Improved Knowledge Transfer for Semi-supervised Domain Adaptation via Trico Training Strategy

Ba Hung Ngo, Yeon Jeong Chae, Jung Eun Kwon, Jae Hyeon Park and Sung In Cho*
Department of Multimedia Engineering, Dongguk University
{ngohung, toshinyg, kje_9912}@dgu.ac.kr {pjh0011, csi2267}@dongguk.edu

Abstract

The motivation of the semi-supervised domain adaptation (SSDA) is to train a model by leveraging knowledge acquired from the plentiful labeled source combined with extremely scarce labeled target data to achieve the lowest error on the unlabeled target data at the testing time. However, due to inter-domain and intra-domain discrepancies, the improvement of classification accuracy is limited. To solve these, we propose the Trico-training method that utilizes a multilayer perceptron (MLP) classifier and two graph convolutional network (GCN) classifiers called inter-view GCN and intra-view GCN classifiers. The first co-training strategy exploits a correlation between MLP and inter-view GCN classifiers to minimize the inter-domain discrepancy, in which the inter-view GCN classifier provides its pseudo labels to teach the MLP classifier, which encourages class representation alignment across domains. In contrast, the MLP classifier gives feedback to the inter-view GCN classifier by using a new concept, 'pseudo-edge', for neighbor's feature aggregation. Doing this increases the data structure mining ability of the inter-view GCN classifier; thus, the quality of generated pseudo labels is improved. The second co-training strategy between MLP and intra-view GCN is conducted in a similar way to reduce the intra-domain discrepancy by enhancing the correlation between labeled and unlabeled target data. Due to an imbalance in classification accuracy between inter-view and intra-view GCN classifiers, we propose the third co-training strategy that encourages them to cooperate to address this problem. We verify the effectiveness of the proposed method on three standard SSDA benchmark datasets: Office-31, Office-Home, and DomainNet. The extended experimental results show that our method surpasses the prior state-of-the-art approaches in SSDA.

1. Introduction

Recently, the semi-supervised domain adaptation (SSDA) task has received much attention because the target classification accuracy significantly increases thanks to a little labeled target data during training. However, it releases a new issue called intra-domain discrepancy presenting the difference between labeled and unlabeled target data within the target domain. Specifically, only the unlabeled target data having a strong correlation with the labeled target data is attracted for alignment, while the unlabeled target data having a less correlation with the labeled target data can be misaligned. Therefore, SSDA is still a challenging task because of existing inter-domain and intra-domain discrepancies, as represented in Figure 1.

The inter-domain discrepancy occurs due to the different data distribution between source and target domains called domain shift [27]. To alleviate inter-domain discrepancy, the previous works [1, 4, 24, 28, 34] rely much on the adversarial learning strategy. In contrast, to solve the intra-domain discrepancy, many approaches [10,11,15,19,25] increase the correlation between labeled and unlabeled target representations by using contrastive learning or clustering integrated with the pseudo-labeling strategy. For example, CDAC [10] combines the pseudo labeling and clustering methods to enhance the relationship between labeled and unlabeled target data, while Con^{2}DA [19] and CLDA [25] select a solution combining pseudo labeling and contrastive learning. However, the classification accuracy of these approaches has opportunities for improvement because their multilayer perceptron (MLP) classifiers often misclassify the unlabeled target data since the inter-domain and intra-domain discrepancies still exist. This is because the quality and quantity of pseudo labels generated by these MLP classifiers are still limited. Indeed, the MLP classifier only has the ability to exploit the semantic information of each individual image; thus, it can be failed to capture neighbor features for generalizing data structure for training. To solve this problem, we take advantage of the graph convolutional network (GCN) classifier for the neighbor feature aggregation that effectively mines the data structure. To be specific, we use an inter-view GCN classifier to elaborately exploit label information on the rich source data that provides an inter-view observation on the unlabeled target data. Then,

[*] Corresponding author.

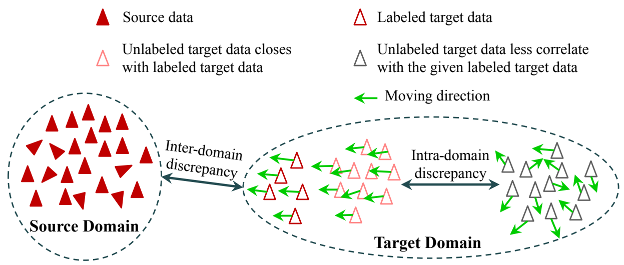

Source data
Labeled target data
Unlabeled target data closes with labeled target data
Unlabeled target data less correlate with the given labeled target data
Moving direction
Inter-domain discrepancy
Intra-domain discrepancy
Source Domain
Target Domain

Figure 1: Illustration of inter-domain and intra-domain discrepancies in the SSDA setting.

this inter-view GCN classifier generates pseudo labels for supporting the MLP classifier to alleviate the inter-domain discrepancy. Similarly, we use an intra-view GCN classifier to mine data structure on the limited labeled target data, which offers an intra-view observation on the unlabeled target data. Then, this intra-view GCN classifier also creates pseudo labels that guide the MLP classifier to mitigate the intra-domain discrepancy by enhancing the correlation between labeled and unlabeled target samples. To increase the quality of pseudo labels generated from inter-view and intra-view GCN classifiers, we introduce a novel concept called ‘*pseudo-edge*’ created by the MLP classifier to train these GCN classifiers. The mutual interaction among two GCN classifiers and the MLP classifier can be represented by two co-training strategies such as MLP and inter-view GCN, and MLP and intra-view GCN.

However, the number of labeled source data is significantly larger than the labeled target data. Therefore, the imbalance of classification accuracy between inter-view GCN and intra-view GCN classifiers can occur. Finally, to solve this problem, we introduce the third co-training strategy, in which these two GCN classifiers teach each other by exchanging their pseudo labels. We summarize the contributions of this paper as follows:

- We propose a method Trico-training (TriCT) that includes three co-training strategies to overcome the inter-and-intra-domain discrepancies and the imbalance classification accuracy issue in the SSDA task.
- We successfully cooperate between GCN and MLP classifier models with the pseudo labeling technique flexibly to boost the classification performance by introducing a novel concept named ‘*pseudo-edge*’.
- The experimental results of the proposed TriCT on three benchmark datasets, including *Office-31*, *Office-Home*, and *DomainNet* surpass the state-of-the-art approaches.

2. Related works

2.1. Semi-supervised domain adaptation (SSDA)

The main goal of SSDA is to use the knowledge extracted from a large amount of labeled source data and a small amount of labeled target data to minimize the classification error on the unlabeled target data. MME [23] is the most popular method in SSDA using a minimax entropy strategy, in which the labeled source and target samples are integrated to estimate the prototypes. Then, the minimax entropy strategy is used to encourage the estimated prototypes toward the unlabeled target samples. Inspired by this approach, UODA [20] and ASDA [21] introduce a new framework that trains multiple classifier models with different minimax entropy strategies for explicit feature alignment. However, the classification accuracy of these approaches still has room to improve due to the biased prototype estimation and intra-domain discrepancy issues. The estimated prototypes are dominated by the rich information of the source data. The intra-domain discrepancy occurs within the target domain, which is firstly concerned and analyzed by APE [8], where only unlabeled target samples are aligned with the labeled target samples if they are located nearby these labeled target samples, while other unlabeled target samples located far from the labeled target samples can be misaligned.

2.2. Pseudo labeling on SSDA

Recently, the pseudo-labeling techniques [10, 11, 14–16, 19, 25, 32] have shown a remarkable ability to improve the target classification performance in the SSDA setting. MAP-F [16], PAC [14], CDAC [10], and MCL [31] use the pseudo labeling and consistency regularization for self-training with a single classifier. Besides, Con^{2}DA [19], and CLDA [25] show outstanding classification performance on the SSDA task by using contrastive learning integrated with pseudo labeling. Furthermore, DECOTA [32] and MVCL [15] significantly improve the target classification accuracy with a divide-to-conquer strategy, in which they split the SSDA task into subtasks; then, they use different models to handle different tasks. Finally, these models teach each other by exchanging their pseudo labels via the proposed co-training strategy. However, the quality and quantity of generated pseudo labels from these abovementioned approaches have an opportunity for improvement. That is because they use the multilayer perceptron (MLP) classifiers to extract pseudo labels, while the MLP classifier only exploits information of each individual image; thus, it can fail to explore the neighborhood structure. To solve this problem, we take advantage of the GCN classifier for the feature aggregation that effectively mines the data structure to increase the number of generated pseudo labels with high reliability.

3. The proposed method

Problem definitions. In the SSDA setting, we have a large amount of labeled source data $\mathcal{D}_{\mathcal{S}} = \{x^i_{\mathcal{S}}, y^i_{\mathcal{S}}\}_{i=1}^{\mathcal{N}_{\mathcal{S}}}$,

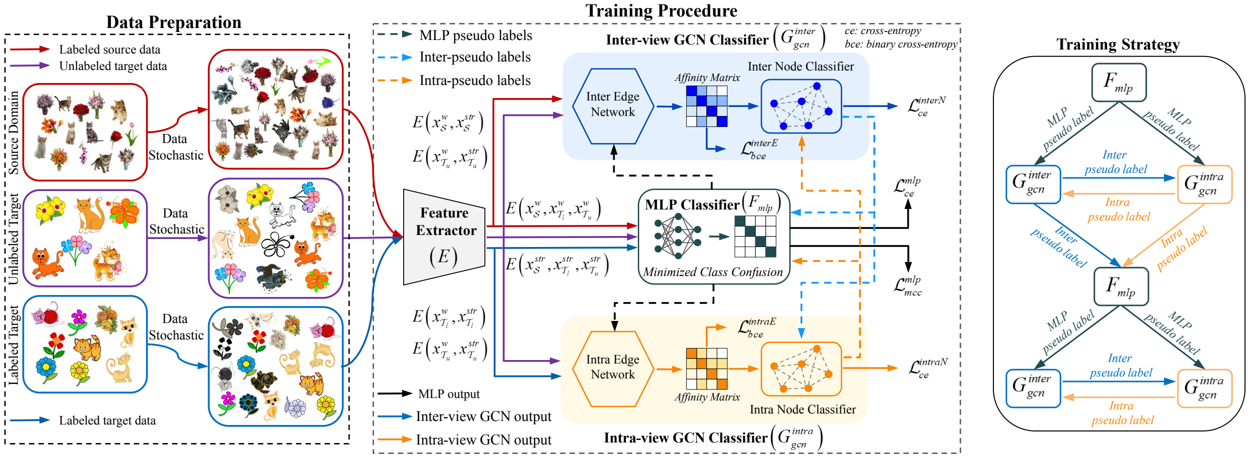

Data Preparation
Labeled source data
Unlabeled target data
Source Domain
Data Stochastic
Unlabeled Target
Data Stochastic
Labeled Target
Data Stochastic
Labeled target data
Training Procedure
MLP pseudo labels
Inter-pseudo labels
Intra-pseudo labels
Inter-view GCN Classifier $\left(G_{gcn}^{inter}\right)$
ce: cross-entropy
bce: binary cross-entropy
Inter Edge Network
Affinity Matrix
Inter Node Classifier
$\mathcal{L}_{ce}^{interN}$
$\mathcal{L}_{bce}^{interE}$
$E\left(x_{\mathcal{S}}^{w},x_{\mathcal{S}}^{str}\right)$
$E\left(x_{\mathcal{T}_u}^{w},x_{\mathcal{T}_u}^{str}\right)$
Feature Extractor (E)
$E\left(x_{\mathcal{S}}^{w},x_{\mathcal{T}_l}^{w},x_{\mathcal{T}_u}^{w}\right)$
$E\left(x_{\mathcal{S}}^{str},x_{\mathcal{T}_l}^{str},x_{\mathcal{T}_u}^{str}\right)$
MLP Classifier $\left(F_{mlp}\right)$
Minimized Class Confusion
$\mathcal{L}_{ce}^{mlp}$
$\mathcal{L}_{mcc}^{mlp}$
$E\left(x_{\mathcal{T}_l}^{w},x_{\mathcal{T}_l}^{str}\right)$
$E\left(x_{\mathcal{T}_u}^{w},x_{\mathcal{T}_u}^{str}\right)$
Intra Edge Network
$\mathcal{L}_{bce}^{intraE}$
Affinity Matrix
Intra Node Classifier
$\mathcal{L}_{ce}^{intraN}$
MLP output
Inter-view GCN output
Intra-view GCN output
Intra-view GCN Classifier $\left(G_{gcn}^{intra}\right)$
Training Strategy
F_{mlp}
MLP pseudo label
MLP pseudo label
G_{gcn}^{inter}
Inter pseudo label
G_{gcn}^{intra}
Intra pseudo label
Inter pseudo label
Intra pseudo label
F_{mlp}
MLP pseudo label
MLP pseudo label
G_{gcn}^{inter}
Inter pseudo label
G_{gcn}^{intra}
Intra pseudo label

Figure 2: The pipeline of our method TriCT (left) and the training strategy (right). We use three classifiers, including a multilayer perceptron classifier F_{mlp}, an inter-view GCN G_{gcn}^{inter} and an intra-view GCN G_{gcn}^{intra} that share a feature extractor E. TriCT consists of three pairs of classifiers $\{F_{mlp}, G_{gcn}^{inter}\}$, $\{F_{mlp}, G_{gcn}^{intra}\}$, and $\{G_{gcn}^{inter}, G_{gcn}^{intra}\}$ which teach each other by exchanging their generated pseudo labels to boost classification performance on the target domain.

with a few labeled target data $\mathcal{D}_{\mathcal{T}_l} = \{x_{\mathcal{T}_l}^i, y_{\mathcal{T}_l}^i\}_{i=1}^{\mathcal{N}_{\mathcal{T}_l}}$, and the unlabeled target data $\mathcal{D}_{\mathcal{T}_u} = \{x_{\mathcal{T}_u}^i\}_{i=1}^{\mathcal{N}_{\mathcal{T}_u}}$, where $x_{\mathcal{S}}$, $x_{\mathcal{T}_l}$, and $x_{\mathcal{T}_u}$ are the image sets of the source, labeled target, and unlabeled target having $\mathcal{N}_{\mathcal{S}}$, $\mathcal{N}_{\mathcal{T}_l}$, and $\mathcal{N}_{\mathcal{T}_u}$ samples, respectively, where $\mathcal{N}_{\mathcal{T}_u} \gg \mathcal{N}_{\mathcal{T}_l}$. The source and target data share the same label vector $\mathbf{y}_{\mathcal{S}}$, $\mathbf{y}_{\mathcal{T}_l} \in \{1, ..., K\}$, where K is the number of classes. We train the model on the labeled set $\mathcal{D}_l = \{\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}_l}\}$ and the unlabeled set $\mathcal{D}_{\mathcal{T}_u}$; and evaluate the trained model on $\mathcal{D}_{\mathcal{T}_u}$.

Data preparation. We apply two different stochastic data transformations: $Aug_w(\cdot)$ and $Aug_{str}(\cdot)$ to a given training input image $x \in \{\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}_l}, \mathcal{D}_{\mathcal{T}_u}\}$, where $Aug_w(\cdot)$ is the weak augmentation function that uses the light perturbation such as random horizontal flipping and random cropping, and $Aug_{str}(\cdot)$ is the strong augmentation function that utilizes the RandAugment [2] including 14 transformation techniques. For example, a source image $x_{\mathcal{S}}^i$ has two transformations $x_{\mathcal{S}}^{i,w}$ and $x_{\mathcal{S}}^{i,str}$ with the same label $y_{\mathcal{S}}^i$. Similarly, $x_{\mathcal{T}_l}^{i,w}$ and $x_{\mathcal{T}_l}^{i,str}$ are weakly and strongly augmented images of a labeled target image $x_{\mathcal{T}_l}^i$ corresponding to label $y_{\mathcal{T}_l}^i$. In contrast, an unlabeled target image $x_{\mathcal{T}_u}^i$ has two versions $x_{\mathcal{T}_u}^{i,w}$ and $x_{\mathcal{T}_u}^{i,str}$ without label information. We sample the training set into multiple mini-batches with size B.

Feature extractor. We use a shared feature extractor $E(\cdot;\theta_E) : \mathbb{R}^{3\times w\times h} \to \mathbb{R}^d$, parameterized by θ_E, to encode the features of input images, where w and h are the width and height of an input image; and d is the output feature dimension.

MLP classifier. The classifier $F_{mlp}(\cdot;\theta_{mlp}) : \mathbb{R}^d \to \mathbb{R}^K$ is the non-linear network, parameterized by θ_{mlp}, consisting of fully connected layers, with output logits dimension of a single image as $\mathbb{R}^K$.

GCN classifier. We use graph convolutional network (GCN) classifier [12] to exploit the relation of all samples in each mini-batch. The GCN classifier consists of an edge network f_E and a node network f_N. Specifically, f_E is used to collect neighboring node information by estimating the similarity among samples, while f_N is used to aggregate all neighbor features.

Following [12], all images in each mini-batch can be formed as an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the node set, $\mathcal{E}$ is the edge set. The feature vector of each image in source and target domains extracted by the shared feature extractor E is represented as a node $\mathbf{v}_i \in \mathcal{V}$. $e_{i,j} \in \mathcal{E}$ denotes an edge between nodes $\mathbf{v}_i$ and $\mathbf{v}_j$ that measures the node affinity represented by the similarity score $\hat{a}_{i,j}$ as follows:

$$\hat{a}_{i,j}^{(l)} = \sigma\big(f_E^{(l)}\big(\|\mathbf{v}_i^{(l-1)} - \mathbf{v}_j^{(l-1)}\|; \theta_{edge}^{(l)}\big)\big), \qquad (1)$$

where σ is the sigmoid function, $\mathbf{v}_i^{(l-1)}$ and $\mathbf{v}_j^{(l-1)}$ are feature vectors of samples x_i and x_j at $(l-1)$-th layer, where features $\mathbf{v}_i^{(0)}$ and $\mathbf{v}_j^{(0)}$ at the initial layer are extracted by the shared feature extractor E, respectively. $f_E^{(l)}$ is parameterized by $\theta_{edge}^{(l)}$ at the l-th layer and is used to estimate the similarity of nodes $\mathbf{v}_i$ and $\mathbf{v}_j$ at layer $(l-1)$. The similarity score $\hat{a}_{i,j}$ of a pair (x_i, x_j) in each mini-batch is an element of the unnormalized affinity matrix $\hat{\mathcal{A}}$. Then, the self-connections of nodes are added to this matrix, and it is normalized as follows:

$$\mathcal{A}^{(l)} = D^{-\frac{1}{2}}(\hat{\mathcal{A}}^{(l)} + I)D^{-\frac{1}{2}}, \qquad (2)$$

where D is the degree matrix of $\hat{\mathcal{A}}^{(l)} + I$, I is the identity matrix used to compute self-connections, and $\mathcal{A}^{(l)}$ is the normalized affinity matrix.

Based on the information provided by the given affinity matrix $\mathcal{A}^{(l-1)}$, the node features are updated at the l-th layer as follows:

$$\mathbf{v}_i^{(l)} = f_N^{(l)}\left(\left[\mathbf{v}_i^{(l-1)}; \sum_{j\in B} a_{i,j}^{(l-1)} \cdot \mathbf{v}_j^{(l-1)}\right]; \theta_{node}^{(l)}\right), \quad (3)$$

where $f_N^{(l)}$ is parameterized by $\theta_{node}^{(l)}$ at the l-th layer. $a_{i,j}^{(l-1)} \in \mathcal{A}^{(l-1)}$ is utilized in $f_N^{(l)}$ to consider the relationship between nodes $\mathbf{v}_i$ and $\mathbf{v}_j$. $[\cdot;\cdot]$ indicates the concatenation function for aggregating features. B is the number of samples in a mini-batch. In Eq. (3), the inputs of f_N consisting of the node feature extracted from the shared feature extractor E, $\mathbf{v}_i^{(0)} = E(x_i)$, and the neighbour features calculated by using the information from f_E. The output dimension of f_N is $\mathbb{R}^K$.

Without loss of generality, we explain the training procedure of the proposed method at the initial step t_0 with the first mini-batch $\mathcal{B}^0$ as an example. The operation of our method performed on all mini-batches is the same. We use two GCN classifiers to exploit information from the source and labeled target datasets separately. The first inter-view GCN classifier G_{gcn}^{inter} is trained on the source data $\mathcal{B}_{\mathcal{S}}^0 = \{x_{\mathcal{S}}^{i,w}, x_{\mathcal{S}}^{i,str}\}_{i=1}^{B}$ with $\{y_{\mathcal{S}}^i\}_{i=1}^{B}$. Similarly, we use the labeled target data $\mathcal{B}_{\mathcal{T}_l}^0 = \{x_{\mathcal{T}_l}^{i,w}, x_{\mathcal{T}_l}^{i,str}\}_{i=1}^{B}$ with $\{y_{\mathcal{T}_l}^i\}_{i=1}^{B}$ to train the second intra-view GCN classifier G_{gcn}^{intra}. The labeled set integrated from the source and labeled target samples $\mathcal{B}_l^0 = \{x_l^{i,w}, x_l^{i,str}\}_{i=1}^{2B} = \{x_{\mathcal{S}}^{i,w}, x_{\mathcal{S}}^{i,str}\}_{i=1}^{B} \cup \{x_{\mathcal{T}_l}^{i,w}, x_{\mathcal{T}_l}^{i,str}\}_{i=1}^{B}$ with $\{y_l^i\}_{i=1}^{2B} = \{y_{\mathcal{S}}^i\}_{i=1}^{B} \cup \{y_{\mathcal{T}_l}^i\}_{i=1}^{B}$, is used to train the MLP classifier F_{mlp}. Finally, F_{mlp}, G_{gcn}^{inter}, and G_{gcn}^{intra} generate pseudo labels on the unlabeled target data $\mathcal{B}_{\mathcal{T}_u}^0 = \{x_{\mathcal{T}_u}^{i,w}, x_{\mathcal{T}_u}^{i,str}\}_{i=1}^{B}$. The data preparation and the pipeline of the training operation are illustrated in Figure 2.

3.1. Supervised training on labeled samples

MLP classifier. Both weak and strong augmentation versions of labeled source and target data are fed into E to extract corresponding features. We obtain the predictions of these features by passing them through the classifier F_{mlp}. Then, we use the standard cross-entropy loss on the labeled samples with ground-truth labels as follows:

$$\mathcal{L}_{ce}^{mlp} = -\sum_{i=1}^{2B}\left(y_l^i \log p\left(x_l^{i,w}\right) + y_l^i \log p\left(x_l^{i,str}\right)\right), \quad (4)$$

where $p(x_l^{i,w}) = \text{softmax}(F_{mlp}(E(x_l^{i,w})))$ and $p(x_l^{i,str}) = \text{softmax}(F_{mlp}(E(x_l^{i,str})))$ are the predictions of two augmentation versions, respectively, with $(x_l^{i,w}, y_l^i), (x_l^{i,str}, y_l^i) \in \mathcal{B}_l^0$.

GCN classifier. The training procedures for both G_{gcn}^{inter} and G_{gcn}^{intra} are the same. Specifically, we use the standard cross-entropy loss to update the node network f_N as follows:

$$\mathcal{L}_{ce}^{N} = -\sum_{i=1}^{B}\left(y^i \log p\left(x^{i,w}\right) + y^i \log p\left(x^{i,str}\right)\right), \quad (5)$$

where $p(x^{i,w}) = \text{softmax}(f_N(E(x^{i,w})))$ and $p(x^{i,str}) = \text{softmax}(f_N(E(x^{i,str})))$ are the output predictions of f_N. We set $x^{i,w} = x_{\mathcal{S}}^{i,w}$, $x^{i,str} = x_{\mathcal{S}}^{i,str}$, and $y^i = y_{\mathcal{S}}^i$ for training the inter-node network $f_N = f_{interN}$ in G_{gcn}^{inter} using the loss function $\mathcal{L}_{ce}^{N} = \mathcal{L}_{ce}^{interN}$. Likely, we set $x^{i,w} = x_{\mathcal{T}_l}^{i,w}$, $x^{i,str} = x_{\mathcal{T}_l}^{i,str}$, and $y^i = y_{\mathcal{T}_l}^i$ for training the intra-node network $f_N = f_{intraN}$ in G_{gcn}^{intra} associated to the loss function $\mathcal{L}_{ce}^{N} = \mathcal{L}_{ce}^{intraN}$.

We use the binary cross-entropy loss to train the edge network f_E as follows:

$$\mathcal{L}_{bce}^{E} = e_{i,j} \log a_{i,j} + \left(1 - e_{i,j}\right) \log\left(1 - a_{i,j}\right), \quad (6)$$

where $e_{i,j}$ denotes the ground-truth edge between samples x^i and x^j. $e_{i,j} = 1$ if only if $y^i = y^j$ that means x^i and x^j belong to the same category, and $e_{i,j} = 0$ otherwise. $a_{i,j}$ is the output prediction of f_E that indicates the similarity score between x^i and x^j.

We set $x^i = x_{\mathcal{S}}^i$, $x^j = x_{\mathcal{S}}^j$, and $e_{i,j} = e_{i,j}^{\mathcal{S}}$ corresponding to $\mathcal{L}_{bce}^{E} = \mathcal{L}_{bce}^{interE}$ for training the inter-edge network $f_E = f_{interE}$ in G_{gcn}^{inter}. Similarly, we set $e_{i,j} = e_{i,j}^{\mathcal{T}_l}$, $x^i = x_{\mathcal{T}_l}^i$ and $x^j = x_{\mathcal{T}_l}^j$ corresponding to $\mathcal{L}_{bce}^{E} = \mathcal{L}_{bce}^{intraE}$ for training the intra-edge network $f_E = f_{intraE}$ in G_{gcn}^{intra}. The output of f_E is used as the input of f_N for neighbor feature aggregation as in Eq. (3).

The cost functions used to train G_{gcn}^{inter} and G_{gcn}^{intra} are calculated as follows:

$$\mathcal{L}_{gcn}^{inter} = \alpha\mathcal{L}_{ce}^{interN} + \beta\mathcal{L}_{bce}^{interE}, \quad (7)$$

$$\mathcal{L}_{gcn}^{intra} = \alpha\mathcal{L}_{ce}^{intraN} + \beta\mathcal{L}_{bce}^{intraE}, \quad (8)$$

where α and β are the weights to control the influence of the node and edge networks. G_{gcn}^{inter} can obtain the transferable knowledge from the source domain by adopting Eq. (7), while G_{gcn}^{intra} can preserve the discriminative representations within the target domain by minimizing Eq. (8).

Feature extractor. The objective cost function to train the shared feature extractor on labeled samples is calculated as follows:

$$\mathcal{L}_{cls}^{E} = \mathcal{L}_{ce}^{mlp} + \mathcal{L}_{gcn}^{inter} + \mathcal{L}_{gcn}^{intra}. \quad (9)$$

3.2. Trico-training strategy on unlabeled samples

The first co-training between $F_{mlp} - G_{gcn}^{inter}$ for minimizing the inter-domain discrepancy. In this training process, we use the reliable pseudo labels (PSs) generated from f_{interN} in G_{gcn}^{inter} to train F_{mlp} to encourage the transferrable knowledge from the source domain to the target domain which can minimize the inter-domain discrepancy.

Firstly, the pseudo label created by f_{interN} is calculated as follows:

$$\hat{y}^i_{interPS} = \text{argmax}\left(\mathbf{p}^{i,w}_{interN}\right),\ if\ \max\left(\mathbf{p}^{i,w}_{interN}\right) > \tau, \tag{10}$$

where τ is the predefined threshold value, $\mathbf{p}^{i,w}_{interN} = p(x^{i,w}_{\mathcal{T}_u}) = \text{softmax}(f_{interN}(E(x^{i,w}_{\mathcal{T}_u})))$ is the prediction vector of f_{interN} on $x^{i,w}_{\mathcal{T}_u}$; it then is converted to the one-hot hard label $\hat{y}^i_{interPS} = \text{argmax}(\mathbf{p}^{i,w}_{interN})$ to train F_{mlp} by using the cross-entropy loss as follows:

$$\mathcal{L}^{interN\rightarrow mlp}_{ce} = -\sum_{i=1}^{B} \mathbb{1}\left[\max\left(\mathbf{p}^{i,w}_{interN}\right) > \tau\right].\hat{y}^i_{interPS}\log\left(\mathbf{q}^{i,str}_{mlp}\right), \tag{11}$$

where $\mathbf{q}^{i,str}_{mlp} = p(x^{i,str}_{\mathcal{T}_u}) = \text{softmax}(F_{mlp}(E(x^{i,str}_{\mathcal{T}_u})))$ is the prediction vector of $x^{i,str}_{\mathcal{T}_u}$.

In G^{inter}_{gcn}, the output of f_{interE} is used as the input of f_{interN}, which means the quality and quantity of PS generated by f_{interN} rely much on f_{interE}. As mentioned in Eq. (6), f_{interE} is trained by using the ground-truth edges that are determined by the information from the labeled data. However, we cannot access the label information in the unlabeled target data.

To solve this problem, we introduce a novel concept named '*pseudo-edge*' using F_{mlp} back to train f_{interE} that encourages f_{interE} to explore the correlation of all unlabeled target samples effectively. Firstly, F_{mlp} generates its pseudo label as follows:

$$\hat{y}^i_{mlp} = \text{argmax}\left(\mathbf{p}^i_{mlp}\right),\ if\ \max\left(\mathbf{p}^i_{mlp}\right) > \tau, \tag{12}$$

where $\mathbf{p}^i_{mlp} = p(x^i_{\mathcal{T}_u}) = \text{softmax}(F_{mlp}(E(x^i_{\mathcal{T}_u})))$ with $x^i_{\mathcal{T}_u} \in \mathcal{B}^0_{\mathcal{T}_u}$. Then, similar to Eq. (6), it is assumed that two unlabeled samples $x^i_{\mathcal{T}_u}$ and $x^j_{\mathcal{T}_u}$ have a high similarity score if their PSs belong to the same class ($\hat{y}^i_{mlp} = \hat{y}^j_{mlp}$) associated with $e^{\mathcal{T}_u}_{i,j} = 1$, and $e^{\mathcal{T}_u}_{i,j} = 0$ otherwise, where $e^{\mathcal{T}_u}_{i,j}$ is called '*pseudo-edge*'. Finally, we can use $e^{\mathcal{T}_u}_{i,j}$ as the ground truth of the edge map indicating the relationship between two unlabeled target samples $x^i_{\mathcal{T}_u}$ and $x^j_{\mathcal{T}_u}$. Then, the training process to update f_{interE} is conducted similar to Eq. (6) as follows:

$$\mathcal{L}^{mlp\rightarrow interE}_{bce} = e^{\mathcal{T}_u}_{i,j}\log\left(\tilde{a}^{\mathcal{T}_u}_{i,j}\right) + (1 - e^{\mathcal{T}_u}_{i,j})\log\left(1 - \tilde{a}^{\mathcal{T}_u}_{i,j}\right), \tag{13}$$

where $\tilde{a}^{\mathcal{T}_u}_{i,j}$ is the output prediction of f_{interE} that estimates the similarity score between two unlabeled target samples $x^i_{\mathcal{T}_u}$ and $x^j_{\mathcal{T}_u}$.

Finally, the loss function of the first co-training is calculated as follows:

$$\mathcal{L}^{inter}_{Co} = \mathcal{L}^{interN\rightarrow mlp}_{ce} + \mathcal{L}^{mlp\rightarrow interE}_{bce}. \tag{14}$$

The second co-training between $F_{mlp} - G^{intra}_{gcn}$ for minimizing the intra-domain discrepancy. This co-training strategy is conducted similarly to the first co-training $F_{mlp} - G^{inter}_{gcn}$ by exploiting the interaction between F_{mlp} and $G^{intra}_{gcn} = \{f_{intraN}, f_{intraE}\}$, which is calculated as follows:

$$\mathcal{L}^{intra}_{Co} = \mathcal{L}^{intraN\rightarrow mlp}_{ce} + \mathcal{L}^{mlp\rightarrow intraE}_{bce}. \tag{15}$$

To improve the reliability of PS generated by the classifier F_{mlp}, we use a cost function that is inspired by [7] to minimize class confusion (MCC) to increase the distinguishability of F_{mlp} on unlabeled target features as follows:

$$\mathcal{L}^{mlp}_{mcc} = \frac{1}{K}\sum_{k=1}^{K}\sum_{k\neq k'}^{K}\left|\tilde{C}_{kk'}\right|, \tag{16}$$

where $|\tilde{C}_{kk'}|$ is a normalized softened probability that measures the confusion level between classes k and k'.

The third co-training between $G^{inter}_{gcn} - G^{intra}_{gcn}$ for alleviating imbalanced classification accuracy. G^{inter}_{gcn} is trained on a large amount of data, while G^{intra}_{gcn} is trained on a small amount of data. Thus, an imbalance in the classification accuracy can occur between these classifiers. To solve this problem, we propose the third co-training strategy, in which the inter-pseudo label set $PS_{inter} = \{x^{i,w}_{\mathcal{T}_u}, \hat{y}^i_{interPS}\}^{\mathcal{N}^{PS}_{inter}}_{i=1}$, having $\mathcal{N}^{PS}_{inter}$ pseudo labels generated by the inter-node network f_{interN} in G^{inter}_{gcn}, is added into the labeled target set $\mathcal{D}_{\mathcal{T}_l}$ to train the classifier G^{intra}_{gcn} as follows:

$$\mathcal{D}^{q+1}_{\mathcal{T}_l} = \mathcal{D}_{\mathcal{T}_l} \cup PS^q_{inter},\ \text{with}\ \mathcal{D}^0_{\mathcal{T}_l} = \mathcal{D}_{\mathcal{T}_l}, \tag{17}$$

where q is the training interval. Similarly, we train G^{inter}_{gcn} with the new labeled set as follows:

$$\mathcal{D}^{q+1}_{\mathcal{S}} = \mathcal{D}_{\mathcal{S}} \cup PS^q_{intra},\ \text{with}\ \mathcal{D}^0_{\mathcal{S}} = \mathcal{D}_{\mathcal{S}}, \tag{18}$$

where $PS_{intra} = \{x^{i,w}_{\mathcal{T}_u}, \hat{y}^i_{intraPS}\}^{\mathcal{N}^{PS}_{intra}}_{i=1}$ is the intra-pseudo label set, having $\mathcal{N}^{PS}_{intra}$ pseudo labels generated by the intra-node network f_{intraN} in G^{intra}_{gcn}. Finally, PS_{inter} and PS_{intra} are combined to add into the labeled set $\mathcal{D}_l$ for training the classifier F_{mlp} as follows:

$$\mathcal{D}^{q+1}_{l} = \mathcal{D}_{l} \cup PS^q_{inter} \cup PS^q_{intra},\ \text{with}\ \mathcal{D}^0_{l} = \mathcal{D}_{l}. \tag{19}$$

These new labeled datasets are used to update the loss functions of the supervised training in Section 3.1.

4. Experiments

Datasets. We used three standard SSDA benchmark datasets to evaluate the effectiveness of the proposed approach TriCT including *Office-31* [30], *Office-Home* [22],

Method	R→C		R→P		P→C		C→S		S→P		R→S		P→R		Mean	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
MME	70.0	72.2	67.7	69.7	69.0	71.7	56.3	61.8	64.8	66.8	61.0	61.9	76.1	78.5	66.4	68.9
BiAT	73.0	74.9	68.0	68.8	71.6	74.6	57.9	61.5	63.9	67.5	58.5	62.1	77.0	78.6	67.1	69.7
UODA	72.7	75.4	70.3	71.5	69.8	73.2	60.5	64.1	66.4	69.4	62.7	64.2	77.3	80.8	68.5	71.2
Con^{2}DA	71.3	74.2	71.8	72.1	71.1	75.0	60.0	65.7	63.5	67.1	65.2	67.1	75.7	78.6	68.4	71.4
APE	70.4	76.6	70.8	72.1	72.9	76.7	56.7	63.1	64.5	66.1	63.0	67.8	76.6	79.4	67.6	71.7
S^{3}D	73.3	75.9	68.9	72.1	73.4	75.1	60.8	64.4	68.2	70.0	65.1	66.7	79.5	80.3	69.9	72.1
STar	74.1	77.1	71.3	73.2	71.0	75.8	63.5	67.8	66.1	69.2	64.1	67.9	80.0	81.2	70.0	73.2
PAC	74.9	78.6	73.0	74.3	72.6	76.0	65.8	69.6	67.9	69.4	68.7	70.2	76.7	79.3	71.4	73.9
MAP-F	75.3	77.0	74.0	75.0	74.3	77.0	65.8	69.5	73.0	73.3	67.5	69.2	81.7	83.3	73.1	74.9
CLDA	76.1	77.7	75.1	75.7	71.0	76.4	63.7	69.7	70.2	73.7	67.1	71.1	80.1	82.9	71.9	75.3
DECOTA	79.1	80.4	74.9	75.2	76.9	78.7	65.1	68.6	72.0	72.7	69.7	71.9	79.6	81.5	73.9	75.6
CDAC	77.4	79.6	74.2	75.1	75.5	79.3	67.6	69.9	71.0	73.4	69.2	72.5	80.4	81.9	73.6	76.0
ECACL	75.3	79.0	74.1	77.3	75.3	79.4	65.0	70.6	72.1	74.6	68.1	71.6	79.7	82.4	72.8	76.4
MCL	77.4	79.4	74.6	76.3	75.5	78.8	66.4	70.9	74.0	74.7	70.7	72.3	82.0	83.3	74.4	76.5
MVCL	78.8	79.8	76.0	77.4	78.0	80.3	70.8	73.0	75.1	76.7	72.4	74.4	82.4	85.1	76.2	78.1
DEEM	79.7	80.5	78.1	79.0	77.0	77.5	**71.9**	74.9	77.7	80.0	76.7	75.9	**85.4**	88.5	78.1	79.5
TriCT	**86.5**	**89.1**	**85.3**	**86.6**	**80.4**	**86.3**	71.0	**79.9**	**80.3**	**84.5**	**78.9**	**82.1**	82.2	**90.1**	**80.7**	**85.5**

Table 1: Accuracy (%) on *DomainNet* under 1-shot and 3-shot settings extracted by the ResNet-34 backbone network.

and *DomainNet* [18]. *Office-31* is a small dataset having three different domains such as *DSLR* (D), *Webcam* (W), and *Amazon* (A) with 31 classes in each domain. *Office-Home* is a moderate-size dataset consisting of four domains: *Real-World* (R), *Clipart* (C), *Product* (P), and *Art* (A), with 65 classes in each domain. Following [8, 10, 11, 23, 32], we selected four domains such *Real* (R), *Clipart* (C), *Sketch* (S), and *Painting* (P) on the *DomainNet* dataset with 126 classes in each domain to extract the results of seven different domain adaptation tasks.

Implementation details. We used Pytorch [17] as the platform to implement the proposed method. For the feature extractor, we utilized ResNet-34 [5] and AlexNet [9] as the backbone networks that are pre-trained on the ImageNet [3] dataset. Following [8, 10, 11, 23, 32], we used the MLP classifier with two fully connected layers for ResNet-34 and a single fully connected layer for AlexNet with a normalization layer, while we selected the GCN classifier as in [12]. We used a Stochastic Gradient Descent (SGD) as an optimizer with an initial learning rate of 0.0005, a momentum of 0.9, and a weight decay of 0.0005. The size of the mini-batch was set to $B = 24$ for ResNet-34 and $B = 32$ for AlexNet. The weight factors in Eqs. (7) and (8) were set to $\alpha = 0.3$ and $\beta = 1.0$, and the threshold value for pseudo-label selection was set to $\tau = 0.8$. We conducted all experiments on a GeForce RTX3090 GPU.

We have noticed that the GCN classifier required a mini-batch for extracting classification results; therefore, for a fair comparison to previous works, we only utilized GCN classifiers as the auxiliary models to support the MLP classifier, and then we used the MLP classifier to extract the final experimental results.

State-of-the-art (SOTA) methods. We compared TriCT with the previous SOTA SSDA methods: MME [23], BiAT [6], UODA [20], Con^{2}DA [19], APE [8], S^{3}D [33], STar [26], PAC [14], MAP-F [16], CLDA [25], DECOTA [32], CDAC [10], ECACL [11], MCL [31], MVCL [15], and DEEM [13].

4.1. Comparison with SOTA methods

For a fair comparison, we used ResNet-34 and AlexNet as the backbone networks to extract results on *DomainNet*, *Office-Home*, and *Office-31* under 1-shot and 3-shot settings. The experimental results were reported in Tables 1, 2, and 3. Due to the limited space, we included other details of the experimental results in the supplementary material.

Comparisons on DomainNet. Following the standard evaluation protocol [10, 15, 23, 25], we reported classification performance on the target domain over seven SSDA scenarios in Table 1. As shown in this table, the proposed method outperformed the second-best method DEEM [13] by 2.6% and 6.0% on the average results of *DomainNet* under the 1-shot and 3-shot settings using ResNet-34 as the backbone network, respectively. Similar to our method, CLDA [25] and CDAC [10] were proposed to minimize inter-domain and intra-domain discrepancies simultaneously. Compared to these approaches, the average classification accuracy of our method surpassed CLDA by 8.4% and 10.2% under 1-shot and 3-shot settings, while it improved over CDAC by 6.7% and 9.5% in 1-shot and 3-shot settings, respectively.

Comparisons on Office-Home and Office-31. The classification results on the *Office-Home* and *Office-31* were reported in Tables 2 and 3. As shown in these tables, the classification results on the target domain of the proposed method achieved the highest accuracy in all domain adaptation scenarios. As shown in Table 2, the average classification result of our approach on the *Office-Home* dataset under the 3-shot setting recorded extensive improvements: 6.7% compared to ECACL [11] using AlexNet, and 9.6% compared to MCL [31] using ResNet-34. Similarly, as listed in Table 3, the average classification accuracy of

Net	Method	R→C	R→P	R→A	P→R	P→C	P→A	A→P	A→C	A→R	C→R	C→A	C→P	Mean
AlexNet	PAC	58.9	72.4	47.5	61.9	53.2	39.6	63.8	49.9	60.0	54.5	36.3	64.8	55.2
	APE	51.9	74.6	51.2	61.6	47.9	42.1	65.5	44.5	60.9	58.1	44.3	64.8	55.6
	Con^{2}DA	52.3	73.5	49.1	64.4	49.3	38.2	66.4	47.7	62.4	59.9	39.9	66.1	55.8
	CDAC	54.9	75.8	51.8	64.3	51.3	43.6	65.1	47.5	63.1	63.0	44.9	65.6	56.8
	MVCL	55.4	73.1	54.6	65.6	49.9	44.7	66.0	47.9	64.5	59.7	42.9	63.3	57.3
	CLDA	51.5	74.1	54.3	67.0	47.9	47.0	65.8	47.4	66.6	64.1	46.8	67.5	58.3
	ECACL	55.4	75.7	56.0	67.0	52.5	46.4	67.4	48.5	66.3	60.8	45.9	67.3	59.1
	TriCT	**59.9**	**84.2**	**61.8**	**76.6**	**55.4**	**50.6**	**75.3**	**55.4**	**74.4**	**69.6**	**49.6**	**76.3**	**65.8**
ResNet-34	MME	64.6	85.5	71.3	80.1	64.6	65.5	79.0	63.6	79.7	76.6	67.2	79.3	73.1
	APE	66.4	86.2	73.4	82.0	65.2	66.1	81.1	63.9	80.2	76.8	66.6	79.9	74.0
	CDAC	67.8	85.6	72.2	81.9	67.0	67.5	80.3	65.9	80.6	80.2	67.4	81.4	74.2
	CLDA	66.0	87.6	76.7	82.2	63.9	72.4	81.4	63.4	81.3	80.3	70.5	80.9	75.5
	MVCL	69.6	88.1	76.4	80.9	66.0	71.2	82.0	66.0	79.6	79.6	67.4	80.7	75.6
	DECOTA	70.4	87.7	74.0	82.1	68.0	69.9	81.8	64.0	80.5	79.0	68.0	83.2	75.7
	MCL	70.1	88.1	75.3	83.0	68.0	69.9	83.9	67.5	82.4	81.6	71.4	84.3	77.1
	TriCT	**81.9**	**94.1**	**86.3**	**92.3**	**78.7**	**83.4**	**91.1**	**76.9**	**91.3**	**91.8**	**80.5**	**92.5**	**86.7**

Table 2: Accuracy (%) on *Office-Home* under 3-shot setting extracted by AlexNet and ResNet-34 backbone networks.

Net	Method	W→A		D→A		Mean	
		1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
AlexNet	PAC	53.6	65.1	54.7	66.3	54.2	65.7
	MME	57.2	67.3	55.8	67.8	56.5	67.6
	BiAT	57.9	68.2	54.6	68.5	56.3	68.4
	STar	59.8	69.1	56.8	69.0	58.3	69.1
	Con^{2}DA	58.3	69.8	56.2	69.7	57.3	69.8
	CDAC	63.4	70.1	62.8	70.0	63.1	70.1
	CLDA	64.6	70.5	62.7	72.5	63.6	71.5
	TriCT	**67.8**	**78.6**	**63.8**	**77.7**	**65.8**	**78.2**

Table 3: Accuracy (%) on *Office-31* under 1-shot and 3-shot settings extracted by the AlexNet backbone network.

Components					Domain adaptation tasks			Mean
$\mathcal{L}_{ce}^{mlp}$	$\mathcal{L}_{mcc}^{mlp}$	$\mathcal{L}_{Co}^{intra}$	$\mathcal{L}_{Co}^{inter}$	$G_{gcn}^{inter} - G_{gcn}^{intra}$	R→P	C→S	S→P	
✓					63.7	56.5	62.7	61.0
✓	✓				70.3	63.5	68.7	67.5
✓	✓		✓		83.9	77.8	80.5	80.7
✓	✓	✓			82.2	71.8	78.3	77.4
✓	✓	✓	✓	✓	86.6	79.9	84.5	83.6
✓		✓	✓	✓	85.3	77.8	81.4	81.5

Table 4: Ablation study on the *DomainNet* dataset with three domain adaptation tasks $R{\rightarrow}P$, $C{\rightarrow}S$ and $S{\rightarrow}P$ under 3-shot settings using ResNet-34 as a backbone network.

our method using AlexNet outperformed the second-best method CLDA [25] by 2.2% and 6.7% on *Office-31* under 1-shot and 3-shot settings, respectively.

4.2. Analysis

Ablation studies. We conducted ablation studies to evaluate the effectiveness of each model in the proposed method over three domain adaptation tasks $R{\rightarrow}P$, $C{\rightarrow}S$ and $S{\rightarrow}P$ on *DomainNet* under 3-shot settings using ResNet-34, as shown in Table 4. Firstly, the baseline model, including the shared feature extractor E and F_{mlp}, was trained on the labeled source and target samples using $\mathcal{L}_{ce}^{mlp}$ in Eq. (4) and then tested on the unlabeled target data. Obviously, the average classification accuracy of this case was the lowest, with 61.0%. Secondly, the classification performance was improved by 6.5%, when we added the MCC loss function, $\mathcal{L}_{mcc}^{mlp}$ in Eq. (16), to train with the baseline model. Thirdly, we investigated the effectiveness of inter-view GCN and intra-view GCN classifier models with $\mathcal{L}_{Co}^{inter}$ and $\mathcal{L}_{Co}^{intra}$ in Eq. (14) and Eq. (15), respectively. The average classification results were significantly increased by 19.7% (with inter-view GCN), and 16.4% (with intra-view GCN) compared to the baseline model. However, the average classification performance on the target domain only reached the optimal result of 83.6% when both GCN classifiers were used with the third co-training $G_{gcn}^{inter} - G_{gcn}^{intra}$. The final experiment was implemented to emphasis that the MCC loss function $\mathcal{L}_{mcc}^{mlp}$ is necessary in the proposed method for minimizing the class confusion. The average classification was degraded by 2.1% when $\mathcal{L}_{mcc}^{mlp}$ was removed.

Effectiveness of Trico-training. We used the extracted classification results of the domain adaptation task $P{\rightarrow}R$

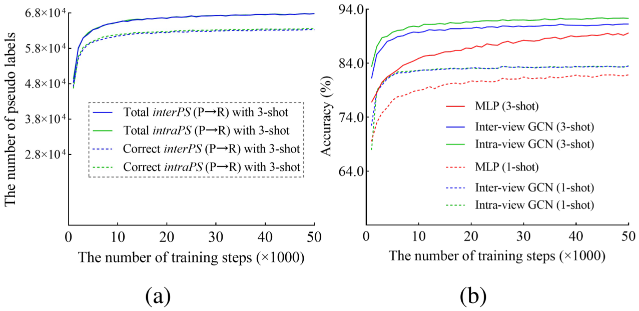
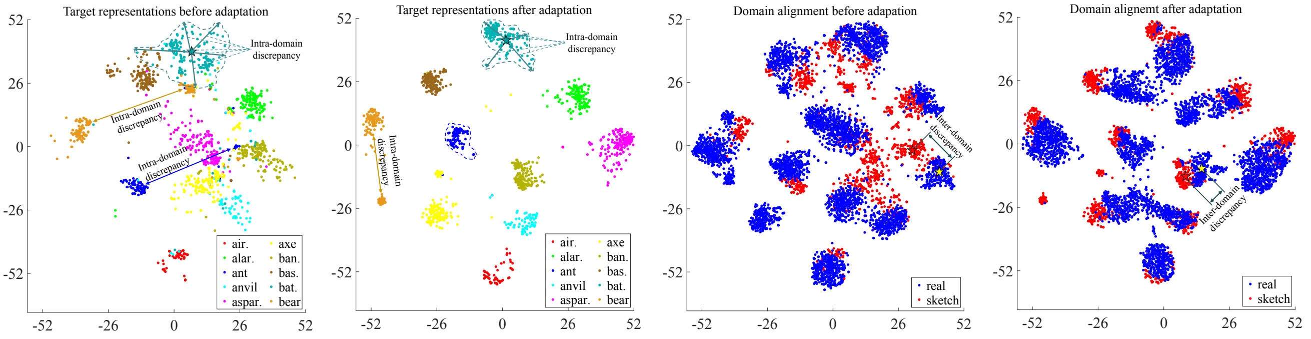


Figure 3: (a) The quantity and quality of the pseudo labels generated by TriCT on *DomainNet* under the 3-shot setting of $P{\rightarrow}R$ using ResNet-34. (b) The classification accuracy of $P{\rightarrow}R$ on *DomainNet* using ResNet-34 under the 1-shot and 3-shot settings.



(a)

(b)

(c)

(d)

Figure 4: t-SNE [29] visualization of the feature embeddings extracted by TriCT using ResNet-34. We selected 10 classes of the domain adaptation task $R \to S$ on *DomainNet* under the 3-shot setting. (a) and (b) illustrated the features of the unlabeled target data before and after adaptation, respectively. (c) and (d) displayed domain alignment between source and target domains before and after adaptation, respectively.

on *DomainNet* using ResNet-34 under 1-shot and 3-shot settings to analyze the effectiveness of TriCT. As shown in Figure 3a, GCN classifiers showed the impressive accuracy of generated pseudo labels up to 93.76% and 94.28% extracted by inter-view GCN and intra-view GCN classifiers under the 3-shot setting, respectively, where the accuracy of pseudo labels was calculated by the ratio of *correct pseudo labels* to *total pseudo labels*.

In the 1-shot setting, the intra-view GCN classifier worked as the MLP classifier because there was no neighbor in each mini-batch. However, as shown in Figures 3a and 3b, the number of PSs and the classification results provided by both intra-view GCN and inter-view GCN classifiers were almost similar, which demonstrated that the proposed method successfully alleviated the imbalance in classification accuracy between these GCN classifiers caused by the bias of the labeled samples for training. Besides, the classification results of the intra-view GCN classifier under the 3-shot setting were slightly higher than that of the inter-view GCN classifier, which revealed that the proposed method effectively exploited a few labeled target samples to generalize on the unlabeled target data. Furthermore, the difference in classification results between the MLP classifier and GCN classifiers was reduced following the increase in training steps that highlighted the effectiveness of the proposed Trico-training strategy, as shown in Figure 3b.

Feature visualization. We used t-SNE [29] to visualize the feature embeddings of 10 classes for the domain adaptation scenario $R \to S$ on *DomainNet* using ResNet-34 with the 3-shot setting. The visualization results of Figures 4a and 4b illustrated representations of the unlabeled target data corresponding to before and after applying adaptation extracted by the baseline model and TriCT, respectively. As shown in these figures, the representations among different classes of unlabeled target data extracted by TriCT were more discriminative compared to the baseline model. Moreover, as shown in Figures 4a and 4b, it was obvious that TriCT was successful in solving the intra-domain discrepancy when the representations of unlabeled target samples belonging to the same class were well clustered, which was indicated by the dashed lines and bidirectional arrows. Figures 4c and 4d displayed domain alignment results before and after applying adaptation extracted by the baseline model and TriCT, respectively. As shown in Figure 4d, the representations of source and target data were well aligned across domains when the distance between source and target domains was significantly reduced, revealing that the proposed method effectively alleviated the inter-domain discrepancy.

5. Conclusion

This paper introduced a novel method called TriCT to overcome both inter-domain and intra-domain discrepancies in SSDA. TriCT utilized two graph convolutional networks as the auxiliary models to exploit the training data structure with three co-training strategies that encouraged the different classifier models to share their knowledge. Therefore, the training model not only provided robustness classification results but converged quickly. Our method showed outstanding classification performance on the target domain compared to previous state-of-the-art SSDA methods on several standard SSDA benchmark datasets.

Acknowledgement. This work was partially supported by 1) National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) under Grant RS-2023-00208763; 2) Institute of Information & communications Technology Planning & Evaluation (IITP) under the Artificial Intelligence Convergence Innovation Human Resources Development (IITP-2023-RS-2023-00254592) grant funded by the Korea government (MSIT).

References

[1] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Proc. ECCV*, 2020. 1

[2] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proc. CVPRW*, 2020. 3

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. CVPR*, page 248–255, 2009. 6

[4] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, Franois Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, page 3723–3732, 2017. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, page 770–778, 2016. 6

[6] Pin Jiang, Aming Wu, Yahong Han, Yunfeng Shao, Meiyu Qi, and Bingshuai Li. Bidirectional adversarial training for semi-supervised domain adaptation. In *Proc. IJCAI*, pages 934–940, 2020. 6

[7] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *Proc. ECCV*, page 464–480, 2018. 5

[8] Taekyung Kim and Changick Kim. Attract, perturb, and explore: Learning a feature alignment network for semi-supervised domain adaptation. In *Proc. ECCV*, pages 591–607, 2020. 2, 6

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NeurIPS*, page 1097–1105, 2012. 6

[10] Jichang Li, Guanbin Li, Yemin Shi, and Yizhou Yu. Cross-domain adaptive clustering for semi-supervised domain adaptation. In *Proc. CVPR*, pages 2505–2514, 2021. 1, 2, 6

[11] Kai Li, Chang Liu, Handong Zhao, Yulun Zhang, and Yun Fu. Ecacl: A holistic framework for semi-supervised domain adaptation. In *Proc. ICCV*, pages 8578–8587, 2021. 1, 2, 6

[12] Yadan Luo, Zijian Wang, Zi Huang, and Mahsa Baktashmotlagh. Progressive graph learning for open-set domain adaptation. In *Proc. ICML*, page 6468–6478, 2020. 3, 6

[13] Ning Ma, Jiajun Bu, Lixian Lu, Jun Wen, Sheng Zhou, Zhen Zhang, Jingjun Gu, Haifeng Li, and Xifeng Yan. Context-guided entropy minimization for semi-supervised domain adaptation. *Neural Networks*, pages 270–282, 2022. 6

[14] Samarth Mishra, Kate Saenko, and Venkatesh Saligrama. Surprisingly simple semi-supervised domain adaptation with pretraining and consistency. In *Proc. BMVC*, page 177, 2021. 2, 6

[15] Ba Hung Ngo, Ju Hyun Kim, Yeon Jeong Chae, and Sung In Cho. Multi-view collaborative learning for semi-supervised domain adaptation. *IEEE Access*, volume 9:166488–166501, 2021. 1, 2, 6

[16] Ba Hung Ngo, Jae Hyeon Park, So Jeong Park, and Sung In Cho. Semi-supervised domain adaptation using explicit class-wise matching for domain-invariant and class discriminative feature learning. *IEEE Access*, volume 9:128467–128480, 2021. 2, 6

[17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6

[18] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proc. ICCV*, 2017. 6

[19] Manuel Pérez-Carrasco, Pavlos Protopapas, and Guillermo Cabrera-Vives. Con^{2}DA: Simplifying semi-supervised domain adaptation by learning consistent and contrastive feature representations. In *Proc. NeurIPS*, 2021. 1, 2, 6

[20] Can Qin, Lichen Wang, Qianqian Ma, Yu Yin, Huan Wang, and Yun Fu. Contradictory structure learning for semi-supervised domain adaptation. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 576–584, 2021. 2, 6

[21] Can Qin, Lichen Wang, Qianqian Ma, Yu Yin, Huan Wang, and Yun Fu. Semi-supervised domain adaptive structure learning. *arXiv preprint arXiv:2112.06161*, 2021. 2

[22] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Deep hashing network for unsupervised domain adaptation. In *Proc. CVPR*, 2017. 5

[23] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proc. ICCV*, pages 8050–8058, 2019. 2, 6

[24] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proc. CVPR*, page 3723–3732, 2018. 1

[25] Ankit Singh. Clda: Contrastive learning for semi-supervised domain adaptation. In *Proc. NeurIPS*, pages 5089–5101, 2021. 1, 2, 6, 7

[26] Anurag Singh, Naren Doraiswamy, Sawa Takamuku, Megh Bhalerao, Titir Dutta, Soma Biswas, Aditya Chepuri, Balasubramanian Vengatesan, and Naotake Natori. Improving semi-supervised domain adaptation using effective target selection and semantics. In *Proc. CVPR*, pages 2709–2718, 2021. 6

[27] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proc. CVPR*, page 2962–2971, 2017. 1

[28] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proc. CVPR*, 2017. 1

[29] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, volume 9:770–778, 2008. 8

[30] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Adapting visual category models to new domains. In *Proc. CVPR*, 2017. 5

[31] Zizheng Yan, Yushuang Wu, Guanbin Li, Yipeng Qin, Xiaoguang Han, and Shuguang Cui. Multi-level consistency learning for semi-supervised domain adaptation. In *Proc. IJCAI*, 2022. 2, 6

[32] Luyu Yang, Yan Wang, Mingfei Gao, Abhinav Shrivastava, Kilian Q. Weinberger, Wei-Lun Chao, and Ser-Nam Lim. Deep co-training with task decomposition for semi-supervised domain adaptation. In *Proc. ICCV*, pages 8906–8916, 2021. 2, 6

[33] Jeongbeen Yoon, Dahyun Kang, and Minsu Cho. Semi-supervised domain adaptation via sample-to-sample self-distillation. In *Proc. WACV*, pages 1978–1987, 2022. 6

[34] Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. In *Proc. NeurIPS*, volume 31, page 8559–8570, 2018. 1