The first term in the bound is the source error, the second term measures the distance between unlabeled samples of source and target domain, and the third term refers that the bound depends on the λ^* which is the optimal joint risk.

Since minimizing the second term of (5) lowers the generalization bound risk of the target domain in domain adaptation, various researches focused on learning invariant features by using unsupervised learning [11,12,22]. Since labeling function $f : \mathcal{Z} \to \hat{\mathcal{Y}}$ works on feature space $\mathcal{Z}$, previous algorithms tried to find a feature transformation $g : \mathcal{X} \to \mathcal{Z}$ such that the induced source ($\mathcal{D}^s_{\mathcal{Z}}$) and target distributions ($\mathcal{D}^t_{\mathcal{Z}}$) are close. At the same time, these methods also tried to learn a classifier on the feature space $\mathcal{Z}$ that minimizes the first term, i.e. empirical error on the labeled source domain.

3.2. New generalization upper bounds

The generalization error bound of (5) is in general not sufficient to minimize the expected target domain error for domain adaptation problems, since learning the indistinguishable representation of the source and the target domains do not guarantee that label conditional distribution between two domains are similar. Therefore a classifier that was supervised trained by labeled source domain may not show the same performance in the target domain, especially when the true labeling functions $f : \mathcal{Z} \to \mathcal{Y}$ that work for induced source and target distributions are different. Also, it is in practice hard to compute the optimal error λ^* in Eq. (6). As intractable term λ^* exists in the error bound, minimizing only the first two terms is not sufficient for domain adaptation.

In this paper, we extend the theoretical upper bound for binary classification to multi-classification case (with K-classes) and will provide a new generalization upper bound for domain adaptation that does not contain intractable term λ.

Given a true target concept or labeling function $c : \mathcal{X} \to \mathcal{Y}_K$ where $\mathcal{Y}_K = \{1, 2, ..., K\}$ (mono-label case) or $\mathcal{Y}_K = \{0, 1\}^K$ (multi-label case), and given a hypothesis set $\mathcal{H} = \{h : \mathcal{X} \to \mathcal{Y}_K\}$, the generalization error (or risk) of a K-class hypothesis $h \in \mathcal{H}$ in a domain $\mathcal{D}^s$ are defined by

$$\epsilon_{\mathcal{D}^s}(h) = \epsilon_{\mathcal{D}^s}(h, c) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}^s}[\|h(\mathbf{x}) - c(\mathbf{x})\|_K]$$

where $\|\cdot\|_K$ is the Hamming distance that measures the number of different components in two vectors. $\epsilon_{\mathcal{D}^t}(h) = \epsilon_{\mathcal{D}^t}(h, c^t)$ are similarly defined. We define the symmetric difference hypothesis space on $\mathcal{H}$ by $\mathcal{H}\Delta\mathcal{H} = \{g : \mathcal{X} \to \mathcal{Y}_K | g(\mathbf{x}) = h(\mathbf{x}) \oplus h'(\mathbf{x}),\ h, h' \in \mathcal{H}\}$ and the $\mathcal{H}\Delta\mathcal{H}$ divergence measures discrepancy between two different domains by

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}^s, \mathcal{D}^t) = 2 \sup_{h,h' \in \mathcal{H}} |\epsilon_{\mathcal{D}^s}(h,h') - \epsilon_{\mathcal{D}^t}(h,h')| \tag{7}$$

Let $\mathcal{Z}$ refer to a hidden feature space that is shared between both the source and the target domains and is induced from the input space $\mathcal{X}$ by using a representation function $g \in \mathcal{G} = \{g : \mathcal{X} \to \mathcal{Z}\}$ where $g(\mathbf{x}) = \mathbf{z}$ for any random variables $\mathbf{x} \in \mathcal{X}$ and $\mathbf{z} \in \mathcal{Z}$. If we denote $f \in \mathcal{F} = \{f : \mathcal{Z} \to \mathcal{Y}_K\}$ a labelling function that operates in the induced feature space $\mathcal{Z}$, then hypotheses $h \in \mathcal{H} = \{f \circ g : f \in \mathcal{F},\ g \in \mathcal{G}\}$ are formed by the process $\mathcal{X} \xrightarrow{g} \mathcal{Z} \xrightarrow{f} \hat{\mathcal{Y}}_K$ where $h = f \circ g = f(g(\cdot))$. We assume that the true labeling functions in the induced feature space of the source and target domains are f^s and f^t, respectively where $c^s = f^s_g \circ g$ and $c^t = f^t_g \circ g$ for some $g \in \mathcal{G}$.

The following theorem gives a new generalization upper bound on hypothesis space $\mathcal{H}_g = \{f \circ g : f \in \mathcal{F}\}$ stratified for g using combined source and target training data.

Theorem 1. *Let $\langle \mathcal{D}^s, c^s = f^s_g \circ g \rangle$ and $\langle \mathcal{D}^t, c^t = f^t_g \circ g \rangle$ be the source and target domains, respectively, and let $\mathcal{D}^s_{\mathcal{Z}}$ and $\mathcal{D}^t_{\mathcal{Z}}$ be induced two distributions over $\mathcal{Z}$ by a representation function $g : \mathcal{X} \to \mathcal{Z} \in \mathcal{G}$. For any hypothesis $h \in \mathcal{H}_g = \{f \circ g : f \in \mathcal{F}\}$, the following inequality holds:*

$$\epsilon_{\mathcal{D}^t}(f \circ g) < \epsilon_{\mathcal{D}^s}(f \circ g) + \frac{1}{2} d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s, \mathcal{D}^t) + \min\{\mathbb{E}_{\mathbf{z} \sim \mathcal{D}^s_{\mathcal{Z}}}[\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K], \mathbb{E}_{\mathbf{z} \sim \mathcal{D}^t_{\mathcal{Z}}}[\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K]\} \tag{8}$$

Proof. See Appendix. □

As a result, the upper bound of empirical expected target error can be decomposed into three parts: The first term comes from empirical source error. The second term is the $\mathcal{H}_g$-divergence between source and target domain stratified for a representation function g. The third term corresponds to difference of true labeling functions between source and target domain in feature space $\mathcal{Z}$.

We next present empirical generalization bounds in the multi-class domain adaptation setting. Following the notations and definitions of Mohri et al. [37], we first give the definition of the Rademacher complexity that measures the capacity of a hypothesis space by its ability to fit random data.

Definition 1 (Rademacher complexity)**.** For a real-valued function class $\mathcal{H} : \mathcal{X} \to [a, b]$ and a sample $\mathcal{D}^s = \{\mathbf{x}_1, ..., \mathbf{x}_m\}$ generated by a distribution $\mathcal{D}^s$, the empirical Rademacher complexity of $\mathcal{H}$ with respect to S is the random variable

$$\hat{\mathfrak{R}}_{\mathcal{D}^s}(\mathcal{H}) = \mathbb{E}_{\sigma}\left[\sup_{h \in \mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} \sigma_i h(\mathbf{x}_i)\right] \tag{9}$$

where $\boldsymbol{\sigma} = \{\sigma_1, ..., \sigma_m\}$ are independent uniform $\{\pm 1\}$-valued Rademacher random variables. The *Rademacher complexity* of $\mathcal{H}$ is the expectation of the empirical Rademacher complexity over all samples of size m:

$$\mathfrak{R}_m(\mathcal{H}) = \mathbb{E}_{\mathcal{D}^s}[\hat{\mathfrak{R}}_{\mathcal{D}^s}(\mathcal{H})] = \mathbb{E}_{S\sigma}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^{m}\sigma_i h(\mathbf{x}_i)\right] \tag{10}$$

We define $\Lambda_g(\mathcal{F})$ for a representation function $g \in \mathcal{G}$ by

$$\Lambda_g(\mathcal{F}) = \{\mathbf{x} \mapsto 1_{y=f\circ g(\mathbf{x})} : y \in \mathcal{Y}_K,\ f \in \mathcal{F}\}.$$

The label associated to point $\mathbf{x}$ is $f \circ g(\mathbf{x})$ which is the one resulting in the largest score $1_{y=f\circ g(\mathbf{x})}$.

Let $S = \{(\mathbf{x}_i^s, y_i^s) \in \mathcal{D}^s \times \mathcal{Y} : i = 1, \ldots, N_s\}$ represent a labeled source data of N_s samples, and $\{\mathbf{x}_j^t \in \mathcal{D}^t,\ j = 1, \ldots, N_t\}$ indicates an unlabeled target data of N_t samples. The *empirical error loss* for multi-class classification is defined by

$$\hat{\epsilon}_{\mathcal{D}^s}(f\circ g) = \frac{1}{m}\sum_{i=1}^{m} 1_{y_i \neq f\circ g(\mathbf{x}_i)}$$

Now we are ready to present the following empirical risk bound for multi-class domain adaptation for a mono-label $\mathcal{Y}_K$ in the PAC (Probably Approximately Correct) learning framework.

Theorem 2 (Empirical Risk bound for multi-class domain adaptation)**.** *Let $\langle\mathcal{D}^s, c^s = f_g^s\circ g\rangle$ and $\langle\mathcal{D}^t, c^t = f_g^t\circ g\rangle$ be the source and target domains, respectively, and let $\mathcal{D}_{\mathcal{Z}}^s$ and $\mathcal{D}_{\mathcal{Z}}^t$ be induced two distributions over $\mathcal{Z}$ by a representation function $g : \mathcal{X} \to \mathcal{Z} \in \mathcal{G}$. Then, for any $\delta > 0$, with probability at least $1-\delta$, the following multi-class classification generalization bound holds for all hypothesis $h \in \mathcal{H}_g = \{f\circ g : f \in \mathcal{F}\}$:*

$$\begin{aligned}\epsilon_{\mathcal{D}^t}(f\circ g) \le\ & \hat{\epsilon}_{\mathcal{D}^s}(f\circ g) + 4K\mathfrak{R}_m(\Lambda_g(\mathcal{F})) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}} + 2\sqrt{\frac{\log\frac{2}{\delta}}{2m}}\\ &+ \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\hat{\mathcal{D}}_m^s, \hat{\mathcal{D}}_m^t) + 4\sqrt{\frac{d\log\frac{em}{2d}}{m}}\\ &+ \min\{\mathbb{E}_{\mathbf{z}\sim\mathcal{D}_{\mathcal{Z}}^s}[\|f_g^s(\mathbf{z}) - f_g^t(\mathbf{z})\|_K], \mathbb{E}_{\mathbf{z}\sim\mathcal{D}_{\mathcal{Z}}^t}[\|f_g^s(\mathbf{z}) - f_g^t(\mathbf{z})\|_K]\}\end{aligned} \tag{11}$$

where $d = \mathrm{VCdim}(\mathcal{H}_g)$. Furthermore, if a representation function $g : \mathcal{X} \to \mathcal{Z} \in \mathcal{G}$ is chosen in such a way to make its induced two distributions over $\mathcal{Z}$, $\mathcal{D}_{\mathcal{Z}}^s$ and $\mathcal{D}_{\mathcal{Z}}^t$ be identically distributed (or similar to each other), i.e. $\mathcal{D}_{\mathcal{Z}}^s \sim \mathcal{D}_{\mathcal{Z}}^t$, then, for any $\delta > 0$, with probability at least $1-\delta$,

$$\begin{aligned}\epsilon_{\mathcal{D}^t}(f\circ g) \le\ & \hat{\epsilon}_{\mathcal{D}^s}(f\circ g) + 4K\mathfrak{R}_m(\Lambda_g(\mathcal{F}))\\ &+2\sqrt{\frac{\log\frac{1}{\delta}}{2m}} + 2\sqrt{\frac{\log\frac{2}{\delta}}{2m}} + \tfrac{1}{m}\textstyle\sum_{i=1}^{m}[\|f_g^s(\mathbf{z}_i) - f_g^t(\mathbf{z}_i)\|_K]\end{aligned} \tag{12}$$

Proof. See Appendix. □

In summary, the generalization bounds of Theorems 1 and 2 extend the existing bounds to multi-class domain adaptation problems and overcomes limitations of the previous approaches as follows. Firstly, the second term in the margin upper bound of (11) provides a tighter upper bound than that of the theorem 9.2 of [37], since $\mathfrak{R}_m(\Lambda_g(\mathcal{F})) \le \mathfrak{R}_m(\Pi_1(\mathcal{H}))$ where $\Pi_1(\mathcal{H}) := \{\mathbf{x} \mapsto h(\mathbf{x}, y) : y \in \mathcal{Y}_K, h \in \mathcal{H}\}$. Secondly, the last term in the upper bound of (11) or (12) explicitly motivates the need for the supervised class-invariance on the feature space $\mathcal{Z}$ induced from the input space $\mathcal{X}$ by using any domain invariant representation function candidate g.

4. Proposed method

4.1. Proposed model

4.1.1. Motivation

In the empirical risk bound for multi-class domain adaptation in (11) of Theorem 2, the first risk bound (the first to the fourth terms) is the empirical source error which can be minimized by reducing the supervised loss of labeled source data. The second risk bound (the fifth and the sixth terms) is the empirical $\mathcal{H}_g$-divergence which can be minimized by building a domain invariant representation. There were approaches using the gradient reverse layer [11] or GAN loss [13]. However, the third risk bound (the last term) is hard to minimize since we don't have labeled target data. With only unlabeled target data, we cannot in general expect how does the true labeling function f^t look like. We observe, however, finding a feature space $\mathcal{Z}$ that makes true labeling functions f^s and f^t similar can minimize this risk bound.

To find the representation space where true labeling functions of source and target domain are similar, we propose a class-conditional domain invariant method to learn a representation function $g : \mathcal{X} \to \mathcal{Z}$ that tries the true labeling function of target samples, f^t, work similarly to that of source samples, f^s, on the induced feature space $\mathcal{Z}$. To this end, we build an appropriate feature space $\mathcal{Z}$ in which data points with the same label are distributed compactly in either source and target domain in such a way to make target labeling function f^t work well on source data.

Fig. 1 shows a toy example with hidden representations. Fig. 1b shows a domain invariant method where the global distribution of source (blue) and target (red) domain samples are indistinguishable. In contrast, our proposed class-conditional model embeds the distribution of source and target samples of the same class nearby as shown in Fig. 1c. Moreover, since the sample distributions of different classes are separated from each other, the true labeling function f^s and f^t can be similar to each other.

4.1.2. Class-conditional loss function

However, since there is no information about neither f^t or labels of samples of the target domain, the class-conditional distribution of the target domain is intractable. To overcome this obstacle, we used the concept of soft assignment in feature space $\mathcal{Z}$ as in the t-distributed stochastic neighbor embedding which embeds similar objects onto nearby points and dissimilar objects onto distant points. To obtain the information of labels in the target domain, we alternatively used distance-based soft assignment q_{ij} in representation space as an auxiliary label. We calculated the following q_{ij} for each data i to measure the similarity between embedded point z_i and centroid μ_j:

$$q_{ij} = \frac{e^{-\alpha||z_i-\mu_j||^2}}{\sum_k e^{-\alpha||z_i-\mu_k||^2}} \tag{13}$$

where q_{ij} is the probability of assigning sample i to j and μ_j is centroid of source domain samples on each classes in feature space $\mathcal{Z}$.

$$\mu_j = \sum_{i=1}^{N_j} z_i^s / N_j$$

where N_j is the number of source data whose true label is j. Since we are calculating L^2 distance $||z_i - \mu_j||^2$ on high dimensional data, we set α as the hyper parameter to prevent overflow problems in empirical experiments.

In our model z_i are the data points of source and target domains in feature space Z and μ_j is the centroid of data points of the source domain of each label j. Therefore q_{ij} can be interpreted as an alternative auxiliary label of unlabeled data x_i^t based on similarity measure. Since domain adaptation assumes source and target domains are from different but similar distribution, we assumed that soft assignment based on similarity measure reflects real labels of target data.

We expected that if we calculate soft assignment q_{ij} in the ideal jointly aligned representation, the values of q_{ij} would have high

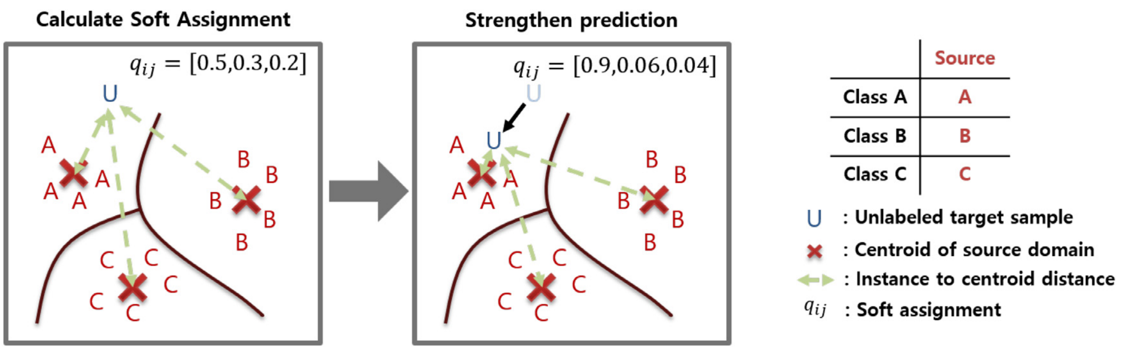

Calculate Soft Assignment
$q_{ij} = [0.5, 0.3, 0.2]$
Strengthen prediction
$q_{ij} = [0.9, 0.06, 0.04]$
Source
Class A A
Class B B
Class C C
U : Unlabeled target sample
: Centroid of source domain
: Instance to centroid distance
q_{ij} : Soft assignment

Fig. 3. Illustration of mechanism of our method. Left figure shows initial embedding of single unlabeled target sample marked as U and right figure shows change of sample embedding after optimization.

confidence of assigning the sample to one centroid, i.e., the probability should be skewed to a single class. It is because the distance between the closest centroid should be much closer than the others. Therefore, we thought if we could learn an embedding that strengthens the prediction of soft assignment, then we can learn a compact representation that similar instances are embedded closely.

To strengthen the prediction, we want to assign unlabeled target data points x_i^t closer to labeled centroid μ_j, so that data points of source data $\mathbf{X}_s = (\mathbf{x}_i^s, y_i^s = L_j)$ and target data $\mathbf{X}_t = (\mathbf{x}_i^t, y_i^t = L_j)$ with the same labels L_j are distributed more compactly. We used the concept of auxiliary target distribution p_{ij}:

$$p_{ij} = \frac{(q_{ij})^n / \sum_i (q_{ij})^n}{\sum_k (q_{ik})^n / \sum_i (q_{ik})^n} \tag{14}$$

where n is the hyper parameter which strengthens the auxiliary target distribution suggested by Xie et al. [38]. We define similarity loss L_s which minimizes KL divergence loss between q_{ij} and p_{ij}. Notice that minimizing KL-divergence in Eq. (14) can be interpreted as self-training as mentioned by Xie et al. [38]. This loss function gives unlabeled target data higher confidence predictions based on its distance as follows.

$$\mathcal{L}_{CCDI} = \mathrm{KL}(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{15}$$

We refer our suggested loss function as CCDI (Compact Class-conditional Domain Invariant) loss function.

Fig. 3 shows how our CCDI loss function works. As shown in the left figure, the embedding of the target sample z_i is not distributed close to any centroid μ_j. However, by optimizing the encoder in the direction of strengthening the assignment prediction, the embedding of the target sample should be located close to its nearest centroid as the right figure. We can find in the figure that as soft assignment q_{ij} has higher confidence in the prediction, the embedding of the sample converges to the nearest centroid. In this paper, we tried to achieve this by minimizing (15).

4.2. Model architecture

The architecture of our suggested model is illustrated in Fig. 4. Our model is composed of one feature encoder $E_g(\mathbf{x}; \theta_g)$ parameterized by θ_g which maps source and target data X_s and X_t into hidden representation z_s and z_t. Then there is a label classifier $C_f(z; \theta_f)$ parameterized by θ_f that predicts task specific label $\hat{y}_i$ of each $\mathbf{x}_i$. There is one more classifier $C_d(z; \theta_d)$ parameterized by θ_d that distinguishes what kind of domain z is from.

Our model is trained by minimizing the following loss function with respect to parameters θ_g, θ_f, and θ_d.

$$\mathcal{L} = \mathcal{L}_T + \lambda_D \mathcal{L}_D + \lambda_{CCDI} \mathcal{L}_{CCDI} \tag{16}$$

The first loss function $\mathcal{L}_T$ is supervised loss function that is applied to labeled source samples. This loss function is related to first to fourth terms of the (12) which implies empirical source error. It is optimized by follows:

$$\min_{\theta_f, \theta_l} \mathcal{L}_T(X_s, E_g, C_f) = -\sum_{i=1}^{N_s} y_i^s \cdot \log \hat{y}_i^s$$

Minimizing loss function $\mathcal{L}_T$ trains feature encoder $E_g(\mathbf{x}; \theta_g)$ and label classifier $C_f(z; \theta_f)$. It forces hidden representation z to better distinguish source data by label. It also trains the label classifier to predict the labels of source data.

The loss function $\mathcal{L}_D$ refers to the loss function suggested by previous domain invariant methods. It reduces empirical $\mathcal{H}$-divergence that is corresponding to the fifth and the sixth term in (12). While any deep learning based domain invariant learning term can be used as $\mathcal{L}_D$, we applied the most widely used adversarial training based methods (DANN [11], CDAN [31]) in our experiment. The minimax optimization of adversarial domain invariant training [11] is as follows:

$$\max_{\theta_g} \min_{\theta_d} \mathcal{L}_D(X_s, X_t, E_g, C_d) = -\sum_{i=1}^{N_s+N_t} d_i \log \hat{d}_i + (1 - d_i) \log(1 - \hat{d}_i)$$

where d_i is domain label for sample i.

$\mathcal{L}_{CCDI}$ is our suggested loss function introduced in (15). This loss function forces the model to learn more compact class-conditional distribution by strengthening the soft assignment q_{ij}. We expect, by learning the compact class-conditional distribution, the true labeling function f^s and f^t can be similar to each other and reduces the seventh term in the suggested empirical risk bound for domain adaptation.

$$\min_{\theta_g} \mathcal{L}_{CCDI}(X_s, X_t, E_g) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

The detailed procedure of the proposed method is presented in Algorithm 1.

5. Experiments

We evaluate our model on three different classification tasks. First, we test the proposed framework on *digit classification* which is representative of the simplest tasks. Simultaneously, we show how our theoretical insight contributes to the improvement of classification performance. Next, we expand our experiment with more complicated *image recognition* datasets. Finally, to generalize our framework, we apply our algorithm to the *sentimental classification*. In each task, we compare the results of our model and several state-of-the-art methods with detailed analysis. The experiments are implemented in the PyTorch.

In order to minimize L_D in Eq. (16), we basically adopt DANN [11] as a domain invariant method. We call this method as CCDI in the experiment section. Further, we propose CCDI* that follows the domain invariant paradigm of CDAN [31]. In each method, we also

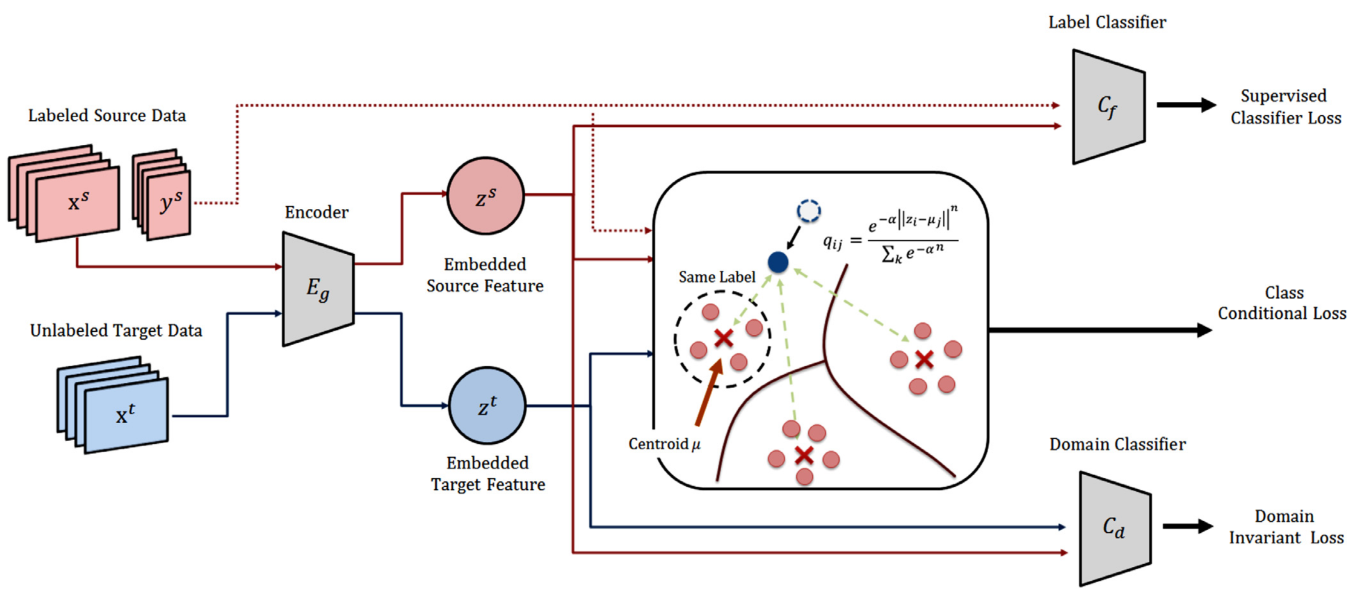


Fig. 4. An overview of our proposed method. We introduce CCDI training framework that could train the encoder E_g to have more compact representation considering label information.

```
Algorithm 1: Compact Class-conditional Domain Invariant (CCDI).
Input  : source images and labels $X_s = \{x_1^s, ..., x_{N_s}^s\}$,
         $Y_s = \{y_1^s, ..., y_{N_s}^s\}$, target images $X_t = \{x_1^t, ..., x_{N_t}^t\}$.
Output: encoder $E_g$, label classifier $C_f$, domain classifier $C_d$.
for 1,...,#epochs do
    $z = \{z^s, z^t\} = E_g(\{x^s, x^t\})$
    for $j = 1, ..., \#classes$ do
        $\mu_j = \sum_{i \in K} z_i^s / |K|$ where $K = \{k | y_k^s = j\}$
    end
    // Task Loss
    $\hat{y}^s = C_f(z^s)$
    $\mathcal{L}_T = -\sum_i y_i^s \cdot \log \hat{y}_i^s$// Domain Invariant Loss
    $\hat{d} = C_d(z)$
    $\mathcal{L}_D = -\sum_i d_i \log \hat{d}_i + (1 - d_i) \log(1 - \hat{d}_i)$// Class conditional
    invariant Loss
    $q_{ij} = e^{-\alpha||z_i - \mu_j||^2} / \sum_k e^{-\alpha||z_i - \mu_k||^2}$
    $p_{ij} = \{(q_{ij})^n / \sum_i (q_{ij})^n\} / \{\sum_k (q_{ik})^n / \sum_i (q_{ik})^n\}$
    $\mathcal{L}_{CCDI} = \sum_i \sum_j p_{ij} \log(p_{ij} / q_{ij})$// Update model weights with
    some optimizer, e.g. SGD
    $\mathcal{L} = \mathcal{L}_T + \lambda_D \mathcal{L}_D + \lambda_{CCDI} \mathcal{L}_{CCDI}$
    $\theta = \theta - \nabla_\theta \mathcal{L}$
end
```

use same settings from Ganin et al. [11] and Long et al. [31] for λ_D. In addition, the performance did not vary greatly depending on the λ_{CCDI}, the detailed analysis of the hyperparameters are in Fig. 7.

5.1. Toy example

5.1.1. Experimental settings

First of all, to verify the effect of our method, we do a simple experiment with MNIST and MNIST-M [11]. MNIST-M is a colorized version of MNIST using background patches from the Berkeley Segmentation Data Set (BSD500). To handle two datasets simultaneously, we stacked the images of MNIST to three channels, e.g., $28 \times 28 \times 3$, unless otherwise stated. The encoder consists of two convolution layers and two fully-connected layers. It maps $28 \times 28 \times 3$ images to two-dimensional feature vectors. The reason why we set the feature dimension as two is to visualize the decision boundary. Although we limit the dimension of feature vectors, the final accuracy of two digits (0 and 1) from MNIST achieves 99.5% with the classifier of two fully-connected layers.

The result is shown in Fig. 1. With the model trained on only two digits (0 and 1) from MNIST, we scatter examples of MNIST-M(target) on the feature dimension marked as a circle. Each color indicates 1(red) and 0(blue). In addition, we plot the decision boundary of the classifier and the center of MNIST(source) examples. The goal of this experiment is to distribute the red sample near the red center and the blue sample to the blue center. The arrow in each figure shows the gradient of loss of the previous class-conditional method [17] (left) and our method (right) on the same model and examples.

5.1.2. Analysis

The most widely used approach of learning class conditional distribution in domain adaptation is to use a label classifier to calculate the pseudo label of the target sample, and then minimize the distance between the sample means of source and target data conditional to their labels. There have been studies that used kernel mapping [4,14–16,18], or deep learning [17] based on this approach. Fig. 2a shows the gradient of each sample induced by the mean matching loss function. In this figure, we followed mean matching settings as [17]. The right figure shows the same results done by our method.

As in Fig. 2, compared to previous mean matching based methods, our proposed method has two different points. First, the mean matching methods rely on the decision boundary of the source classifier (black line), whereas our method focuses on the distance between each sample and two centers. However, since the decision boundary between source and target domain is different in domain adaptation settings, it has the problem of false pseudo labels. In the center of Fig. 2a, in the case when the classifier misclassifies target samples (blue samples in the red area), the previous methods make the examples with false pseudo label moves to the opposite center which is a completely wrong direction. However, in Fig. 2b, we can find that our method better indicates the correct gradient direction.

Table 1
Classification accuracy (%) on digit datasets for domain adaptation.

Methods	MNIST→MNIST-M	MNIST→USPS	SVHN→MNIST	Avg
Source Only	49.9	75.2	69.6	64.9
DANN [11]	76.7	77.7	71.4	75.2
DSN [12]	83.2	-	**82.8**	83.0
ADDA [13]	-	90.1	76.0	83.1
CCDI (Ours)	**93.6**	**98.8**	82.1	**91.5**

Second, while mean matching methods take into account the overall mean of samples of a pseudo class, our method considers each sample to learn a class conditional distribution. This can be found in the left figure that the samples in the same boundary have identical gradients, regardless of the location of each sample. Therefore, in this case, it may enforce a large negative effect on the model if there exists a false pseudo label. This phenomenon can lead to unstable learning of the whole model, especially in the initial periods of training. On the other hand, our method reasonably assigns different sizes of the gradient to the examples in inverse proportion to the distance to the source center by considering the distribution as shown in the right figure.

5.2. Digit classification

5.2.1. Setup

In this experiment, we use four different digit datasets: MNIST, MNIST-M, USPS, and SVHN. All of these datasets have 10 digits (0∼9) as labels. USPS comprises handwritten address data in grayscale 16×16 pixel images. SVHN consists of house number signs from Google Street View in three channels 32×32 pixel images.

We test our method on three transfer tasks : MNIST → MNIST-M, MNIST → USPS and SVHN → MNIST. The first task MNIST → MNIST-M can be interpreted as a real world application of a grayscale image to a color image. The second task MNIST → USPS tests how our suggested algorithm can handle different sizes of data. At last, SVHN → MNIST evaluates transferring knowledge between complicated datasets and simple datasets in real world applications.

We compare our method with four different methods: (i) Source Only: the model used only labeled source data for training without domain adaptation, (ii) DANN: Domain Adversarial Neural Networks [11], (iii) DSN: Domain Separation Networks [12], (iv) ADDA: Adversarial Discriminative Domain Adaptation [13].

For digit classification experiments, we use a custom design network. The encoder consists of two convolution layers and two fully-connected layers. After each of the convolution layers, we add the batch-norm layer, max-pool layer, and ReLU as an activation function. The label classifier and domain classifiers use only fully-connected layers with ReLU.

In addition, we use Adam as an optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$. The learning rate initially set to 0.001 and decreases 5% for every epoch. We set α in Eq. (13) as 0.1 and batch size is fixed as 200. Again, all images are rescaled to $28 \times 28 \times 3$.

5.2.2. Results

Classification performance. The results of accuracy on the test data are shown in Table 1. Our suggested model achieves the best results on tasks MNIST → MNIST-M and MNIST → USPS except for SVHN → MNIST. Especially, in the first two tasks, our model outperformed others with over 90% accuracy. In the case of SVHN → MNIST, although DSN shows the highest accuracy, the performance of the proposed model is similar to DSN. We carefully speculate that it's because of the characteristic of SVHN that contains several numbers in one image. In addition to these results, the total average of the three tasks is 21.7% increased from DANN. Thus we can conclude that our class-conditional loss function has conspicuously increased the performance of domain adaptation.

Embedding Visualization. The purpose of our suggested model is to learn a representation in which features in the same class in two domains are expected to be mapped nearby so that the classifier trained on source data can be applied well on target data. To verify our suggested model has been well trained as we intended, we visualize the hidden embedding of source and target data.

Fig. 5 illustrates feature space $\mathcal{Z}$ in three experiments. We visualize hidden representation of source domain $\mathbf{h_s}$ and target domain $\mathbf{h_t}$ by using t-SNE. Samples from the source domain are represented by $\times$, from the target domain by $\circ$ and from the centroid μ_j by ♣. Labels of each sample are illustrated in different colors. Each row represents visualization results of MNIST → MNIST-M, MNIST → USPS and SVHN → MNIST from top to bottom. Each column represents results of Source Only, DANN, and the proposed model from left to right.

In the task of MNIST → MNIST-M, we can easily see that visualization results of Source Only separately embedded source and target samples. The embedding of source samples and target samples are separately located. In this case, the classifier is unable to classify target samples with a classifier trained on source samples. In fact, this is already reflected in Table 1. Source Only shows 49.9% of accuracy on the task MNIST → MNIST-M.

In the case of DANN, the global distribution of source and target domain samples is more mixed than before. However, as this model does not consider compact class-conditional alignment, the joint distribution (a distribution with the same colors) is not compactly embedded.

In contrast, our model shows a compact class-conditional visualization of source and target representations. We can easily find out that samples from different labels (colors) are distributed separately, while samples from the same labels are compactly embedded near each centroid. In this embedding space, the classifier is able to fit both source and target domain samples, which leads to the best performance of 93.6%.

The results of other tasks show a similar tendency to MNIST → MNIST-M. In the task MNIST → USPS, Source Only shows better distribution than MNIST → MNIST-M. This can be interpreted as the evidence of this MNIST → USPS is an easier task than MNIST → MNIST-M and Source Only actually shows the high accuracy of 76.7%.

Spectral Analysis of Embedding Vectors. In the previous section, we visually verify that our suggested model makes a compact representation where data points are converged into each centroid μ. As shown in Eq. (15), we tried to make embedding z_i close to the nearest centroid μ_j. In this way, we can learn a compact representation that source and target data in label j are mapped close to centroid μ_j.

Now, in this section, we quantitatively show that our method encourages an encoder to map in the same class nearby. First, we try to compare distributions whether each data is actually located near to its labeled centroid μ.

Let q_{ij}^s denotes q_{ij} of the source data and q_{ij}^t denotes q_{ij} of the target data. As shown before, the value of q_{ij} indicates the probability of assigning sample z_i to centroid j based on similarity measures. If a sample z_i is closely embedded to one centroid μ_j, then the value of q_{ij} will be close to one and other $q_{ij'}$ will be close to zero. When a sample z_i is not embedded near any of the centroids, then every value of the q_{ij} will be equally distributed to $1/L$ where L is the number of classification labels.

We illustrate a Histogram to visualize the distribution of the values in soft assignment q_{ij} in Fig. 6. In this figure, we use a distribution of q_{ij} in a single batch which is 200 samples and used frequency as a log scale. As it shows in the bottom row which

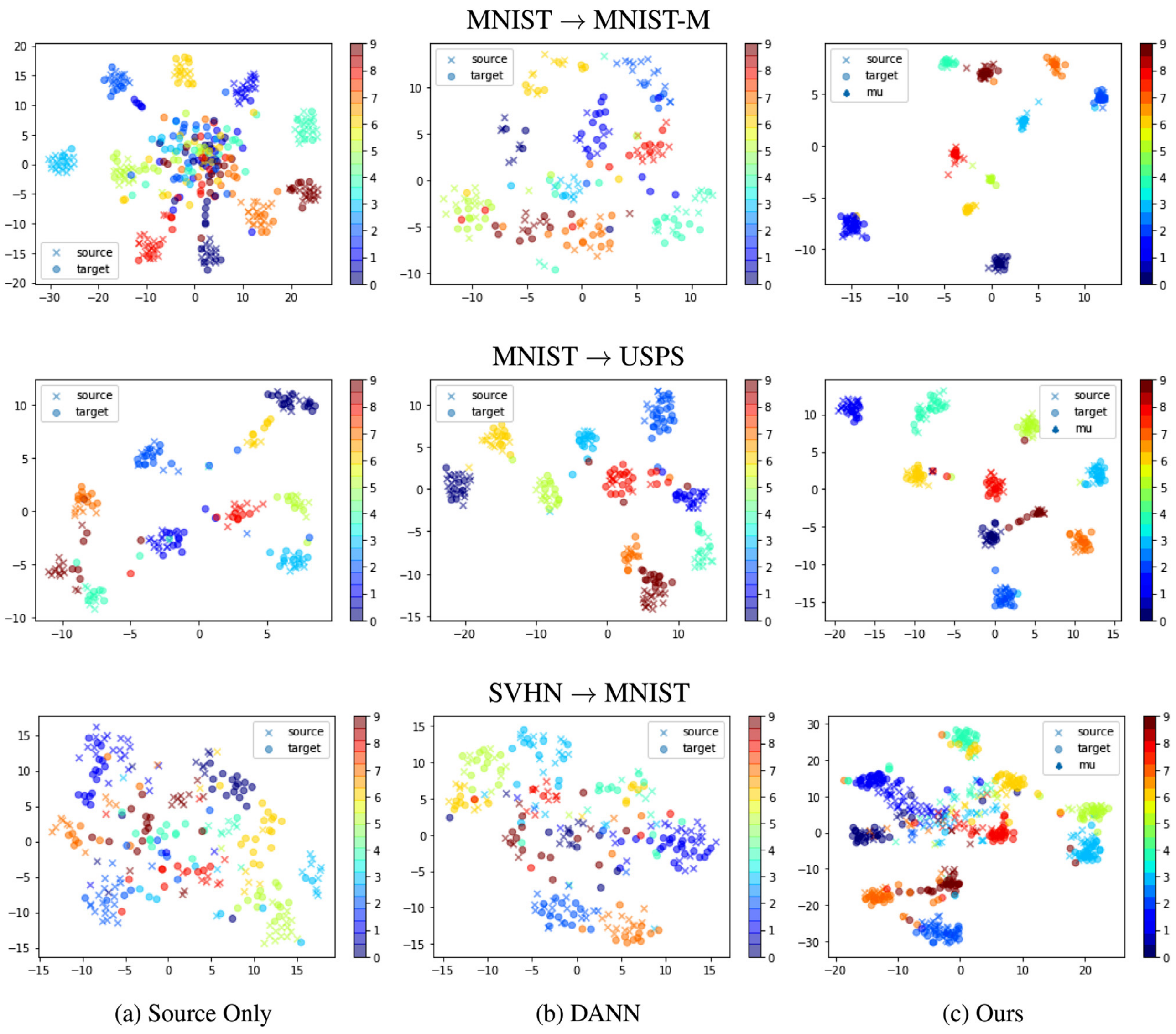


Fig. 5. t-SNE visualization of hidden representation of source domain $\mathbf{h_s}$ and target domain $\mathbf{h_t}$. Sample from source domain is represented in $\times$, target domain in $\circ$ and centroid μ_j is in ♣. Label of each samples is illustrated as different colors. Each row represents visualization results of MNIST → MNIST-M, MNIST → USPS, and SVHN → MNIST from top to bottom. Each column represents Fig. 5a: Source Only, Fig. 5b : DANN, and Fig. 5c: our suggested model.

is our method, the values of q_{ij} from our model are concentrated around zero and one compared to Source Only and DANN. These results show that each embedding vector is distributed near the center of the corresponding label. These results also correspond with the result of MNIST → USPS. As we can see in Fig. 5, all three methods successfully learned class-conditional distribution in this experiment settings. As a result, the Histogram of all three different methods are similar (q_{ij} values are concentrated on zero and one).

Furthermore, to quantitatively show that the value of our probability soft assignment q_{ij} is more concentrated on zeros and ones, we calculate information entropy of each embedding vectors as follows.

$$Entropy(q) = \sum -q_{ij}log(q_{ij})$$

Entropy is the negative logarithm of the probability mass function for the value. It can be a measure of the unpredictability of the state. If the probability distribution is deterministic, entropy would be low, and it gets high when the distribution is uniform. In this case, when the entropy of the q_{ij} is low, it indicates the soft assignment is more deterministic. In other words, embedding sample z_i is closely mapped into single centroid μ_j.

In Table 2, entropy values of three experiments are summarized. Our suggested model showed the lowest value in all experiments except for source data q^s_{ij} in MNIST → USPS. Again, MNIST → USPS is the easiest task so that Source Only is enough to reach the ideal mapping without any additional methods. This result shows the same conclusion with the above visualization results that each embedding vector is distributed near to the center of the corresponding label.

Table 2
Entropy of embedding vectors for each model.

Entropy	MNIST → MNIST-M		MNIST → USPS		SVHN → MNIST	
	q^s_{ij}	q^t_{ij}	q^s_{ij}	q^t_{ij}	q^s_{ij}	q^t_{ij}
Source Only	0.0758	0.1577	**0.0024**	0.0112	0.0543	0.0478
DANN [11]	0.1035	0.1861	0.0112	0.0072	0.0710	0.0279
CCDI (Ours)	**0.0126**	**0.0207**	0.0044	**0.0049**	**0.0096**	**0.0028**

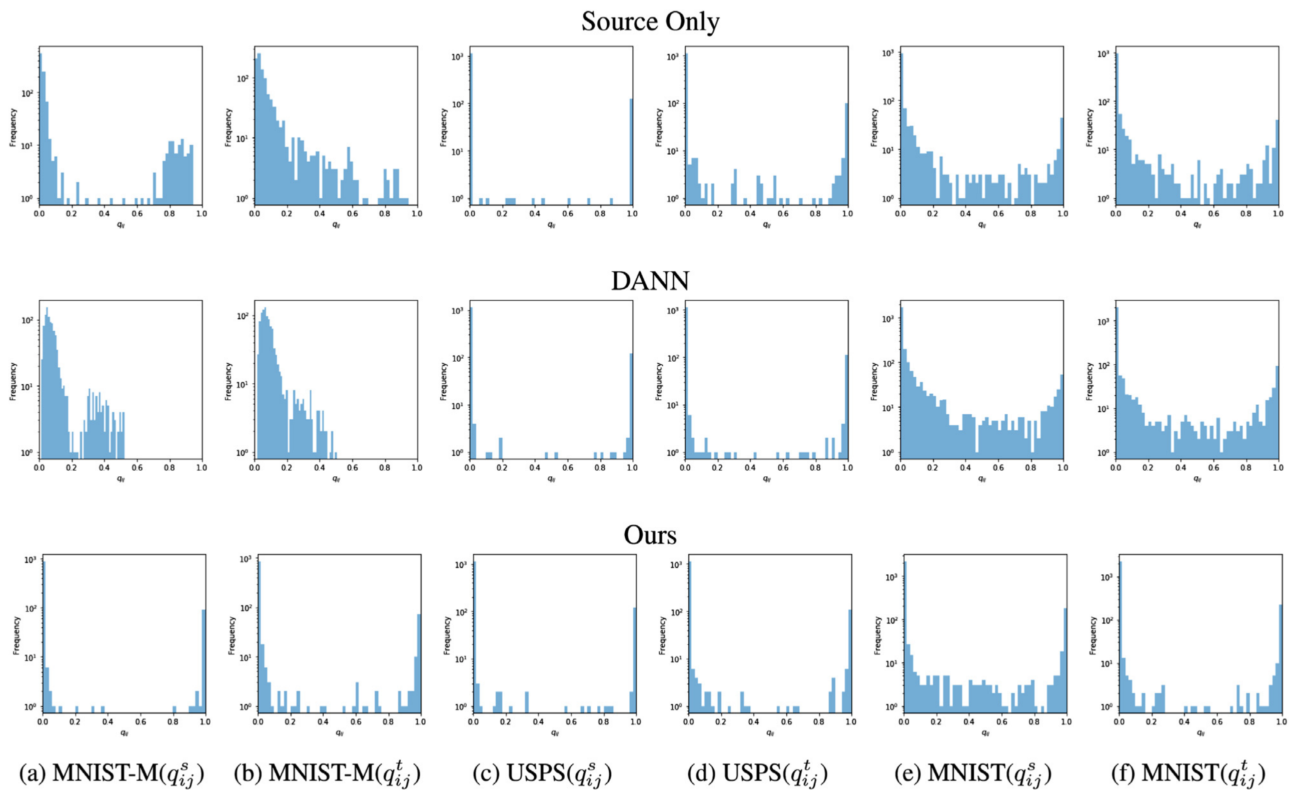


Fig. 6. Histogram of embedding vectors q_{ij} in three digit classification experiments. We used distribution of q_{ij} in a single batch which is 128 data samples and used frequency as log scale. q^s_{ij} and q^t_{ij} refers to q_{ij} in source and target data respectively. The top row shows the result of Source Only, the second row is DANN, and the bottom is our model. The left two column are the results on MNIST → MNIST-M, the middle two column are the results on MNIST → USPS and the right two column are the results on SVHN → MNIST. The caption of each column (a)-(f) indicates target dataset on each task and q_{ij} of source or target.

5.3. Image classification

5.3.1. Setup

Among widely used image datasets, we choose Office-31, Office-Home, Office-Caltech, and VisDA-2017. Office-31 contains 31 classes from three domains: *Amazon* (A), *Webcam* (W), and *DSLR* (D). *Amazon* images are gathered from amazon.com, *Webcam* and *DSLR* are taken by a camera. On the other hand, Office-Home consists of more difficult images from four domains: *Art images* (A), *Clip Art* (C), *Product images* (P), and *Real-World images* (R). Office-Caltech contains 10 common categories between the Office-31, and Caltech256, i.e., *Amazon* (A), *Webcam* (W), *DSLR* (D) and *Caltech* (C). VisDA-2017 is a little different dataset from the previous two. For each of the 12 classes, there are two distinct domains: synthetic and real images. Synthetic images are gathered from 3D models and real images are a collection of real world dashcam images.

In this experiment, we test all possible transfer tasks except Vidsa-2017. For Office-31, there are six tasks A→W, A→D, W→A, W→D, D→A D→W. Similarly, 12 tasks are considered in Office-Home and Office-Caltech in total. In the case of Vidsa-2017, we use synthetic images to train models and test them with real images across 12 classes.

We use different sets of methods for a thorough comparison with state-of-the-art methods in each dataset that shared the same experimental setting as ours (ResNet-50 and ResNet-101). Following methods are compared: DAN [5], RTN [39], DANN [11], ADDA [13], JAN [30], GTA [40], CDAN [31], 3CATN [35], BSP [41], PACET [27], and SPL [28].

PACET and SPL had a slightly different experimental setting as ours. These methods are not end-to-end deep learning methods, so they used an additional feature extractor (fine-tuned ResNet 50) for dimension reduction. Nevertheless, we have added the results to Table 3 and 4 for thorough method.

We adopt ResNet-50 for Office datasets and ResNet-101 for Visda-2017 as an encoder. We optimized the network by using SGD with momentum 0.9 and weight decay 0.0005. α in Eq. (13) is set to 100 and batch size is fixed as 36. The results in image classification of our methods are reported as an average over 10 runs with random initialization.

While we have compared our method with deep learning based domain adaptation methods on large image dataset Office-31, Office-Home, and VISDA, we also compared our method with kernel mapping based methods on Office-Caltech dataset. For Office-Caltech, we use 4096 dimensional DeCAF6 [29] features. We compared with five previous kernel mapping methods: DAN [5], LSC [14], JGSA [15], MEDA [16], DLA-DA [42], SPL [28], and GKE [25].

5.3.2. Results

Classification performance

The experimental results on Office datasets are shown in Tables 3, 4. Compared to the other methods, our method (CCDI*) shows the best performance in overall Office-31 tasks as shown in Table 3. Our method achieves the highest accuracy on four tasks. Focusing more on our model, we can see the proposed method increases average classification accuracy 5.4%p (82.2% to 87.6%) and 3.4%p (86.6% to 90.0%) over DANN and CDAN respectively. Here, superscript * is given to the data if the result of our method is significantly better than baseline methods (DANN and CDAN) supervised by Wilcoxon signed rank test with the level of the significance α is 0.05. The great aspect of our method is that performance improvements have been significantly observed for all tasks. Among them, on the task A→D, our method contributes tremendously to improve DANN as much as 14.6%.

Table 3
Classification accuracy (%) on Office-31(ResNet-50). Here, superscript † is given to the methods that use pre-processed features.

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 [5]	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
RTN [39]	84.5±0.2	96.8±0.1	99.4±0.1	77.5±0.3	66.2±0.2	64.8±0.3	81.6
DANN [11]	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
ADDA [13]	86.2±0.5	96.2±0.3	98.4±0.3	77.8±0.3	69.5±0.4	68.9±0.5	82.9
JAN [30]	85.4±0.3	97.4±0.2	99.8±0.2	84.7±0.3	68.6±0.3	70.0±0.4	84.3
GTA [40]	89.5±0.5	97.9±0.3	99.8±0.4	87.7±0.5	72.8±0.3	71.4±0.4	86.5
CDAN [31]	93.1±0.2	98.2±0.2	**100.0±0.0**	89.8±0.3	70.1±0.4	68.0±0.4	86.6
CDAN+E [31]	94.1±0.1	98.6±0.1	**100.0±0.0**	92.9±0.2	71.0±0.3	69.3±0.3	87.7
3CATN [35]	**95.3±0.2**	**99.3±0.5**	**100.0±0.0**	94.1±0.3	73.1±0.2	71.5±0.6	88.9
BSP [41]	93.3±0.2	98.2±0.2	**100.0±0.0**	93.0±0.2	73.6±0.3	72.6±0.3	88.5
PACET† [27]	89.1	97.6	99.8	90.8	73.5	73.6	87.4
SPL† [28]	92.7	98.7	99.8	93.0	76.4	**76.8**	89.6
CCDI	93.5±0.3*	98.4±0.2*	**100.0±0.0***	91.9±0.5*	72.4±0.4*	69.8±0.5*	87.6
CCDI*	**95.3**±0.3*	98.5±0.2*	**100.0±0.0**	**95.2±0.1***	**76.6±0.2***	74.5±0.2*	**90.0**

Table 4
Classification accuracy (%) on Office-Home. Here, superscript † is given to the method that uses pre-processed features.

Methods	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg
Source Only	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DAN [5]	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DANN [11]	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
JAN [30]	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
CDAN [31]	49.0	69.3	74.5	54.4	66.0	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8
CDAN+E [31]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
BSP [41]	52.0	68.6	76.1	58.0	70.3	70.2	58.6	50.2	77.6	72.2	59.3	81.9	66.3
SPL† [28]	54.5	**77.8**	**81.9**	**65.1**	**78.0**	**81.1**	**66.0**	53.1	**82.8**	69.9	55.3	**86.0**	**71.0**
CCDI	49.8*	66.9*	74.7*	52.1*	64.2*	64.2*	54.6*	49.3*	73.9*	68.6*	55.6*	79.4*	62.8
CCDI*	**55.2***	73.6	78.8*	63.5*	72.6*	73.4*	63.9*	**53.8***	79.3*	**74.5***	**60.6***	83.6*	69.4

Table 5
Classification accuracy (%) on Office-Caltech.

Methods	C→A	D→A	W→A	A→C	D→C	W→C	A→D	C→D	W→D	A→W	C→W	D→W	Avg
Source only	91.9	87.1	83.8	83.0	79.0	73.0	87.4	87.1	100.0	79.5	93.7	97.7	86.9
DAN [5]	92.0	90.0	92.1	84.1	80.3	81.2	91.7	89.3	100.0	91.8	90.6	98.5	90.1
LSC [14]	94.3	92.4	93.3	87.9	86.2	88.0	95.0	95.3	100.0	88.8	91.2	99.3	92.6
JGSA [15]	91.4	92.0	90.7	84.9	86.2	85.0	88.5	93.6	100.0	81.0	86.8	99.7	90.0
MEDA [16]	93.4	93.2	99.4	87.4	87.5	93.2	88.1	91.1	99.4	88.1	95.6	97.6	92.8
FSDA [6]	92.8	89.3	87.6	88.3	80.0	80.1	88.0	91.1	99.4	82.7	88.8	98.0	88.8
DLA-DA [42]	93.9	91.5	92.6	88.1	84.8	85.8	89.8	87.9	100.0	89.8	91.2	**100.0**	91.3
GKE [25]	93.5	93.5	**94.4**	88.4	83.8	**88.9**	**99.7**	94.3	100.0	**97.6**	**98.3**	99.7	**94.3**
SPL [28]	92.7	93.0	92.0	87.4	**88.6**	87.0	89.2	**98.7**	100.0	95.3	93.2	98.6	93.0
CCDI*	**94.7**	**94.0**	93.7	**88.7**	87.7	88.5	97.8	95.7	**100.0**	95.7	94.8	**100.0**	**94.3**

In Table 4, our proposed method also shows performance improvement of about 4%p compared to DANN and CDAN. In this experiment, SLP shows slightly better performance (1.6%) compared to ours. However, as mentioned above, considering SPL has different experimental settings with ours, it is difficult to say that the performance difference is significant. Nevertheless, among the deep learning based methods, our method outperformed other methods in all 12 experiments.

Similarly, in Office-Caltech, our proposed method shows the best results on 10 tasks out of 12 as shown in Table 5. GKE shows the same average performance compared to the proposed method. However, we note that there is a critical difference between GKE and others. To train GCN, GKE must use all target images to predict their labels in the training session. In this respect, GKE might produce better feature embedding, but it has a disadvantage in predicting the labels of new inputs on the target domain since it needs to reconstruct the graph of the target samples.

The results of Visda-2017 are shown In Tables 6 and 7. Table 6 shows the accuracy of each class and increased performance by adopting our method. Similar to Office datasets, we can see the proposed method improves the performance of each domain invariant methods on almost all data. Even though our model has greatly reduced accuracy for certain data, we can see that the average accuracy has improved from Table 7. In addition, we can see the feasibility of the proposed model in the fact that the average of the performance has been increased from based domain invariant methods close to the highest performance that BSP showed.

Additional analysis We have done sensitivity analysis for hyperparameters λ_{CCDI} and n in Eqs. (16) and (14), respectively (Fig. 7). In Fig. 7a, we can find that model performance did not vary greatly depending on value of λ_{CCDI} in range of $10^{-5} \sim 10^2$. However, if the hyperparameter exceeds 10^2, the model performance degrades significantly. Fig. 7b depicts the results on different values of n. The performance of our model is not sensitive on

Table 6
Classification accuracy (%) and performance change (%p) on Visda-2017. Numbers in bold indicates positive changes.

Tasks	DANN [11]	CDDI	Δ	CDAN [31]	CCDI*	Δ
Plane	86.3	88.2	**1.9**	83.5	89.2	**5.7**
Bcycl	63.1	67.3	**4.2**	66.3	69.9	**3.6**
Bus	71.5	73.8	**2.3**	79.8	79.5	−0.3
Car	43.5	55.1	**11.6**	61.3	57.4	−3.9
Horse	84.9	88.8	**3.9**	83.1	88.2	**5.1**
Knife	24.5	39.0	**14.5**	48.5	68.9	**20.4**
Mcycl	83.3	86.5	**3.2**	89.1	89.2	**0.1**
Person	72.8	76.3	**3.5**	73.0	73.9	**0.9**
Plant	85.4	88.0	**2.6**	88.3	86.4	−1.9
Sktbrd	64.4	43.7	−20.7	56.6	72.7	**16.1**
Train	75.0	80.0	**5.0**	74.5	82.8	**8.3**
Truck	39.2	40.0	**0.8**	38.6	33.8	−4.8
Avg	66.2	68.9	**2.7**	70.2	74.3	**4.1**

Table 7
Classification accuracy (%) on Visda-2017.

Methods	Source Only	DAN [5]	DANN [11]	MCD [32]	CDAN [31]	BSP [41]	CCDI	CCDI*
Avg	52.4	61.1	66.2	71.9	70.2	**75.9**	68.9	74.3

Table 8
Classification accuracy (%) on Amazon review dataset.

Methods	B→D	B→E	B→K	D→B	D→E	D→K	E→B	E→D	E→K	K→B	K→D	K→E	Avg
Source Only	76.0	70.0	74.0	73.6	70.9	73.2	67.9	69.2	82.4	67.8	70.2	81.6	73.1
SCL [19]	79.4	76.8	**80.1**	77.3	78.1	**80.3**	71.5	74.5	84.8	73.0	**76.3**	84.0	78.0
DANN [11]	78.4	73.3	77.9	72.3	75.4	78.3	71.3	73.8	**85.4**	70.9	74.0	84.3	76.3
CCDI	**80.6±0.2***	**78.6±0.3***	78.0±0.3	**80.0± 0.2***	**78.7±0.4***	78.2±0.3	**78.7±0.2***	**74.6±0.1***	85.2±0.4	**74.4±0.2***	73.7±0.5	**84.5±0.3**	**78.8**

Table 9
CPU running time(s) per epochs on Office 31 data.

	DANN	CDAN	CCDI	CCDI*
time(s)	0.475	0.477	0.466	0.49

the value of n. We used grid search to select the hyperparameters. The sensitive analysis on λ_D is illustrated in Fig. 7c. We can find that our model shows best performance when the value of the λ_D is between 10^0 and 10^1.

We have also measured the running time of our method in the task A → D on the Office 31 data. It is summarized in Table 9. The experiments were conducted on Xeon Gold 6126 CPU 2.60GHz with 256 GB of RAM. The running time per epoch of our model CCDI was 0.477 seconds. It was also the same as the running time of DANN which was 0.475 seconds. Results of CCDI* and CDAN were 0.490 and 0.466 seconds, respectively. Since our computational complexity of the proposed loss function is $O(n)$, it did not increase the running time. We can conclude that our method shows better performance than baseline methods without sacrificing computational time.

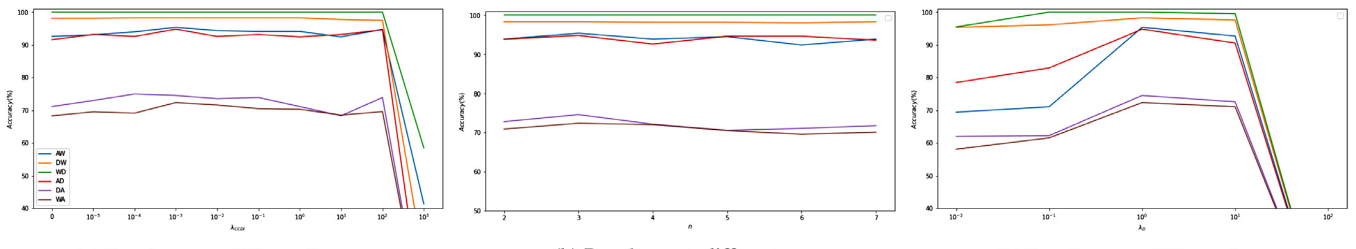
5.4. Sentimental classification

5.4.1. Setup

Among various categories, four categories that are mainly used in domain adaptation experiments are selected from the Amazon review dataset: Book(B), DVD(D), Kitchen(K), and Electronics(E). Based on these categories, all possible tasks are generated. Thus, a total of twelve tasks are considered in this experiment.

From the original review data, we first split sentences by words and make numerical vectors with Bag Of Words. Then, we use TF-IDF and Truncated SVD for generating non-sparse inputs. Through this process, we can get 200 dimensional vectors. For the test set, we apply the same pre-processing method with trained Bag Of Words, TF-IDF, and truncated SVD.

To simplify the tasks, we transform review scores between 1 and 5 to 0 and 1. This new sentiment labels, regarding scores of 1



(a) Results *w.r.t.* different λ_{CCDI} (b) Results *w.r.t.* different n (c) Results *w.r.t.* different λ_D

Fig. 7. Sensitivity of the parameters λ_{CCDI}, n and λ_D on tasks on Office-31.

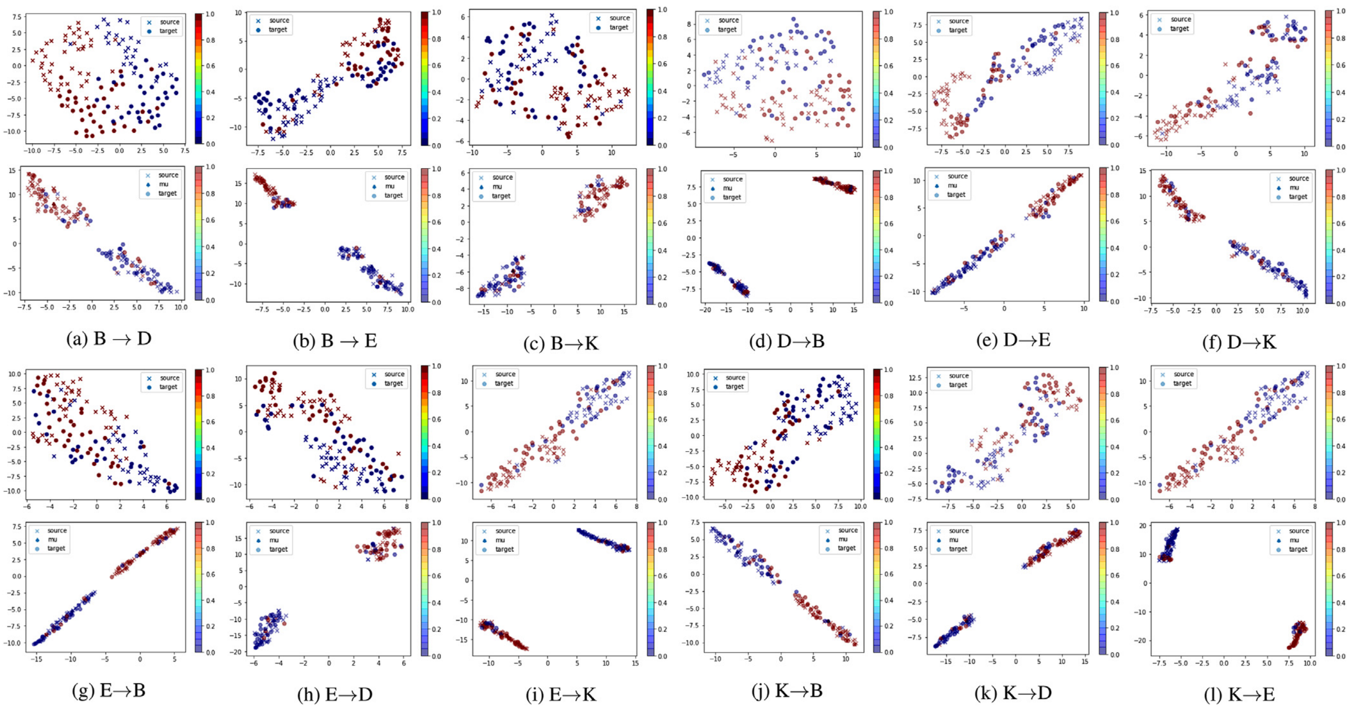


Fig. 8. Visualization of the Amazon review dataset. In each dataset, the top row is the result of DANN and the bottom is ours.

and 2 as a negative sentiment(0) and scores of 4 and 5 as a positive sentiment(1). Scores of 3 are truncated. Then, we compare our method with three different methods: Source Only, SCL [19], and DANN [11]. An encoder in this experiment we construct two fully connected layers with ReLU and Adam with a learning rate 0.001 is used without any decay. We set $\alpha = 1$ in Eq. (13).

5.4.2. Results

Classification performance. Table 8 shows the results of sentimental classification. We calculate the classification accuracy of twelve domain adaptation experiments in the Amazon review dataset. Our method shows the best performance in eight domain adaptation experiments. However, in B→K, D→K, E→K, and K→D, SCL and DANN achieve better results.

Embedding visualization. We visualize feature spaces of source and target domain after applying t-SNE. Fig. 8 shows the visualization results. The blue cross corresponds to a positive source, the blue circle corresponds to a negative source, and the target domain is colored red. The support regions of the source and target domains are separated in the visualization results of our method. Compared to DANN, we can find that our suggested class-conditional training loss function learns a data distribution where the same classes are aligned across two different domains. Therefore, we presumed that the performance degradation in three experiments in Table 8 is caused by other factors such as limitations in the pre-processing of review sentences rather than the malfunction of the proposed framework.

6. Conclusion & future works

In this paper, we proposed a class-conditional domain invariant learning for domain adaptation. We presented a new generalization upper bound for domain adaptation the motivates the need for class-conditional domain invariant learning. Then we suggested a novel algorithm that can learn class-conditional domain invariant representation without using the information of target domain labels.

Our method is straightforward in that it does not require any specific complex network structure. It can be easily applied to any previous deep learning based domain-invariant methods and forces the model to learn class-conditional alignment. Besides, since classifiers in early epochs do not guarantee that it will provide appropriate pseudo labels for target samples, our method deviates from the problem of false pseudo labels by using soft assignment of each sample as label information of target data. Moreover, beyond the joint alignment of the distribution, the proposed method allows the encoder to form a compact distribution near each centroid. The experimental result on the toy dataset shows the advantage of our proposed method on learning class conditional distribution over previous mean matching methods.

We experimented our algorithm on various classification tasks: digit classification (MNIST, USPS, MNIST-M, and SVHN), object classification (Office 31, Office-Home, Office-Caltech, and VISDA), and sentimental classification (Amazon review). Classification results show that our method outperforms previous state-of-the-art methods. Moreover, feature visualization results and spectral analysis show that our model mapped features of source and target domain similar to the centroid of their corresponding labels than that obtained from previous methods.

While our method focuses on standard domain adaptation problems, there exists a variant of domain adaptation called heterogeneous domain adaptation [43,44]. It addresses the problem where data in different domains are sampled from not only different data distributions but also different types of features with independent dimensionalities [45]. For example, the source domain is image data, whereas the target domain is text data. Hence, because the underlying assumptions about the data space are different, the problem to be solved is also different; Our problem seeks a transferable model while Qi et al. [43] seek a suitable distance metric. Our method can be extended to heterogeneous problems when the generalization bound for heterogeneous space is available, which leaves as future work.

Declaration of Competing Interest

The authors declare that they do not have any financial or non-financial conflict of interests

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (NRF-2019R1A2C2002358).

Appendix A

A1. Proof of Theorem 1

Proof.

$$\begin{aligned}\epsilon_{\mathcal{D}^t}(f\circ g) &= \epsilon_{\mathcal{D}^t}(f\circ g, f_g^t\circ g) = \epsilon_{\mathcal{D}^s}(f\circ g, f_g^t\circ g)\\ &\quad +\epsilon_{\mathcal{D}^t}(f\circ g, f_g^t\circ g) - \epsilon_{\mathcal{D}^s}(f\circ g, f_g^t\circ g)\\ &\le \epsilon_{\mathcal{D}^s}(f\circ g, f_g^t\circ g) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^s}[\|f\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K] + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^s}[\|f\circ g(\mathbf{x}) - f_g^s\circ g(\mathbf{x}) + f_g^s\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K]\\ &\quad +\tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\le \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^s}[\|f\circ g(\mathbf{x}) - f_g^s\circ g(\mathbf{x})\|_K]\\ &\quad +\mathbb{E}_{\mathbf{x}\sim\mathcal{D}^s}[\|f_g^s\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K] + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\le \epsilon_{\mathcal{D}^s}(f\circ g, f_g^s\circ g) + \epsilon_{\mathcal{D}^s}(f_g^s\circ g, f_g^t\circ g)\\ &\quad +\tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &= \epsilon_{\mathcal{D}^s}(f\circ g) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\quad +\mathbb{E}_{\mathbf{x}\sim\mathcal{D}^s}[\|f_g^s\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K]\\ &= \epsilon_{\mathcal{D}^s}(f\circ g) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\quad +\mathbb{E}_{\mathbf{z}\sim\mathcal{D}^s_{\mathcal{Z}}}[\|f_g^s(\mathbf{z}) - f_g^t(\mathbf{z})\|_K]\end{aligned}$$

since $\mathcal{D}^s_{\mathcal{Z}}$ and $\mathcal{D}^t_{\mathcal{Z}}$ are two distributions over $\mathcal{Z}$ induced by the representation function $g:\mathcal{X}\to\mathcal{Z}\in\mathcal{G}$ and $f_g^s\circ g = c^s$ and $f_g^t\circ g = c^t$. Similarly, we have

$$\begin{aligned}\epsilon_{\mathcal{D}^t}(f\circ g) &= \epsilon_{\mathcal{D}^t}(f\circ g, f_g^t\circ g) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^t}[\|f\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K]\\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^t}[\|f\circ g(\mathbf{x}) - f_g^s\circ g(\mathbf{x}) + f_g^s\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K]\\ &\le \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^t}[|h(\mathbf{x}) - f_g^s\circ g(\mathbf{x})|]\\ &\quad +\mathbb{E}_{\mathbf{x}\sim\mathcal{D}^t}[|f_g^s\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})|]\\ &= \epsilon_{\mathcal{D}^t}(f\circ g, f_g^s\circ g) + \epsilon_{\mathcal{D}^t}(f_g^s\circ g, f_g^t\circ g)\\ &= \epsilon_{\mathcal{D}^s}(f\circ g, f_g^s\circ g) + \epsilon_{\mathcal{D}^t}(f\circ g, f_g^s\circ g)\\ &\quad -\epsilon_{\mathcal{D}^s}(f\circ g, f_g^s\circ g) + \epsilon_{\mathcal{D}^t}(f_g^s\circ g, f_g^t\circ g)\\ &\le \epsilon_{\mathcal{D}^s}(f\circ g, f_g^s\circ g) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\quad +\epsilon_{\mathcal{D}^t}(f_g^s\circ g, f_g^t\circ g)\\ &= \epsilon_{\mathcal{D}^s}(f\circ g) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\quad +\mathbb{E}_{\mathbf{x}\sim\mathcal{D}^t}[\|f_g^s\circ g(\mathbf{x}) - f_g^t\circ g(\mathbf{x})\|_K]\\ &= \epsilon_{\mathcal{D}^s}(f\circ g) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)\\ &\quad +\mathbb{E}_{\mathbf{z}\sim\mathcal{D}^t_{\mathcal{Z}}}[\|f_g^s(\mathbf{z}) - f_g^t(\mathbf{z})\|_K]\end{aligned}$$

□

A2. Proof of Theorem 2

Proof. We first define the *margin* $\rho_f(x, y)$ at a source labeled example $(\mathbf{x}, y)$ by

$$\rho_f(\mathbf{x}, y) = \begin{cases}01 & \text{if } y = f\circ g(\mathbf{x})\\ -1 & \text{otherwise}\end{cases}$$

Thus, $f\circ g$ misclassifies $(\mathbf{x}, y)$ iff $\rho_f(\mathbf{x}, y)\le 0$. For any $0<\rho<1$, we define the *empirical margin loss* for multi-class classification by

$$\hat{\epsilon}_{S,\rho}(f\circ g) := \frac{1}{m}\sum_{i=1}^{m}\Psi_\rho(\rho_f(\mathbf{x}_i, y_i))$$

$$\le \frac{1}{m}\sum_{i=1}^{m}1_{\rho_f(\mathbf{x}_i, y_i)\le\rho} = \hat{\epsilon}_{\mathcal{D}^s}(f\circ g)$$

where Φ_ρ is the margin loss function defined by

$$\Psi_\rho(x) = \begin{cases}0 & \text{if } \rho\le x\\ 1 - x/\rho & \text{if } 0\le x\le\rho\\ 1 & \text{if } x\le 0\end{cases}$$

This upper bound is the fraction of the source data points that have been misclassified by or correctly classified with confidence less than or equal to ρ.

To derive an empirical risk bound of $\epsilon_{\mathcal{D}^s}(f\circ g)$, we observe that

$$\begin{aligned}\rho_{f,+}(\mathbf{x}, y) &:= 1_{y=f\circ g(\mathbf{x})} - \max_{y'\in\mathcal{Y}_K}(1_{y'=f\circ g(\mathbf{x})} - 2\rho 1_{y'=y})\\ &= \begin{cases}\min\{2\rho, 1\} & \text{if } y = f\circ g(\mathbf{x})\\ -1 & \text{otherwise}\end{cases}\end{aligned}$$

We have $\rho_{f,+}(\mathbf{x}, y)\le\rho_f(\mathbf{x}, y)$ and so $\mathbb{E}[1_{\rho_f(\mathbf{x}, y)}]\le\mathbb{E}[1_{\rho_{f,+}(\mathbf{x}, y)}]$. Also we have $\Psi_\rho(\rho_{f,+}(\mathbf{x}_i, y_i)) = \Psi_\rho(\rho_f(\mathbf{x}_i, y_i))$ by the definition of Ψ_ρ.

Now let $\tilde{\mathcal{F}}_\rho = \{(\mathbf{x}, y)\mapsto\rho_{f,+}(\mathbf{x}, y) : f\in\mathcal{F}\}$. By theorem 3.3 in [37], for any $\delta>0$, with probability at least $1-\delta$, for all $f\in\mathcal{F}$:

$$\mathbb{E}[\Psi_\rho(\rho_{f,+}(\mathbf{x}, y))]\le\frac{1}{m}\sum_{i=1}^{m}\Psi_\rho(\rho_{f,+}(\mathbf{x}_i, y_i)) + 2\Re_m(\Psi_\rho\circ\tilde{\mathcal{F}}_\rho) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}.$$

The slope of the function Ψ_ρ defining the margin loss is at most $1/\rho$, thus Ψ_ρ is $1/\rho$-Lipschitz and so $\Re_m(\Psi_\rho\circ\tilde{\mathcal{F}}_\rho)\le\frac{1}{\rho}\Re_m(\tilde{\mathcal{F}}_\rho)$ by Talagrand's lemma. Since $1_{u\le 0}\le\Psi_\rho(u)$ for all $u\in\mathbb{R}$, $\epsilon_{\mathcal{D}^s}(f\circ g) = \mathbb{E}[1_{\rho_f(\mathbf{x}, y)}]\le\mathbb{E}[1_{\rho_{f,+}(\mathbf{x}, y)}]\le\mathbb{E}[\Psi_\rho(\rho_{f,+}(\mathbf{x}, y))]$, and $\Psi_\rho(\rho_{f,+}(\mathbf{x}_i, y_i)) = \Psi_\rho(\rho_f(\mathbf{x}_i, y_i))$, we have for any $\delta>0$, with probability at least $1-\delta$, for all $f\in\mathcal{F}$:

$$\epsilon_{\mathcal{D}^s}(f\circ g)\le\frac{1}{m}\sum_{i=1}^{m}\Psi_\rho(\rho_f(\mathbf{x}_i, y_i)) + \frac{2}{\rho}\Re_m(\tilde{\mathcal{F}}_\rho) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}.$$

Next to derive an upper bound of $\Re_m(\tilde{\mathcal{F}}_\rho)$, we observe the following, similarly to the proof of theorem 9.2 in [37]:

$$\begin{aligned}\Re_m(\tilde{\mathcal{F}}_\rho) &= \frac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i\rho_{f,+}(\mathbf{x}_i, y_i)\right]\\ &= \frac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i(1_{y_i=f\circ g(\mathbf{x}_i)} - \max_{y\in\mathcal{Y}_K}(1_{y=f\circ g(\mathbf{x}_i)} - 2\rho 1_{y=y_i}))\right]\\ &\le \frac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i 1_{y_i=f\circ g(\mathbf{x}_i)}\right]\\ &\quad +\frac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i\max_{y\in\mathcal{Y}_K}(1_{y=f\circ g(\mathbf{x}_i)} - 2\rho 1_{y=y_i})\right]\end{aligned}$$

The first term in the right hand side of the above inequality is upper-bounded as follows:

$$\begin{aligned}
&\tfrac{1}{m}\mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\textstyle\sum_{i=1}^m \sigma_i 1_{y_i=f\circ g(\mathbf{x}_i)}\right]\\
&\quad= \tfrac{1}{m}\mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\textstyle\sum_{i=1}^m \sum_{y\in\mathcal{Y}_K} \sigma_i 1_{y=f\circ g(\mathbf{x}_i)} 1_{y=y_i}\right]\\
&\quad\le \tfrac{1}{m}\textstyle\sum_{y\in\mathcal{Y}_K}\mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i 1_{y=f\circ g(\mathbf{x}_i)} 1_{y=y_i}\right]\\
&\quad= \tfrac{1}{m}\textstyle\sum_{y\in\mathcal{Y}_K}\mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i 1_{y=f\circ g(\mathbf{x}_i)}\left(\frac{2\cdot 1_{y=y_i}-1}{2}+\frac{1}{2}\right)\right]\\
&\quad\le \tfrac{1}{2m}\textstyle\sum_{y\in\mathcal{Y}_K}\mathbb{E}_\sigma[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i\epsilon_i 1_{y=f\circ g(\mathbf{x}_i)}]\\
&\qquad+\tfrac{1}{2m}\textstyle\sum_{y\in\mathcal{Y}_K}\mathbb{E}_\sigma[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i 1_{y=f\circ g(\mathbf{x}_i)}]\\
&\quad= \textstyle\sum_{y\in\mathcal{Y}_K}\tfrac{1}{m}\mathbb{E}_\sigma[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i 1_{y=f\circ g(\mathbf{x}_i)}] \le \sum_{y\in\mathcal{Y}_K}\hat{\mathfrak{R}}_{\mathcal{D}^s}(\Lambda_g(\mathcal{F}))
\end{aligned}$$

where $\epsilon_i := 2\cdot 1_{y=y_i} - 1 \in \{-1,\ +1\}$ and σ_i and $\sigma_i\epsilon_i$ admit the same distribution. Therefore we have

$$\tfrac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\textstyle\sum_{i=1}^m \sigma_i 1_{y_i=f\circ g(\mathbf{x}_i)}\right] \le K\mathfrak{R}_m(\Lambda_g(\mathcal{F}))$$

The second term in the right hand side of the above inequality is upper-bounded as follows. Observing that $-\sigma_i$ and σ_i are distributed in the same way and using the sub-additivity of sup and lemma 9.1 in Mohri et al. [37] leads to

$$\begin{aligned}
&\tfrac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\textstyle\sum_{i=1}^m \sigma_i \max_{y\in\mathcal{Y}_K}(1_{y=f\circ g(\mathbf{x}_i)} - 2\rho 1_{y=y_i})\right]\\
&\quad\le \textstyle\sum_{y\in\mathcal{Y}_K}\tfrac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i(1_{y=f\circ g(\mathbf{x}_i)} - 2\rho 1_{y=y_i})\right]\\
&\quad= \textstyle\sum_{y\in\mathcal{Y}_K}\tfrac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\left(\sum_{i=1}^m \sigma_i(1_{y=f\circ g(\mathbf{x}_i)})\right) - 2\rho\sum_{i=1}^m \sigma_i 1_{y=y_i}\right]\\
&\quad= \textstyle\sum_{y\in\mathcal{Y}_K}\tfrac{1}{m}\mathbb{E}_{S,\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^m \sigma_i 1_{y=f\circ g(\mathbf{x}_i)}\right] \le K\mathfrak{R}_m(\Lambda_g(\mathcal{F}))
\end{aligned}$$

since σ_i have zero mean. Therefore we have the following general margin bound of $\epsilon_{\mathcal{D}^s}(f\circ g)$

$$\epsilon_{\mathcal{D}^s}(f\circ g) \le \hat{\epsilon}_{S,\rho}(f\circ g) + \frac{4K}{\rho}\mathfrak{R}_m(\Lambda_g(\mathcal{F})) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

By letting $\rho \to 1$, we have the first empirical risk bound on $\epsilon_{\mathcal{D}^s}(f\circ g)$ as:

$$\epsilon_{\mathcal{D}^s}(f\circ g) \le \hat{\epsilon}_{\mathcal{D}^s}(f\circ g) + 4K\mathfrak{R}_m(\Lambda_g(\mathcal{F})) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

Next we derive a generalization upper bound for $d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t)$ as follows: By theorem 3.3 in [37], for any $\delta > 0$, with probability at least $1-\delta/2$ for all $f\in\mathcal{F}$ (for all $g = (f\circ g)\oplus(f'\circ g)\in\mathcal{H}_g\Delta\mathcal{H}_g$):

$$\begin{aligned}
\tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\hat{\mathcal{D}}^s_m) &= \sup_{f,f'\in\mathcal{H}_g}|\epsilon_{\mathcal{D}^s}(f\circ g, f'\circ g) - \hat{\epsilon}_{\hat{\mathcal{D}}^s_m}(f\circ g, f'\circ g)|\\
&= \sup_{g\in\mathcal{H}_g\Delta\mathcal{H}_g}|\mathbb{E}_{\mathcal{D}^s}[1_g] - \hat{\mathbb{E}}_{\hat{\mathcal{D}}^s_m}[1_g]|\\
&\le 2\mathfrak{R}_{\mathcal{D}^s}(\mathcal{H}_g\Delta\mathcal{H}_g) + \sqrt{\tfrac{\log\frac{2}{\delta}}{2m}}.
\end{aligned}$$

Also similarly we have, for any $\delta > 0$, with probability at least $1-\delta/2$ for all $f\in\mathcal{F}$:

$$\tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^t,\hat{\mathcal{D}}^t_m) \le 2\mathfrak{R}_{\mathcal{D}^t}(\mathcal{H}_g\Delta\mathcal{H}_g) + \sqrt{\tfrac{\log\frac{2}{\delta}}{2m}}.$$

Utilizing the triangle inequality and symmetry property of $d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\cdot,\cdot)$, we have, for any $\delta > 0$, with probability at least $1-\delta$ for all $f\in\mathcal{F}$:

$$\begin{aligned}
\tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\mathcal{D}^t) &\le \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\mathcal{D}^s,\hat{\mathcal{D}}^s_m) + \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\hat{\mathcal{D}}^s_m,\hat{\mathcal{D}}^t_m)\\
&\quad+\tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\hat{\mathcal{D}}^t_m,\mathcal{D}^t)\\
&\le \tfrac{1}{2}d_{\mathcal{H}_g\Delta\mathcal{H}_g}(\hat{\mathcal{D}}^s_m,\hat{\mathcal{D}}^t_m) + 2\mathfrak{R}_{\mathcal{D}^s}(\mathcal{H}_g\Delta\mathcal{H}_g)\\
&\quad+2\mathfrak{R}_{\mathcal{D}^t}(\mathcal{H}_g\Delta\mathcal{H}_g) + 2\sqrt{\tfrac{\log\frac{2}{\delta}}{2m}}
\end{aligned}$$

Observe that $\mathrm{VCdim}(\mathcal{H}_g\Delta\mathcal{H}_g) \le 2\mathrm{VCdim}(\mathcal{H}_g)$ as in Anthony and Bartlett [46] since any $g\in\mathcal{H}_g\Delta\mathcal{H}_g$ can be represented as a linear threshold network of depth 2 with 2 hidden units. Note also that $2g(\mathbf{x})-1\in\{-1,1\}$ since $g(\mathbf{x})\in\{0,1\}$. Therefore this result combined with Corollary 3.9 of [37] leads to

$$\begin{aligned}
\mathfrak{R}_{\mathcal{D}^s}(\mathcal{H}_g\Delta\mathcal{H}_g) &\le \frac{1}{2}\sqrt{\frac{4d\log\frac{em}{2d}}{m}}, \quad\text{and}\quad \mathfrak{R}_{\mathcal{D}^t}(\mathcal{H}_g\Delta\mathcal{H}_g)\\
&\le \frac{1}{2}\sqrt{\frac{4d\log\frac{em}{2d}}{m}}
\end{aligned}$$

where $d = \mathrm{VCdim}(\mathcal{H}_g)$.

Finally, let a representation function $g:\mathcal{X}\to\mathcal{Z}\in\mathcal{G}$ be chosen in such a way to make its induced two distributions over $\mathcal{Z}$, $\mathcal{D}^s_\mathcal{Z}$ and $\mathcal{D}^t_\mathcal{Z}$ be identically distributed (or similar to each other), i.e. $\mathcal{D}^s_\mathcal{Z}\sim\mathcal{D}^t_\mathcal{Z}$, then,

$$\begin{aligned}
&\min\{\mathbb{E}_{\mathbf{z}\sim\mathcal{D}^s_\mathcal{Z}}[\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K], \mathbb{E}_{\mathbf{z}\sim\mathcal{D}^t_\mathcal{Z}}[\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K]\}\\
&\quad= \mathbb{E}_{\mathbf{z}\sim\mathcal{D}^s_\mathcal{Z}}[\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K]
\end{aligned}$$

Therefore, using theorem 3.3 again in [37], for any $\delta > 0$, with probability at least $1-\delta$, we have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{z}\sim\mathcal{D}^s_\mathcal{Z}}[\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K] \le \tfrac{1}{m}\textstyle\sum_{i=1}^m[\|f^s_g(\mathbf{z}_i) - f^t_g(\mathbf{z}_i)\|_K] + 2\mathfrak{R}_m(\mathcal{F}^s)\\
&\quad+\sqrt{\tfrac{\log\frac{1}{\delta}}{2m}}
\end{aligned}$$

where $\mathcal{F}^s = \{\mathbf{z}\to\|f^s_g(\mathbf{z}) - f^t_g(\mathbf{z})\|_K : \mathbf{z}\in\mathcal{Z}\}$. Since

$$\mathfrak{R}_m(\mathcal{F}^s) = \mathbb{E}_S\mathbb{E}_{\boldsymbol{\sigma}}\left[\tfrac{1}{m}\textstyle\sum_{i=1}^m \sigma_i\|f^s_g(\mathbf{z}_i) - f^t_g(\mathbf{z}_i)\|_K\right] = 0$$

the result follows. □

References

[1] J. Li, K. Lu, Z. Huang, H.T. Shen, On both cold-start and long-tail recommendation with social data, IEEE Trans. Knowl. Data Eng. (2019).

[2] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, Analysis of representations for domain adaptation, in: Advances in Neural Information Processing Systems, 2007, pp. 137–144.

[3] M. Sugiyama, M. Krauledat, K.-R. MÃžller, Covariate shift adaptation by importance weighted cross validation, J. Mach. Learn. Res. 8 (May) (2007) 985–1005.

[4] M. Long, J. Wang, G. Ding, J. Sun, S.Y. Philip, Transfer feature learning with joint distribution adaptation, in: 2013 IEEE International Conference on Computer Vision, IEEE, 2013, pp. 2200–2207.

[5] M. Long, Y. Cao, J. Wang, M. Jordan, Learning transferable features with deep adaptation networks, in: International Conference on Machine Learning, 2015, pp. 97–105.

[6] F. Sun, H. Wu, Z. Luo, W. Gu, Y. Yan, Q. Du, Informative feature selection for domain adaptation, IEEE Access 7 (2019) 142551–142563.

[7] W.-Y. Deng, A. Lendasse, Y.-S. Ong, I.W.-H. Tsang, L. Chen, Q.-H. Zheng, Domain adaption via feature selection on explicit feature map, IEEE Trans. Neural Netw. Learn.Syst. 30 (4) (2018) 1180–1190.

[8] W. Lee, H. Kim, J. Lee, Self training domain adaptation for real world application of deep learning model, in: Proceedings of International Conference on Computers and Industrial Engineering, CIE, 2018, pp. 1975–1983.

[9] W. Lee, J. Lee, S. Park, Instance weighting domain adaptation using distance kernel, Ind. Eng. Manage. Syst. 17 (2) (2018) 334–340.

[10] S. Park, W. Lee, J. Lee, Learning of indiscriminate distributions of document embeddings for domain adaptation, Intell. Data Anal. 23 (4) (2019) 779–797.

[11] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, TheJournal of Machine Learning Research 17 (1) (2016). 2096–2030

[12] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, D. Erhan, Domain separation networks, in: Advances in Neural Information Processing Systems, 2016, pp. 343–351.

[13] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial discriminative domain adaptation, in: Computer Vision and Pattern Recognition (CVPR), vol. 1, 2017, p. 4.

[14] C.-A. Hou, Y.-H.H. Tsai, Y.-R. Yeh, Y.-C.F. Wang, Unsupervised domain adaptation with label and structural consistency, IEEE Trans. Image Process. 25 (12) (2016) 5552–5562.

[15] J. Zhang, W. Li, P. Ogunbona, Joint geometrical and statistical alignment for visual domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1859–1867.

[16] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, P.S. Yu, Visual domain adaptation with manifold embedded distribution alignment, in: Proceedings of the 26th ACM International Conference on Multimedia, 2018, pp. 402–410.

[17] S. Xie, Z. Zheng, L. Chen, C. Chen, Learning semantic representations for unsupervised domain adaptation, in: International Conference on Machine Learning, 2018, pp. 5419–5428.

[18] J. Wang, Y. Chen, S. Hao, W. Feng, Z. Shen, Balanced distribution adaptation for transfer learning, in: 2017 IEEE International Conference on Data Mining (ICDM), IEEE, 2017, pp. 1129–1134.

[19] J. Blitzer, R. McDonald, F. Pereira, Domain adaptation with structural correspondence learning, in: Proceedings of the 2006 conference on empirical methods in natural language processing, Association for Computational Linguistics, 2006, pp. 120–128.
[20] Y. Li, N. Wang, J. Shi, X. Hou, J. Liu, Adaptive batch normalization for practical domain adaptation, Pattern Recognit. 80 (2018) 109–117.
[21] B. Yang, A.J. Ma, P.C. Yuen, Learning domain-shared group-sparse representation for unsupervised domain adaptation, Pattern Recognit. 81 (2018) 615–632.
[22] X. Glorot, A. Bordes, Y. Bengio, Domain adaptation for large-scale sentiment classification: a deep learning approach, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 513–520.
[23] M. Chen, Z. Xu, K.Q. Weinberger, F. Sha, Marginalized denoising autoencoders for domain adaptation, in: Proceedings of the 29th International Coference on International Conference on Machine Learning, Omnipress, 2012, pp. 1627–1634.
[24] W. Wang, S.J. Pan, Transferable interactive memory network for domain adaptation in fine-grained opinion extraction, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 7192–7199.
[25] H. Wu, Y. Yan, Y. Ye, M.K. Ng, Q. Wu, Geometric knowledge embedding for unsupervised domain adaptation, Knowl.-Based Syst. 191 (2020) 105155.
[26] H. Zhao, R.T. Des Combes, K. Zhang, G. Gordon, On learning invariant representations for domain adaptation, in: International Conference on Machine Learning, 2019, pp. 7523–7532.
[27] J. Liang, R. He, Z. Sun, T. Tan, Exploring uncertainty in pseudo-label guided unsupervised domain adaptation, Pattern Recognit. 96 (2019) 106996.
[28] Q. Wang, T.P. Breckon, Unsupervised domain adaptation via structured prediction based selective pseudo-labeling, (2019) arXiv:1911.07982.
[29] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, DeCAF: A deep convolutional activation feature for generic visual recognition, in: International Conference on Machine learning, 2014, pp. 647–655.
[30] M. Long, H. Zhu, J. Wang, M.I. Jordan, Deep transfer learning with joint adaptation networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 2208–2217.
[31] M. Long, Z. Cao, J. Wang, M.I. Jordan, Conditional adversarial domain adaptation, in: Advances in Neural Information Processing Systems, 2018, pp. 1640–1650.
[32] K. Saito, K. Watanabe, Y. Ushiku, T. Harada, Maximum classifier discrepancy for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3723–3732.
[33] C. Chen, W. Xie, W. Huang, Y. Rong, X. Ding, Y. Huang, T. Xu, J. Huang, Progressive feature alignment for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 627–636.
[34] S. Cicek, S. Soatto, Unsupervised domain adaptation via regularized conditional alignment, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 1416–1425.
[35] J. Li, E. Chen, Z. Ding, L. Zhu, K. Lu, Z. Huang, Cycle-consistent conditional adversarial transfer networks, in: Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 747–755.
[36] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J.W. Vaughan, A theory of learning from different domains, Mach. Learn. 79 (1-2) (2010) 151–175.
[37] M. Mohri, A. Rostamizadeh, A. Talwalkar, Foundations of Machine Learning, MIT press, 2018.
[38] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, in: International Conference on Machine Learning, 2016, pp. 478–487.
[39] M. Long, H. Zhu, J. Wang, M.I. Jordan, Unsupervised domain adaptation with residual transfer networks, in: Advances in Neural Information Processing Systems, 2016, pp. 136–144.
[40] S. Sankaranarayanan, Y. Balaji, C.D. Castillo, R. Chellappa, Generate to adapt: aligning domains using generative adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8503–8512.
[41] X. Chen, S. Wang, M. Long, J. Wang, Transferability vs. discriminability: batch spectral penalization for adversarial domain adaptation, in: International Conference on Machine Learning, 2019, pp. 1081–1090.
[42] L. Luo, L. Chen, S. Hu, et al., Discriminative label consistent domain adaptation, (2018) arXiv:1802.08077.
[43] G.-J. Qi, C. Aggarwal, T. Huang, Transfer learning of distance metrics by cross–domain metric sampling across heterogeneous spaces, in: Proceedings of the 2012 SIAM International Conference on Data Mining, SIAM, 2012, pp. 528–539.
[44] L. Jingjing, J. Mengmeng, L. Ke, Z. Lei, S.H. Tao, Locality preserving joint transfer for domain adaptation, (2019) arXiv:1906.07441.
[45] J. Li, K. Lu, Z. Huang, L. Zhu, H.T. Shen, Transfer independently together: a generalized framework for domain adaptation, IEEE Trans. Cybern. 49 (6) (2018) 2144–2155.
[46] M. Anthony, P.L. Bartlett, Neural Network Learning: Theoretical Foundations, Cambridge University Press, 2009.

Woojin Lee is a researcher in Seoul National University. He received the B.S. degree in industrial engineering from Yonsei University, and the Ph.D. degree in industrial engineering from Seoul National University. His research interests include transfer learning, machine learning, adversarial robustness, and its applications.

Hoki Kim received the B.S. degree in industrial engineering from Seoul National University, South Korea in 2018, where he is currently pursuing the Ph.D. degree with the department of industrial engineering. His research interests include machine learning, deep learning, and its applications.

Jaewook Lee is a professor in the Department of Industrial Engineering at Seoul National University, Seoul, Korea. He received the B.S. degree in mathematics from Seoul National University, and the Ph.D. degree in applied mathematics from Cornell University in 1993 and 1999, respectively. His research interests include machine learning, neural networks, global optimization, and their applications to data mining and financial engineering.