Correntropy-Induced Wasserstein GCN: Learning Graph Embedding via Domain Adaptation

Wei Wang, Gaowei Zhang, Hongyong Han, and Chi Zhang

***Abstract*— Graph embedding aims at learning vertex representations in a low-dimensional space by distilling information from a complex-structured graph. Recent efforts in graph embedding have been devoted to generalizing the representations from the trained graph in a source domain to the new graph in a different target domain based on information transfer. However, when the graphs are contaminated by unpredictable and complex noise in practice, this transfer problem is quite challenging because of the need to extract helpful knowledge from the source graph and to reliably transfer knowledge to the target graph. This paper puts forward a two-step correntropy-induced Wasserstein GCN (graph convolutional network, or CW-GCN for short) architecture to facilitate the robustness in cross-graph embedding. In the first step, CW-GCN originally investigates correntropy-induced loss in GCN, which places bounded and smooth losses on the noisy nodes with incorrect edges or attributes. Consequently, helpful information are extracted only from clean nodes in the source graph. In the second step, a novel Wasserstein distance is introduced to measure the difference in marginal distributions between graphs, avoiding the negative influence of noise. Afterwards, CW-GCN maps the target graph to the same embedding space as the source graph by minimizing the Wasserstein distance, and thus the knowledge preserved in the first step is expected to be reliably transferred to assist the target graph analysis tasks. Extensive experiments demonstrate the significant superiority of CW-GCN over state-of-the-art methods in different noisy environments.**

***Index Terms*— Graph embedding, graph convolutional network (GCN), cross-graph embedding, domain adaptation, correntropy.**

I. Introduction

GRAPH embedding refers to learning a unique low-dimensional, compact, and continuous vector representation for each graph node [1], [2]. These representations play a substantial role to the success of various tasks, including action recognition [3], object classification [4] and dimensionality reduction [5]. Recently, graph embedding has made great progress by simultaneously capturing topological structures, node attributes and node labels from nonlinear structured graphs, where different sources of information can compensate with each other [6], [7], [8], [9], [10], [11], [12]. To further generalize the representation from trained graphs to new (sub)graphs, the graph embedding methods can be easily expanded by defining the embedding as a parametric function of the feature vectors. This expansion brings satisfactory generalization under the assumption that the new graph follows the same distribution as the trained graph, meaning that the two graphs are from the same domain. With these representations, if we learn some classic machines (e.g., node classifiers or visualization models) using the labels from the trained graph, these machines can be directly applied to the new graph.

However, in some practical applications, it is expensive or even impossible to obtain node labels for graphs in a target domain. In this case, some mature graph from a related but different domain (i.e., source domain) can be used as prior knowledge, which contains plenty of label information. For instance, given a newly formed social graph (target), no users have labels reflecting their interests and it is difficult to classify the users into different groups based on their interests. We can utilize the abundant class information in a mature graph (source). As for a newly collected protein-protein interaction graph (target) and a well-established protein graph (source), we could classify the proteins of the target graph into different function categories for disease diagnosis by leveraging knowledge from the labeled source graph. The difference in attribute distributions and structures between target and source graphs, limits the generalization of the above representation methods. For instance, the source graph is an English language graph and the target graph is a French language graph, where each node represents a document and the link weight is the cosine similarity. Apparently, the two graphs have greatly different text features and the same link weights in different graphs indicate different relevances. Even so, the two graphs still share a lot of the same semantic information which is difficult to be uncovered.

Many efforts in graph embedding have been made to realize the knowledge transfer between two graphs from different domains [13], [14], [15], [16], [17]. Inspired from domain adaptation in machine learning, some recent works have shown their superiority by estimating and minimizing the distribution discrepancy of two graphs [18], [19], [20], [21], [22], [23], [24], [25], [26], [27]. Specifically, these works mainly contain two components: 1) extracting information from source graph

Manuscript received 21 March 2022; revised 8 March 2023 and 10 May 2023; accepted 24 June 2023. Date of publication 12 July 2023; date of current version 18 July 2023. This work was supported in part by the Major Project for New Generation of AI under Grant 2018AAA0100400, in part by the Natural Science Foundation of China under Grant 62076232 and Grant 62072457, and in part by the Xiaomi Young Talents Program. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Kui Jia. *(Corresponding author: Chi Zhang.)*

Wei Wang, Gaowei Zhang, and Hongyong Han are with the Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: weiwang@bupt.edu.cn; zhanggaowei@bupt.edu.cn; hanhongyong@bupt.edu.cn).

Chi Zhang is with the Institute of Automation, the Centre for Artificial Intelligence and Robotics (HKISI_CAS), and the State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Chinese Academy of Sciences (CASIA), Beijing 100190, China (e-mail: chi.zhang@nlpr.ia.ac.cn).

Digital Object Identifier 10.1109/TIP.2023.3293774

(i.e., topological structures, attributes and labels) using a graph embedding model, e.g., graph convolutional network (GCN) [6] or the autoencoder with positive pointwise mutual information (PPMI) matrix [28]; 2) transferring information to target graph by deriving a common latent space where the representation distributions of two graphs are minimized based on a measurement, e.g., maximum mean discrepancy (MMD) [29] or Wasserstein distance [30]. However, as the key role in knowledge transfer, the source graph is always contaminated by unpredictable and severe noise in practice (e.g., attribute and edge pollution exist simultaneously), especially when the graph comes from the web. Coping with various kinds of complex noise is not an easy problem mainly due to the following two reasons. (1) **Difficult knowledge extraction**. Most existing graph embedding models rely on the assumption that the graph edges and node attributes reflect the likelihood of label agreement. In other words, the nodes in the neighborhood correspond to similar attributes and the same label. When the source graph contains noisy edges and attributes, this noisy information may induce non-correspondence and make it difficult to extract helpful knowledge from the source graph. (2) **Difficult knowledge transfer**. The previous discrepancy measurements will be easily impacted by noise, leading to unreliable graph alignment. For instance, MMD is a nonparametric criterion in a reproducing kernel Hilbert space (RKHS), where the empirical estimate is sensitive to noise. Consequently, minimizing the gap between graphs without the influence of noise remains another central problem.

In this paper, the two aforementioned challenges are tackled by a two-step correntropy-induced Wasserstein GCN (CW-GCN) architecture. In step 1, the low-dimensional representations of the source graph are learned by: 1) detecting and suppressing the severely contaminated nodes which have noisy attributes or links, bypassing any specific assumptions on noise (e.g., sparse); 2) distilling three kinds of helpful information (structures, attributes and labels) only from the clean nodes. To this end, CW-GCN originally investigates correntropy-induced loss [31], [32] in GCN, which is derived from information theory and proven to have the theoretical foundation of handling unpredictable noise and outliers. By introducing auxiliary variables, this step is proven to be optimized under Half-Quadratic (HQ) [33] analysis. The auxiliary variables further pave the way for the reliable knowledge transfer in step 2. Specifically, integrating the auxiliary variables with the assumption that the node representations in the same class follow a Gaussian distribution, step 2 explores a robust and efficient Wasserstein distance to measure the difference in marginal distributions between graphs without the influence of noise. With the aim of minimizing the Wasserstein distance, the target graph is mapped to the same embedding space as the source graph. Consequently, in the shared embedding space, the rich knowledge preserved from the source graph in the first step (especially the label-discriminative information) is expected to be reliably transferred to assist the target graph analysis tasks. It is emphasized that in our two-step paradigm, the target mapping has unshared weights with the source mapping and the source mapping is fixed in step 2. It is a flexible and powerful paradigm to: 1) extract representations more specific to target graph; 2) reduce the impact of the contaminated source nodes on target graph mapping; 3) avoid overly complicated models. The main contributions of our work are summarized as follows.

- Based on the theoretic guarantee of correntropy, we investigate correntropy-induced loss in graph embedding. The new loss can be generally applied in graph embedding to tackle the noisy cases where the assumption (i.e., the graph edges and node attributes reflect the likelihood of label agreement) does not hold.
- A novel Wasserstein distance is introduced to address the scenario in which source and target graph representations have different distributions. Compared with the previous measurements, our distance facilitates both robustness and efficiency in reducing the cross-graph bias.
- The above two components are implemented in a two-step paradigm, which allows independent and asymmetric source and target mappings. This powerful paradigm helps the proposed CW-GCN achieve robust knowledge extractor and knowledge transfer successively.
- Experiments on real-world graphs demonstrate the promise of CW-GCN by considerable improvements under different noisy tasks, compared with the state-of-the-art graph embedding and domain adaptation methods.

The rest of this paper is organized as follows. We briefly review related works in Section II. Section III develops the problem definition and proposed method CW-GCN. Then, in Section IV, we report the experimental results. Finally, the concluding remarks are given in Section V.

II. Related Works

A. Graph Embedding

Graph embedding or network embedding learns vector representations to reveal the semantics of the original graphs. Existing graph embedding methods can be further divided into two categories. The methods in the first category focus on preserving topological proximities between nodes, e.g., local neighborhood structures and global community structures [34], [35], [36], [37]. To further analyse attributed graphs and get better representations, the methods in the second category aim at jointly embedding the graph structure, vertex attributes and labels, based on the assumption that the two nodes connected by an edge likely have the similar attributes and the same label [6], [7], [8], [9], [10], [11], [12], [38]. In this line, GCN [6] is among the most successful paradigms, which extends existing convolutional neural networks for processing graphs. Specifically GCN directly embeds the graph structure and the node attribute with a spectral convolutional function for each layer, and minimizes the cross-entropy error over all labeled nodes. GCN has successful applications in broad areas [3], [4] and various models based on GCN have been proposed [11], [38], [39], [40]. For instance, GAM [38] applies an agreement model on the top of GCN to propagate labels in semi-supervised setting. The attributed graph embedding methods show great generalization capability by assuming that training and test graphs follow the same distribution. Based on the learnt representations, off-the-shelf supervised learning

machines such as node classifiers or visualization models can be directly applied to the new graph using the labels from the trained graph.

The problem of realizing knowledge between two different graphs from different distributions has received increasing attention [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27]. For instance, some methods assume that some links across two graphs or some common nodes are available and utilize these pieces of prior information for graph alignment [15], [16], [17]. Another line of works relies on shallow structures [13], [14] to preform linear transformation for linear structural data. Recently, cross-graph embedding methods based on domain adaptation and end-to-end network architectures [18], [19], [20], [21], [22], [23], [24], [25], [26], [27] are suggested to have a potential to outperform the traditional methods. For instance, CDNE [18] constructs two autoencoders for source and target graphs respectively, where the input is PPMI [28] matrix to capture the graph structure. Then the MMD distances between two graph are minimized for graph-invariant representations. ACDNE [19] constructs PPMI based autoencoders as well and employs the adversarial-based distance measurement as in DANN [41]. ASN [22] adopts a dual GCN model to combine local and global consistency in network topology capturing and also uses the adversarial-based distance measurement to reduce the distribution discrepancy across domains. AdaGIn [25] employs the spatial GNN layers to compute node representations for the source and target graphs. Afterwards, conditional adversarial networks [42] are employed to reduce the domain discrepancy. SR-GNN [21] focuses on the use of CMD [43] and MMD as distance metrics to measure distributions discrepancy for efficiency. Different from the aforementioned methods, we develop the correntropy loss based GCN to handle the complex uncertainty cased by unpredictable noise, resulting in clean information preservation from the noisy source graph. Furthermore, the Wasserstein distance is extended to efficiently and robustly adapt marginal distributions between graphs.

Most recently, Graph Augmentation Learning (GAL) [44], [45], [46], [47] has provided promising solutions in addressing restrictions in graph learning, e.g., low-quality node attributes or low-quality graph structures. Specifically, LA-GNN [46] proposes a local augmentation strategy to learn representations of nodes with few neighbors. PTDNet [44] enhances the robustness performance of GNNs by removing task-irrelevant edges. DropEdge [45] randomly removes graph edges in message passing mechanism to alleviate over-smoothing. Pro-GNN [47] jointly learns a structural graph and a robust graph neural network model to defend adversarial attacks on graphs, i.e., adding or deleting or rewiring edges. Our work differs from the GAL methods in two aspects. First, we introduce the empirical correntropy-induced loss into graph embedding to effectively handle the challenges when the graph data may be contaminated by complex noise in real-world applications (e.g., noisy edges and corrupted node attributes exist simultaneously). Second, our work focuses on reliably transferring knowledge from the source graph to the target graph. Therefore, a robust Wasserstein distance is explored based on the C-loss to measure the difference in marginal distributions between graphs without the influence of noise.

B. Domain Adaptation

Feature extraction based domain adaptation in machine learning aims at deriving a common latent space by minimizing the distributions discrepancy of two domains with different measurements (strategies) [48]. Recently, building a deep structure for nonlinear spaces is regarded as a powerful way to bridge the distribution gap [41], [42], [43], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58]. In this deep family, two categories are mainly explored: matching the marginal distributions of two domains (e.g. [41], [43], [49], [50]); implicitly or explicitly aligning the class-conditional distributions based on the target pseudo-labels provided by the source classifier (e.g. [42], [56], [58]). The p-th Wasserstein distance [59] is a kind of distance measure between two probabilities $\mathbb{P}$ and $\mathbb{Q}$, defined as

$$W_p(\mathbb{P},\mathbb{Q}) = \Big(\inf_{\mu\in\Gamma(\mathbb{P},\mathbb{Q})}\int \rho(x,y)^p d\mu(x,y)\Big)^{1/p}, \tag{1}$$

where $\mathbb{P},\mathbb{Q} \in \{\mathbb{P} : \int \rho(x,y)^p d\mathbb{P}(x) < \infty, \forall y \in M\}$ are two probability measures on the set M with order p and $\Gamma(\mathbb{P},\mathbb{Q})$ is the set of all measures on $M \times M$ with marginals $\mathbb{P}$ and $\mathbb{Q}$. The 1-st Wasserstein distance [30], [60] has been successfully applied in domain adaptation methods [50], [59], showing gradient superiority and theoretic advantages (e.g., generalization guarantee) compared with other measurements. These methods utilize the 1-st Wasserstein distance to adapt the marginal distributions between domains, where the distance can be estimated through constructing and training a multi-layered network in a compact space. In each step towards minimizing the distance, the multi-layered network will firstly be iteratively trained via weight clipping, which will greatly increase the overall complexity and may cause gradient vanishing or exploding problems [61]. In addition, the applied distance will be easily impacted by real-world noise, leading to degraded (or even negative) information transfer.

C. Correntropy

In information-theoretic learning, correntropy [31] is a similarity measure between random variables X and Y:

$$V(X,Y) = E[k(X-Y)], \tag{2}$$

where $k(.)$ is a translation-invariant Mercer kernel function and $E[\cdot]$ denotes the expectation operator. Correntropy is essentially the correlation in RKHS and has a close relationship with M-estimation [62]. Given a finite number of samples $\{(x_i,y_i)\}_{i=1}^n$, the empirical correntropy-induced loss function or the C-loss function is

$$l_c(X,Y) = \sum_{i=1}^{n}(k(0) - k(x_i - y_i)). \tag{3}$$

When the Gaussian kernel is considered, C-loss is proven to have the nice property [63], [64]: 1) it is Bayes consistent; 2) it embeds the higher order statistics; 3) it behaves like

L_2-norm for a small error vector and L_0-norm for a large error vector. As a bounded, smooth, and nonconvex loss, C-loss has been successfully applied to many applications (e.g., face recognition, signal processing) and is proven to be applicable under a variety of unpredicted noisy environments (e.g., missed entries, dense corruptions or heavy-tailed noise) [65], [66], [67]. In this paper, the proposed CW-GCN is designed for graphs with node-to-node interactions and dependencies, which investigates C-loss in GCN and makes it possible to boost graph embedding performance on noisy graphs. Furthermore, CW-GCN is proven to be optimized in HQ way, where the robustness is explicitly explained.

III. Proposed Method

A. Problem Definition

Let $G_s = (\mathcal{V}_s, \mathcal{E}_s, \mathbf{X}_s, \mathbf{Y}_s)$ be a source attributed graph. Specifically, the set $\mathcal{V}_s = \{v_s^1, \ldots, v_s^n\}$ represents n vertexes, the matrix $\mathbf{X}_s = [\mathbf{x}_s^1, \ldots, \mathbf{x}_s^n] \in \mathbb{R}^{d_s \times n}$ collects the d_s-dimensional attribute vector, the set $\mathcal{E}_s = \{e_s^{i,j}\}_{i,j=1}^n$ represents the edges, the matrix $\mathbf{Y}_s = [\mathbf{y}_s^1, \ldots, \mathbf{y}_s^n] \in \mathbb{R}^{C \times n}$ collects the labels with C classes, $\mathbf{y}_s^{ic} = 1$ if $\mathbf{x}_s^i$ is associated with label c; otherwise, $\mathbf{y}_s^{ic} = 0$. The topological structure of G_s can be represented by an adjacency matrix $\mathbf{A}_s \in \mathbb{R}^{n \times n}$, where $(\mathbf{A}_s)_{i,j} = 0$ for unlinked nodes, otherwise $(\mathbf{A}_s)_{i,j} = 1$ for unweighted graph and $(\mathbf{A}_s)_{i,j} > 0$ for weighted graph. We consider a target attributed graph $G_t = (\mathcal{V}_t, \mathbf{X}_t)$ where $\mathcal{V}_t = \{v_t^1, \ldots, v_t^m\}$ represents the set of unlabeled m vertexes and the matrix $\mathbf{X}_t = [\mathbf{x}_t^1, \ldots, \mathbf{x}_t^m] \in \mathbb{R}^{d_t \times m}$ collects the d_t-dimensional attribute vector. d_t is assumed to be equal to d_s for simplicity, but it can be easily generalized to the setting $d_t \neq d_s$. The topological structure of G_t can be represented by an adjacency matrix $\mathbf{A}_t \in \mathbb{R}^{m \times m}$ as well. These two relative graphs have the properties: 1) the marginal probability distributions of $\mathbf{X}_t$ and $\mathbf{X}_s$ are not equal, $P_t(\mathbf{X}_t) \neq P_s(\mathbf{X}_s)$; 2) $\mathcal{E}_s$ and $\mathcal{E}_t$ indicate different relevances associated with the attributes; 3) they share the same label space. The task of our method is to uncover the shared space between G_s and G_t by preserving knowledge from G_s and minimizing the graph bias. Consequently, in this shared space, if we train a graph analysis model using the labels from G_s, the model is expected to have good performance on G_t. Fig. 1 shows the workflow of our new architecture consisting of the following two steps.

1) The first step aims at mapping the source nodes to low-dimensional vectors, where two fundamental questions need to be addressed: how to preserve the topological structure $\mathbf{A}_s$, the content information $\mathbf{X}_s$ as well as the label information $\mathbf{Y}_s$? How to detect the contaminated nodes in G_s with noisy attributes or links, and eliminate their negative impacts on mapping process?

2) The second step maps the target graph to the same embedding space as the source graph by solving the question: how to efficiently and robustly match the marginal probability distributions of source and target graph representations?

B. The First Step: Representation Learning for G_s

1) Feature Extractors: To encode the graph structure $\mathbf{A}_s$ and content information $\mathbf{X}_s$ directly, we adopt GCN [6] in this paper. Note that our graph learning framework is general to various graph embedding models. Specifically, the following spectral convolutional function $f(\mathbf{Z}_s^{k+1}, \mathbf{A}_s)$ is defined to build the transformation for each layer:

$$f(\mathbf{Z}_s^{k+1}, \mathbf{A}_s) = ReLU(\widetilde{\mathbf{D}}_s^{-\frac{1}{2}} \widetilde{\mathbf{A}}_s \widetilde{\mathbf{D}}_s^{-\frac{1}{2}} \mathbf{Z}_s^k \mathbf{W}_s^k), \tag{4}$$

where $\widetilde{\mathbf{A}}_s = \mathbf{A}_s + \mathbf{I}$ is the adjacency matrix with added self-connections, $(\widetilde{\mathbf{D}}_s)_{ii} = \sum_j (\widetilde{\mathbf{A}}_s)_{ij}$. $\mathbf{Z}_s^k$ is the matrix of activations in the k^{th} layer and $\mathbf{W}_s^k$ is the parameter matrix to be learned.

Based on the well defined spectral convolution function, the arbitrary deep convolutional neural networks can be constructed. In this work, we adopt two layers to map the node features to the node embedding space,

$$\mathbf{Z}_s^1 = f(\mathbf{X}_s, \mathbf{A}_s), \quad \mathbf{Z}_s^2 = f(\mathbf{Z}_s^1, \mathbf{A}_s). \tag{5}$$

2) The Overall Objective Function: To make $\mathbf{Z}_s^1$ and $\mathbf{Z}_s^2$ label-discriminative, a fully connected layer is added as the classifier,

$$\widehat{\mathbf{Y}}_s = \sigma(\mathbf{Z}_s^2 \mathbf{W}_s^y), \tag{6}$$

where σ is a softmax function for multi-class nodes or a sigmoid function for multi-label nodes. $\mathbf{W}_s^y$ is the trainable parameter matrix in the fully connected layer. The output $\widehat{\mathbf{Y}}_s = [\hat{\mathbf{y}}_s^1, \ldots, \hat{\mathbf{y}}_s^n]$, $\hat{\mathbf{y}}_s^i \in R^{c \times 1}$ predicts a probability distribution for v_s^i over a set of c classes. Finally, the whole network is trained by minimizing the loss function $l(\widehat{\mathbf{Y}}_s, \mathbf{Y}_s)$ over all labeled source nodes. In the basic case without noise contamination, the cross-entropy function or the L_2-norm function is commonly employed as $l(\cdot)$. Note that the spectral convolutional function indicates that the representation of each node in $\mathbf{Z}_s^1$ or $\mathbf{Z}_s^2$ depends only on the attributes itself and the attributes of the neighbors. In this term, minimizing the cross-entropy function or the L_2-norm function relies on the basic assumption that the nodes in the neighborhood correspond to

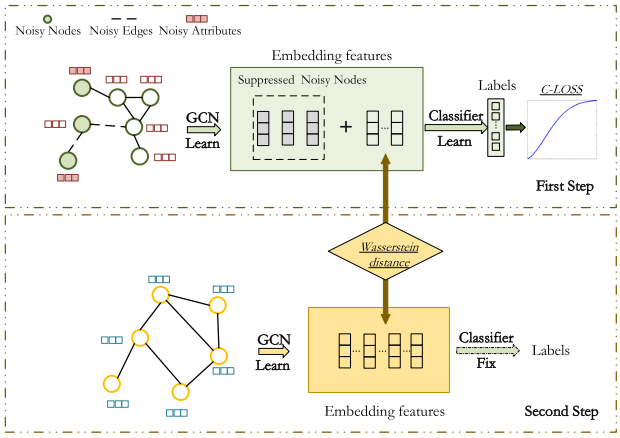
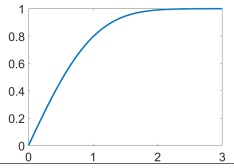
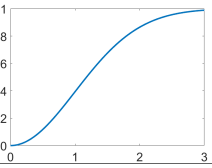
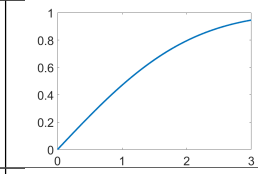
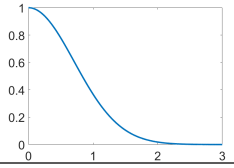
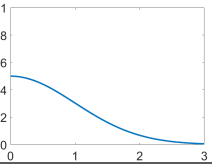
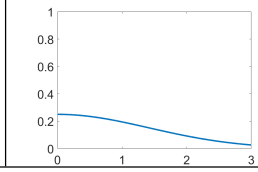


Fig. 1. Framework of our proposed method. We first learn the source GCN and classifier using labeled source graph. In the process, the noisy nodes (contain noisy links or corrupted attributes, or both) are suppressed and their negative impacts are eliminated. Next, we learn the target GCN based on extending the Wasserstein distance to avoid noisy source nodes as well. At the testing time, the target nodes in the shared space are classified by the source classifier learned in the first step.

TABLE I
GRAPHIC REPRESENTATIONS OF SOME $\phi(e)$ AND THE CORRESPONDING b

Function	$\mu^2 = 1$	$\mu^2 = 2$	$\mu^2 = 4$
$\phi(e)$			
b			

the same label. That is, the quality of the learnt representations relies on the quality of G_s.

In the noisy conditions, the source nodes contain two parts: the clean nodes and the contaminated nodes (i.e., the outliers). When we assume nothing regarding the noise, it is hard to separate these two parts. The outliers in G_s essentially contain noisy links or corrupted attributes, or both. The distinctive attributes between the outliers and their neighbors may result in different labels, thus leading to large prediction error. However, the traditional cross-entropy or L_2-norm functions are sensitive to noise, which put great emphasize on minimizing the prediction error of these contaminated nodes. Consequently, based on the learnt $\mathbf{Z}_s^1$ and $\mathbf{Z}_s^2$, the low-dimensional representations of the clean nodes loss the ability of preserving reliable information. To optimally eliminate the negative influence of the contaminated nodes, it is necessary to explore a robust measurement for the prediction error. We originally introduce the empirical C-loss into graph embedding. Defining $e_i = \|\hat{\mathbf{y}}_s^i - \mathbf{y}_s^i\|_2$ and $\phi(e_i) = (1 - exp(-e_i^2/\mu^2))$, the C-loss in Eq. (3) with Gaussian kernel can be written as: $l_c(\widehat{\mathbf{Y}}_s, \mathbf{Y}_s) = \sum_{i=1}^n \phi(e_i)$. In this term, the following expression with an L_2-norm regularization can be obtained:

$$\begin{aligned}&\min_{\mathbf{W}_s^1, \mathbf{W}_s^2, \mathbf{W}_s^y} \sum_{i=1}^{n} \phi(e_i) + \alpha(\|\mathbf{W}_s^1\|_F^2 + \|\mathbf{W}_s^2\|_F^2) \\ &= \sum_{i=1}^{n} 1 - exp(-\|\hat{\mathbf{y}}_s^i - \mathbf{y}_s^i\|_2^2/\mu^2) + \alpha(\|\mathbf{W}_s^1\|_F^2 + \|\mathbf{W}_s^2\|_F^2),\end{aligned} \tag{7}$$

where μ is the Gaussian kernel width and α is the trade-off parameter. The column-wise minimization $\|\hat{\mathbf{y}}_s^i - \mathbf{y}_s^i\|_2^2$ penalizes the error corresponding to a single node as a whole. It is derived from $L_{2,1}$-norm (group sparsity) and is used to control node-specific error. In Eq. (7), the outliers with large prediction error will obtain stable C-loss values and only make limited impacts on the minimization. To give a clear illustration of the robustness, we graphically depict some functions $\phi(e)$ in Table I. With the increasing of e, the curves tend to be flattened out. In this term, $\phi(e)$ imposes bounded penalties to the large outliers, where the kernel width μ controls the increasing slope of $\phi(e)$. The explanation of b in Table I will be given in the following section. With the robust measurement for the prediction error, the learnt $\mathbf{Z}_s^1$ and $\mathbf{Z}_s^2$ successfully capture the structure proximity, attribute proximity and label-discriminative information.

3) Optimization and Robustness Analysis: We explicitly analyse the robustness of Eq. (7) and propose an iteratively learning procedure in this section. Based on the convex conjugate function theory [68], we have Lemma 1 [69].

Lemma 1: Let us consider the function ϕ that satisfies the hypotheses:

ϕ *is even*

ϕ *is continuous near zero and* $\mathcal{C}^1$ *on* $\mathbb{R}^* = \mathbb{R}\backslash\{0\}$

$\phi(\sqrt{\cdot})$ *is concave on* $\mathbb{R}_+$

$$\lim_{x\to\infty} \frac{\phi(x)}{x^2} = 0,$$

and let $b_\infty = \lim_{x\to 0} \frac{\phi'(x)}{2x}$. *Then*

1) The function ϕ can be expressed as

$$\phi(x) = \inf_{b\in(0,b_\infty]} (bx^2 + \psi(b)), \tag{8}$$

where $\psi(b) = \sup_{x\in\mathbb{R}}(\phi(x) - bx^2)$.

2) $\psi(b)$ is convex.

3) The infimum of (8) is uniquely reached at

$$b_x = \begin{cases} b_\infty, & if \quad x = 0 \\ \dfrac{\phi'(x)}{2x}, & otherwise \end{cases} \tag{9}$$

and the expression of $\psi(b)$ is not required to compute b_x.

As a consequence of Lemma 1, Proposition 1 can be derived which enables Eq. (7) to be minimized in HQ way.

Proposition 1: As for $\phi(e) = (1 - exp(-e^2/\mu^2))$, there exists a convex function $\psi(b)$ such that

$$\phi(e) = \inf_{b\in(0,b_\infty]} (be^2 + \psi(b)), \tag{10}$$

where $\psi(b) = \sup_{e\in\mathbb{R}}(\phi(e) - be^2)$. *For a fixed e, the infimum of Eq. (10) is uniquely reached at* $b = exp(-e^2/\mu^2)/\mu^2$.

Proof: The continuity, concavity and limit properties of $\phi(e)$ listed in Lemma 1 can be easily obtained. Then, based

on Eq. (9), the infimum of Eq. (10) is uniquely reached at $b = \phi'(e)/2e = exp(-e^2/\mu^2)/\mu^2$. □

From Proposition 1, Eq. (7) is translated into an augmented objective function,

$$\begin{aligned}\min_{\mathbf{W}_s^1,\mathbf{W}_s^2,\mathbf{W}_s^y} &\sum_{i=1}^{n}\phi(e_i)+\alpha(\|\mathbf{W}_s^1\|_F^2+\|\mathbf{W}_s^2\|_F^2)\\ &=\min_{\mathbf{W}_s^1,\mathbf{W}_s^2,\mathbf{W}_s^y,B}\sum_i\{b_i\|\hat{\mathbf{y}}_s^i-\mathbf{y}_s^i\|_2^2+\psi(b_i)\}\\ &\quad+\alpha(\|\mathbf{W}_s^1\|_F^2+\|\mathbf{W}_s^2\|_F^2),\end{aligned}\tag{11}$$

where $B = \{b_1, \ldots, b_n\}$ are called auxiliary variables. Inheriting from HQ analysis [69], Eq. (11) can be minimized in an alternative strategy. The detailed procedure is in Algorithm 1, containing two components: 1) updating $\mathbf{W}_s^1$, $\mathbf{W}_s^2$ and $\mathbf{W}_s^y$ using gradient descent, where $\psi(b_i)$ is a constant; 2) directly obtaining the closed-form of b_i following Proposition 1.

Algorithm 1 Step 1: Representation Learning for G_s

Require:
$G_s = (\mathcal{V}_s, \mathcal{E}_s, \mathbf{X}_s, \mathbf{Y}_s)$, μ, α, d_1, d_2, T.
Ensure: Representations $\mathbf{Z}_s^1 \in \mathbb{R}^{d_1\times n}$, $\mathbf{Z}_s^2 \in \mathbb{R}^{d_2\times n}$, variables B.
1: Randomly initialize the parameters $\Theta = \{\mathbf{W}_s^1, \mathbf{W}_s^2, \mathbf{W}_s^y\}$.
2: Initialize $B = \{b_1, ..., b_n\}$ with all ones $\{1, ..., 1\}$.
3: for iterator = 1,2,3,..., T do
4: Fix B
5: Backpropagate the loss in Eq. (11) and update Θ;
6: Fix Θ
7: Based on Proposition 1, $\{b_i\}_{i=1}^n$ can be easily updated as

$$b_i = exp(-\|\hat{\mathbf{y}}_s^i - \mathbf{y}_s^i\|_2^2)\|_2^2/\mu^2)/\mu^2; \tag{12}$$

8: end for
9: Return $\mathbf{Z}_s^1$, $\mathbf{Z}_s^2$ and B.

Remark 1: The variable b in Algorithm 1 explicitly explains the robustness. The polluted node, which may have dissimilar attributes and different labels with its neighbours, will yield large prediction error base on GCN. In this term, the polluted node will get small value b following Proposition 1. Table I depicts some specific functions $\phi(e)$ and their corresponding auxiliary variables b. As shown in the table, this auxiliary variable greatly decreases when the error increases. Consequently, b, which acts as the weight in the loss function in Eq. (11), dismisses the negative influence of the polluted node on updating Θ. Inheriting from the superiority of correntropy in robust learning, b can effectively deal with noisy graphs in the challenging conditions with different kinds of complex noise. We emphasize that b also plays an important role in knowledge transfer in the next section.

C. The Second Step: Representation Learning for G_t

1) Feature Extractors: We adopt the two-layered GCN as well which maps G_t to the same space as G_s,

$$\begin{aligned}\mathbf{Z}_t^1 &= f(\mathbf{X}_t, \mathbf{A}_t) = ReLU(\widetilde{\mathbf{D}}_t^{-\frac{1}{2}}\widetilde{\mathbf{A}}_t\widetilde{\mathbf{D}}_t^{-\frac{1}{2}}\mathbf{X}_t\mathbf{W}_t^1),\\ \mathbf{Z}_t^2 &= f(\mathbf{Z}_t^1, \mathbf{A}_t) = ReLU(\widetilde{\mathbf{D}}_t^{-\frac{1}{2}}\widetilde{\mathbf{A}}_t\widetilde{\mathbf{D}}_t^{-\frac{1}{2}}\mathbf{Z}_t^1\mathbf{W}_t^2),\end{aligned}\tag{13}$$

where $\widetilde{\mathbf{A}}_t = \mathbf{A}_t + \mathbf{I}$, $(\widetilde{\mathbf{D}}_t)_{ii} = \sum_j(\widetilde{\mathbf{A}}_t)_{ij}$, $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$ are the parameter matrixes to be learned.

2) The Wasserstein Distance: Given the node representations $\mathbf{z}_t^l \in \mathbf{Z}_t^l$ and $\mathbf{z}_s^l \in \mathbf{Z}_t^l$ ($l = 1, 2$), we further assume $P_s(\mathbf{z}_s^l)$ and $P_t(\mathbf{z}_t^l)$ to be Gaussian distributions:

$$P_s(\mathbf{z}_s^l) = \mathcal{N}(\mathbf{m}_s^l, \mathbf{C}_s^l),\ P_t(\mathbf{z}_t^l) = \mathcal{N}(\mathbf{m}_t^l, \mathbf{C}_t^l),$$

where $\mathbf{m}_{s,t}^l$ is the mean of the l-th layer and $\mathbf{C}_{s,t}^l$ is the covariance. We impose the 2-nd Wasserstein distance $W_2(\mathbb{P}, \mathbb{Q})$ as defined in Eq. (1) to measure the distribution gap. Based on the simplified form of the 2-nd Wasserstein distance under Gaussian assumption [70], [71], the following novel 2-nd Wasserstein distance under noisy cases is introduced:

$$\begin{aligned}&W_2(P_s(\mathbf{z}_s^l),\ P_t(\mathbf{z}_t^l))^2\\ &\quad=\frac{1}{2}[\|\mathbf{m}_s^l-\mathbf{m}_t^l\|_2^2+(\mathbf{C}_s^l+\mathbf{C}_t^l-2\sqrt{\mathbf{C}_s^l\mathbf{C}_t^l})]\\ &\quad=\frac{1}{2}[\|\mathbf{m}_s^l-\mathbf{m}_t^l\|_2^2+\|\sigma_s^l-\sigma_t^l\|_2^2],\end{aligned}\tag{14}$$

where

$$\mathbf{m}_s^l=\sum_i b_i\mathbf{z}_s^{il}/\sum_i b_i,\quad \mathbf{m}_t^l=\sum_i \mathbf{z}_t^{il}/m,$$

$$\sigma_s^l=\sqrt{\frac{\sum_i b_i(\mathbf{z}_s^{il}-\mathbf{m}_s^l)^2}{\sum_i b_i-(\sum_i b_i^2/\sum_i b_i)}},\quad \sigma_t^l=\sqrt{\frac{\sum_i(\mathbf{z}_t^{il}-\mathbf{m}_t^l)^2}{m-1}}.$$

Actually, $\mathbf{m}_s^l$ is the weighted mean of the nodes in G_s with the corresponding non-negative weights B, and σ_s^l is the unbiased estimate of the weighted sample variance [72]. Compared with the previous 1-st Wasserstein distance [30], [50], [60], the 2-nd Wasserstein distance under Gaussian assumption circumvents training a multi-layered network and has shown its effectiveness in face recognition [70] and image generation [71]. However, the above 1-st or 2-nd Wasserstein distance treats each data sample equally. The contaminated nodes in G_s will seriously affect the computation of mean and covariance, and may lead to poor knowledge transfer. By contrast, Eq. (14) is based on the weights B from a clear foundation of robustness. The contaminated nodes with low weights contribute little to $\mathbf{m}_s^l$ and σ_s^l, and thus have limited impacts on the distance measurement.

3) The Overall Objective Function and Optimization: Combining the robust Wasserstein distance in Eq. (14) and an L_2-norm regularization, the loss function of training $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$ is described as

$$\min_{\mathbf{W}_t^1,\mathbf{W}_t^2}\sum_l W_2(P_t(\mathbf{z}_t^l),\ P_s(\mathbf{z}_s^l))^2+\gamma(\|\mathbf{W}_t^1\|_F^2+\|\mathbf{W}_t^2\|_F^2),\tag{15}$$

where γ is the trade-off parameter to balance different terms. In Eq. (15), $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$ have unshared and independent weights with $\mathbf{W}_s^1$ and $\mathbf{W}_s^2$, where $\mathbf{W}_s^1$ and $\mathbf{W}_s^2$ are fixed during training. This flexible strategy reaps the following advantages: 1) it allows G_t get more specific feature representations according its graph properties, and thus effectively uncovers the common semantic information among two different graphs; 2) the noisy source nodes suppressed in the first step will have limited impact on training $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$; 3) overly complicated models with degeneration solutions are avoided.

Consequently, the target model is optimally modified to match the source model, and thus the marginal distributions between graphs are adapted (i.e., $P_t(\mathbf{z}_t^l) \approx P_s(\mathbf{z}_s^l)$). We then assume that such source and target models satisfy $P_t(\mathbf{y}_t \mid \mathbf{z}_t^l) \approx P_s(\mathbf{y}_s \mid \mathbf{z}_s^l)$, which is the covariate shift assumption in transfer learning [73]. As a result, the label-discriminative information preserved in the first step is expected to be transferred from G_s to assist the graph analysis tasks on G_t.

Algorithm 2 Step 2: Representation Learning for G_t

Require:
$G_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$, β, γ, d_1, d_2, T, $\{\mathbf{W}_s^1, \mathbf{W}_s^2, \mathbf{W}_s^y\}$, B.
Ensure: Representations $\mathbf{Z}_t^1 \in \mathbb{R}^{d_1 \times m}$, $\mathbf{Z}_t^2 \in \mathbb{R}^{d_2 \times m}$.
1: Initialize $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$ with $\mathbf{W}_s^1$ and $\mathbf{W}_s^2$ respectively.
2: for iterator = 1,2,3,..., T do
3: Backpropagate the loss in Eq. (15), update $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$;
4: end for
5: Return $\mathbf{Z}_t^1$ and $\mathbf{Z}_t^2$.

The overall procedure of iteratively optimizing $\mathbf{W}_t^1$ and $\mathbf{W}_t^2$ is summarized in Algorithm 2. The gradients of the first term in Eq. (15) can be computed as

$$\begin{aligned}
&\frac{\partial W_2(P_s(\mathbf{z}_s^l),\, P_t(\mathbf{z}_t^l))^2}{\partial \mathbf{z}_t^{il}} \\
&= -\frac{1}{n}(\mathbf{m}_s^l - \mathbf{m}_t^l) + (\sigma_s^l - \sigma_t^l)\frac{\partial(\sigma_s^l - \sigma_t^l)}{\partial \mathbf{z}_t^{il}} \\
&= -\frac{1}{n}[(\mathbf{m}_s^l - \mathbf{m}_t^l) + 2(\sigma_s^l - \sigma_t^l)\frac{\mathbf{z}_t^{il} - \mathbf{m}_t^l}{\sqrt{(\sigma_t^l)^2 + \epsilon}}],
\end{aligned}$$

where ϵ is a constant. Note that the target graph has no label access, a target model may quickly learn a degenerate solution without proper initialization. Therefore, the learnt source model is used as the initialization for the target model.

IV. Experiments

In this section, we systematically evaluate different properties of CW-GCN using a set of experiments on multiple real-world datasets. The experimental results demonstrate the robustness of our architecture to different types of noise on single-graph and cross-graph classification tasks, compared with the state-of-the-art methods. The source code of our method is provided in https://github.com/CocoLab-2022/CW-GCN.

A. Robust Single-Graph Node Classification

In this section, our C-loss is tested for traditional single-graph embedding under two conditions successively: no noise; noisy edges and attributes.

1) Data Preparation: Cora [74] and Citeseer [74] are two benchmark graph datasets, where each node represents an article and each edge represents the citation relationship. Each article has only one class label indicating one topic. The features of nodes are sparse bag-of-words vectors indicating whether each unique word is present in each article. These two datasets have been the de facto standard for evaluating graph node classification and their statistics are shown in Table II.

TABLE II
Dataset Statistics in Single-Graph Experiments

Dataset	#Nodes	# Edges	#Features	#Classes
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6

TABLE III
Node Classification Result on Original Graphs

	Cora		Citeseer	
Result (%)	Accuracy		Accuracy	
PDTNet [44]	82.8		72.7	
GCN [6]	81.5		70.3	
C-GCN (ours)	81.7		71.5	
GAT [12]	83.0		72.5	
C-GAT (ours)	82.8		72.6	
Result (%)	Micro-F1	Macro-F1	Micro-F1	Macro-F1
ANRL [75]	77.3	74.8	72.9	67.3
C-ANRL (ours)	77.6	75.1	72.4	67.7

2) Comparison Methods and Experiments Settings: Note that our C-loss function can be easily integrated to various graph embedding models. In this section, we consider three base models GCN [6], ANRL [75] and GAT [12]. Our methods are denoted as C-GCN, C-ANRL and C-GAT which have the same network structure as the base models respectively, expect that C-loss is employed on the output. To further demonstrate the robustness of C-loss, we also include PDTNet [44] in the category of graph augmentation learning. All the comparison methods are implemented based on the source codes provided by the authors and we follow the same parameter settings respectively. The main hyper parameter in our models is the Gaussian kernel width σ and we empirically search it in the range $[1, 10]$.

3) Experimental Results on Original Graphs: We use the train/validation/test splits in [6] and [10], where 20 labels per class are available during training. Table III reports the mean classification results on the test nodes after 20 runs with random weight initializations. The best results of the baselines are taken from the original papers respectively. Note that we report Micro-F1 and Macro-F1 of ANRL and C-ANRL as in [75] for fair comparison. As can be seen from the table, C-loss function achieves similar and comparable results across different base models, showing its effectiveness in handling clean graphs without noise pollution. In particular, PTDNet improves the performances of GCN by actively removing task-irrelevant edges or decreasing their weights.

4) Experimental Results on Noisy Graphs: In this section, we simultaneously add two types of complex noise on the graph datasets. (a) Noisy edges: 30 percent of the labeled training nodes are randomly chosen whose original correct edges are deleted, and then 15 random edges are added on each chosen node. (b) Noisy features: 30 percent of the labeled training nodes are randomly chosen as well whose features are replaced with i.i.d samples from a typical heavy-tailed noise, i.e., Cauchy distribution. The Cauchy noise is centered at 0 with the scale parameter $S = 1$. Table IV summarizes the comparison results after 20 runs with random weight

TABLE IV
NODE CLASSIFICATION RESULT ON NOISY GRAPHS

	Cora		Citeseer	
Result (%)	Accuracy		Accuracy	
PDTNet [44]	69.9		60.3	
GCN [6]	72.1		63.4	
C-GCN (ours)	74.7		65.9	
GAT [12]	75.5		65.7	
C-GAT (ours)	75.8		67.1	
Result (%)	Micro-F1	Macro-F1	Micro-F1	Macro-F1
ANRL [75]	71.4	68.7	66.8	62.9
C-ANRL (ours)	74.0	71.8	69.0	64.7

TABLE V
DATASET STATISTICS IN CROSS-GRAPH EXPERIMENTS

Dataset	#Nodes	# Edges	#Features	#Classes
DBLPv7	5484	8130	6775	5
Citationv1	8935	15113	6775	5
ACMv9	9360	15602	6775	5
Blog1	2300	33471	8189	6
Blog2	2896	53836	8189	6

initializations and random noise pollution. By introducing the C-loss function, our methods consistently outperform the basic models. For instance, the performances of C-GCN over GCN are 74.7% to 72.1% on cora, and 65.9% to 63.4% on citeseer. This loss replacement is simple and does not incur any additional computation cost, yet brings powerful results. These severe corruptions limit the learning cabability of PTDNet and may lead to suboptimal performances in robustness and generalization. In summary, correntropy is a more effective loss to deal with heavily polluted edges and attributes in graph embedding.

B. Robust Cross-Graph Node Classification

1) Data Preparation: DBLPv7 (D7) [76], Citationv1 (C1) [76] and ACMv9 (A9) [76] are three public citation graphs (nodes correspond to articles and edges to citations) from ArnetMiner. They are extracted from three different original sources respectively: DBLP Computer Science Bibliograph, Microsoft Academic Graph and Association for Computer Machinery. Following the previous work [18], [19], the union bag-of-words features are constructed for each node and each node has multiple labels showing its research topics. Blog1 (B1) [77] and Blog2 (B2) [77] are two public social graphs (nodes correspond to bloggers and edges to friendship) from the BlogCatalog dataset. The keywords of the bloggers self-description compose the node attributes and each node is associated with one label indicating its interest group. Dataset statistics are summarized in Table V. Note that compared with the citations graphs, each node in the social graphs has a larger number of neighbors and higher feature dimension. The attribute distributions and the node-to-node interactions across the above graphs are related but varied to some extent. By randomly selecting two graphs as source and target graphs respectively, we construct 8 cross-graph node classification tasks: $D7 \rightarrow C1$, $D7 \rightarrow A9$, $C1 \rightarrow D7$, $C1 \rightarrow A9$, $A9 \rightarrow C1$, $A9 \rightarrow D7$, $B1 \rightarrow B2$ and $B2 \rightarrow B1$.

2) Comparison Methods and Experiments Settings: We systematically compare our method with several state-of-the-art domain adaptation and graph representation algorithms.

- Logistic Regression Classifier (LR) is trained using labeled source nodes and is directly applied for node classification in target graph.
- DANN [41] and WDGRL [50] minimize the distribution gap using node attributes and source node labels. They are not designed for graphs and neglect the powerful node-to-node interactions.
- ASNE [8], ANRL [75], SEANO [9], Planetoid [10], GCN [6] and GAT [12] learn the representations of both graphs without considering the distribution discrepancy. Specifically, ASNE [8] and ANRL [75] utilize structure proximity and node attribute from two graphs. SEANO [9], Planetoid [10], GCN [6] and GAT [12] further introduce abundant node labels from the source graph.
- CDNE [18] and ASN [22] jointly combine structure proximity, node attributes and source node labels for graph embedding. Meanwhile, the distance between two graphs is minimized for graph-invariant representations.
- CGDM+GCN adopts GCN as the feature extractor in the framework of the domain adaptation algorithm CGDM [52].
- CW-GCN is our proposed method for robustly preserving knowledge and minimizing the graph bias.

The proposed CW-GCN is initialized using Glorot initialization and trained using the Adam optimizer.In the first step, the weight of L_2-norm regularization α is chosen from $\{5 \times 10^{-6}, 5 \times 10^{-7}\}$. For example, α is set to 5×10^{-6} on citation graphs and set to 5×10^{-7} on social graphs. The learning rate is 0.01, and the dropout with $p = 0.5$ is applied to the first and the second layers. The maximum epoch is 5000 and we stop training early when the results on the source graph achieve 95%. The Gaussian kernel width σ is chosen from $\{10, 100, 1000\}$. In the second step, the weight of L_2-norm regularization γ is 5×10^{-6}, the learning rate is chosen from $\{0.001, 0.01\}$, and the dropout with $p = 0.1$ is applied to the first and the second layers. We stop training if the training loss does not decrease for 1000 consecutive epochs. In both steps, the hidden dimensionality of the first layer is set to be 256 and the hidden dimensionality of the second layer is set to be 128.

The implementations of the comparison methods realized by the original authors are used. For a fair comparison, the node representations in each baseline are also set to have 128 dimensions. ASNE, ANRL, SEANO and Planetoid sample node sequences from the graph for structure preservation based on skip-gram model [78]. Following the same setting as [18], a unified graph is constructed in the above methods, where the first n nodes are from the source graph and the last m nodes are from the target graph. Therefore, the sequences sampled from the target nodes give more information. Note that the training process of GCN and GAT cannot utilize the target graph since there is no edge between two graphs. For each experiment, all labeled source nodes and unlabeled target nodes are used

TABLE VI
MICRO-F1 AND MACRO-F1 SCORES OF CROSS-GRAPH NODE CLASSIFICATION. THE BEST SCORES ARE IN BOLD FONT

Graphs	F1(%)	*D7* → *C1*	*D7* → *A9*	*C1* → *D7*	*C1* → *A9*	*A9* → *C1*	*A9* → *D7*	Average
LR	Micro	53.82	51.57	56.82	53.83	53.32	53.37	53.79
	Macro	49.73	46.50	53.00	50.39	50.79	49.99	50.07
DANN [41]	Micro	56.27	53.11	57.85	55.53	56.73	55.35	55.81
	Macro	54.13	50.07	55.15	53.45	54.92	52.49	53.37
WDGRL [50]	Micro	56.95	54.52	58.93	56.71	58.19	58.17	57.25
	Macro	53.75	49.70	55.26	53.27	55.65	54.25	53.65
ASNE [8]	Micro	67.60	63.70	64.19	65.15	68.02	64.88	65.59
	Macro	60.07	57.88	60.55	62.98	63.26	60.34	60.85
ANRL [75]	Micro	66.64	63.08	66.03	64.46	68.41	64.48	65.52
	Macro	63.44	60.19	62.78	62.02	65.77	61.03	62.54
Planetoid [10]	Micro	72.69	67.37	70.75	72.42	73.74	69.03	71.00
	Macro	70.58	66.86	68.08	71.41	71.59	66.29	69.14
SEANO [9]	Micro	71.50	66.64	69.31	67.81	72.03	66.13	68.90
	Macro	69.54	65.28	66.94	66.25	70.29	63.33	66.94
GCN [6]	Micro	71.63	66.83	71.24	71.32	73.56	68.22	70.47
	Macro	67.19	62.91	68.12	69.19	70.03	64.13	66.93
GAT [12]	Micro	62.13	55.94	70.94	66.90	71.05	66.43	65.57
	Macro	52.99	46.09	61.56	56.36	64.42	56.23	56.28
CDNE [18]	Micro	79.61	**76.59**	74.15	**77.52**	78.91	72.03	76.47
	Macro	**78.05**	**75.91**	71.71	**76.79**	77.00	69.78	74.87
ASN [22]	Micro	78.38	72.02	**76.98**	73.87	79.63	**76.89**	76.29
	Macro	74.30	73.47	74.97	75.16	76.93	**75.14**	74.99
CGDM [52]+GCN	Micro	77.11	68.08	75.65	72.13	79.43	74.28	75.44
	Macro	76.20	69.29	73.78	73.30	78.15	72.69	73.90
CW-GCN (ours)	Micro	**80.71**	73.68	76.77	76.04	**82.91**	76.74	**77.81**
	Macro	77.97	70.22	**75.77**	75.15	**80.28**	73.73	**75.52**

for training. The labels of the unlabeled target nodes are then predicted by the trained source classifier.

3) Experimental Results on Citation Graphs: In this section, our method CW-GCN is evaluated in three conditions successively: no noise; noisy edges; noisy feaatures.

In the no-noise condition, we report the Micro-F1 and Macro-F1 [79] results on the unlabeled target nodes over 10 random weight initializations in Table VI. Micro-F1 and Macro-F1 are widely used for evaluating classification performance on multi-label datasets. Results for DANN, ANRL, SEANO, GCN and CDNE are taken from the CDNE paper [18]. The results can be summarized as follows.

Firstly, the domain adaptation methods DANN and WDGRL improve the Micro-F1 and Macro-F1 of LR on all the datasets, which verifies that bridging the gap between graphs is necessary in these challenging node classification tasks. Secondly, the traditional graph embedding methods (i.e, ANRL, ASNE, SEANO, Planetoid, SEANO, GCN, GAT) always outperform DANN and WDGRL. The results show that preserving topological structure is important in obtaining valuable graph representations. Thirdly, the graph adaptation methods CDNE, ASN and CW-CCN always achieve higher results than CGDM+GCN. The possible reason is that the specific loss in CGDM designed for non-graph data (e.g., self-supervised loss) is not suitable for complex graph structure. Finally, CW-GCN, ASN and CDNE obtain comparable performance, which successfully distill different kinds of information (i.e., structure proximity, node attributes and node labels) from the source graph and reducing the graph discrepancy.

CW-GCN is proposed with the aim of extracting and transferring reliable information from the severely polluted source graph with noisy edges or features, that is, the graph edges and node attributes do not reflect the likelihood of label agreement. To demonstrate the ability of CW-GCN in detail, we further conduct experiments under two different noisy conditions successively. (a) 10 percent of the labeled source nodes are randomly chosen whose original correct edges are deleted, and then 100 random edges are added on each chosen node. (b) 10 percent of the labeled source nodes are randomly chosen whose features are replaced with i.i.d samples from a Cauchy distribution centered at 0 with the scale parameter $S = 1$. Table VII shows the results over 10 random repetitions in the first noisy condition. Note that LR, DANN and WDGRL will not be affected by the noisy edges. The following observations can be drawn.

Firstly, the performances of ANRL and GAT are a little better than those in the basic cases respectively. A possible explanation for ANRL is that it aims at minimizing the autoencoder loss function between the reconstruction output of each node and its neighbors. Inheriting from denoising autoencoder [80], the loss between the chosen node (i.e., correct attributes) and corrupted version (i.e., partially destroyed neighbors) may yield better performances. A possible explanation for GAT is that it allows for assigning different importance to nodes of a same neighborhood, and thus the chosen nodes may get more informative neighbors from random edges. Secondly, the noisy edges have different levels of negative impacts on the performances of the other methods (i.e., ASNE, SEANO, Planetoid, SEANO, GCN, CDNE, ASN and CGDM+GCN). In particular, the performances of the cross-graph method ASN greatly degrade from 76.29% to 29.50% on the average Micro-F1, from 74.99% to 9.09% on the average Macro-F1. The possible reason is that ASN employs the reconstruction loss to maintain the graph

TABLE VII

MICRO-F1 AND MACRO-F1 SCORES OF CROSS-GRAPH NODE CLASSIFICATIONS WITH NOISY EDGES. THE BEST SCORES ARE IN BOLD FONT

Graphs	F1(%)	*D7* → *C1*	*D7* → *A9*	*C1* → *D7*	*C1* → *A9*	*A9* → *C1*	*A9* → *D7*	Mean
LR	Micro	53.82	51.57	56.82	53.83	53.32	53.37	53.79
	Macro	49.73	46.50	53.00	50.39	50.79	49.99	50.07
DANN [41]	Micro	56.27	53.11	57.85	55.53	56.73	55.35	55.81
	Macro	54.13	50.07	55.15	53.45	54.92	52.49	53.37
WDGRL [50]	Micro	56.95	54.52	58.93	56.71	58.19	58.17	57.25
	Macro	53.75	49.70	55.26	53.27	55.65	54.25	53.65
ASNE [8]	Micro	66.32	59.80	58.77	54.70	68.60	52.72	60.15
	Macro	63.01	50.69	50.12	51.39	57.53	47.69	53.41
ANRL [75]	Micro	67.61	64.55	68.21	66.98	72.09	65.95	67.57
	Macro	64.67	62.16	65.08	65.53	69.62	62.84	64.98
Planetoid [10]	Micro	71.13	66.52	69.97	72.40	72.26	71.03	70.55
	Macro	69.09	66.89	67.27	70.98	70.43	68.24	68.82
SEANO [9]	Micro	61.99	63.12	71.87	67.07	70.22	66.83	66.85
	Macro	58.43	59.17	67.14	62.90	67.87	63.99	63.25
GCN [6]	Micro	70.42	65.88	65.61	63.93	69.92	68.75	67.42
	Macro	58.47	54.75	54.85	52.76	57.91	56.61	55.89
GAT [12]	Micro	69.56	63.83	69.59	64.64	69.10	68.32	67.51
	Macro	64.24	57.05	61.81	54.97	63.09	62.93	60.68
CDNE [18]	Micro	64.10	66.43	65.52	68.69	77.24	68.36	68.39
	Macro	59.41	57.87	61.07	63.88	75.04	65.50	63.80
ASN [22]	Micro	25.99	29.56	32.96	29.56	25.99	32.96	29.50
	Macro	8.25	9.12	9.91	9.12	8.25	9.91	9.09
CGDM [52]+GCN	Micro	60.23	55.88	65.08	57.27	66.19	63.36	65.57
	Macro	50.43	48.21	53.82	48.36	63.46	58.36	53.77
CW-GCN (ours)	Micro	**79.05**	**73.19**	**74.01**	**73.55**	**81.11**	**73.22**	**75.69**
	Macro	**77.80**	**73.05**	**71.92**	**72.92**	**79.60**	**70.55**	**74.31**

TABLE VIII

MICRO-F1 AND MACRO-F1 SCORES OF CROSS-GRAPH NODE CLASSIFICATIONS WITH NOISY ATTRIBUTES. THE BEST SCORES ARE IN BOLD FONT

Graphs	F1(%)	*D7* → *C1*	*D7* → *A9*	*C1* → *D7*	*C1* → *A9*	*A9* → *C1*	*A9* → *D7*	Average
LR	Micro	56.85	53.83	56.91	54.93	53.95	53.02	54.92
	Macro	52.93	48.83	53.26	51.35	52.02	49.14	51.26
DANN [41]	Micro	50.21	49.01	53.15	51.53	53.02	53.13	51.68
	Macro	46.55	45.78	49.22	48.42	49.69	48.55	48.04
WDGRL [50]	Micro	55.36	52.90	58.89	56.08	57.77	57.55	56.43
	Macro	51.80	48.69	54.91	52.56	54.90	53.52	52.73
ASNE [8]	Micro	49.53	44.91	47.28	54.19	40.36	45.35	46.94
	Macro	40.81	35.97	32.45	45.19	29.48	30.98	35.81
ANRL [75]	Micro	40.19	37.93	38.22	36.90	36.39	36.98	37.77
	Macro	33.63	28.27	29.59	29.83	28.65	29.29	29.88
Planetoid [10]	Micro	35.26	31.33	29.90	32.57	27.56	33.65	31.71
	Macro	26.95	19.43	18.81	21.41	16.00	23.20	20.97
SEANO [9]	Micro	23.95	28.13	27.37	29.20	27.35	33.60	28.27
	Macro	9.62	13.25	9.22	13.62	10.40	10.15	11.04
GCN [6]	Micro	72.55	65.87	72.09	71.14	72.86	71.14	70.94
	Macro	60.81	55.19	59.76	58.62	62.76	58.62	59.29
GAT [12]	Micro	60.32	54.13	72.84	68.19	67.95	67.60	65.17
	Macro	51.45	44.06	65.17	58.73	59.23	57.83	56.08
CDNE [18]	Micro	43.15	24.61	21.62	26.24	27.93	32.57	29.35
	Macro	38.24	19.65	13.37	21.47	21.59	28.54	23.81
ASN [22]	Micro	77.18	67.06	74.21	69.32	78.96	71.26	72.99
	Macro	70.40	66.85	69.25	63.51	73.66	65.07	68.12
CGDM [52]+GCN	Micro	26.52	25.25	26.91	26.05	24.58	28.93	25.37
	Macro	16.24	14.77	17.37	16.59	15.72	15.37	16.01
CW-GCN (ours)	Micro	**80.13**	**73.72**	**76.28**	**75.89**	**83.14**	**76.35**	**77.59**
	Macro	**77.71**	**72.98**	**74.91**	**75.66**	**81.24**	**73.82**	**76.05**

structure, while the reconstruction progress is highly influenced by the noisy edges. These results demonstrate that it is hard to extract helpful information from the source graph and to minimize the graph gap under this noisy case. Finally, putting emphasis on promoting robustness, our method consistently obtains the best result by a significant margin in each of the six tasks.

The average Micro-F1 and Macro-F1 results under the second noisy condition are shown in Table VIII and we achieve the following observations. Firstly, LR performs better than

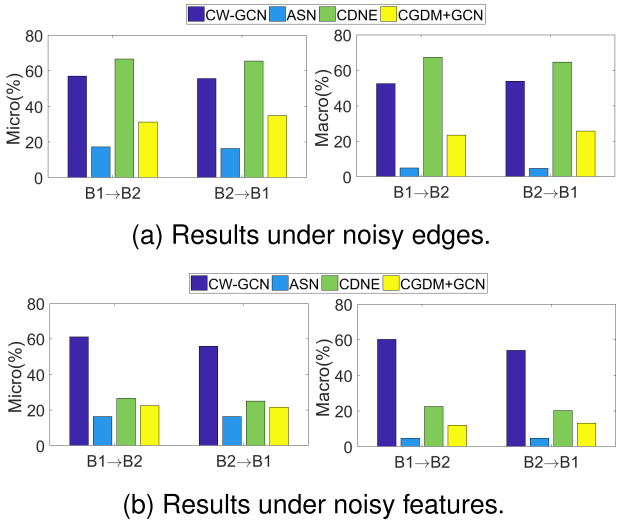


Fig. 2. Micro-F1 and Macro-F1 scores of cross-graph node classifications on social graphs.

TABLE IX
CLASSIFICATION ACCURACY (%) ON CORA AND CITESEER WITH DIFFERENT NOISE LEVELS

	Cora			Citeseer		
Noise Level	I	II	III	I	II	III
GCN [6]	72.12	69.19	72.01	63.43	58.79	62.97
C-GCN (ours)	**74.69**	**70.30**	**74.34**	**65.85**	**60.76**	**64.96**

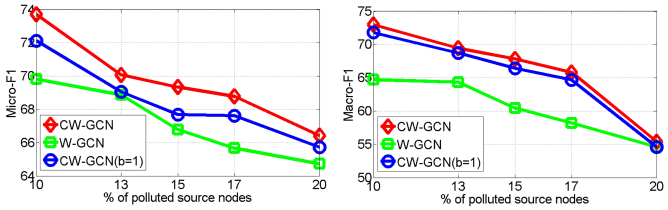


Fig. 3. Micro-F1 and Macro-F1 scores (%) of CW-GCN and its variants on $D7 \rightarrow A9$ with different noise levels.

it under original conditions. The possible reason is that the distributions of source and target graphs may get more overlap with added noise, which may be helpful for direct classification on the target graph. However, the noise makes DANN and WDGRL more difficult to estimate and reduce distribution gap. Secondly, the performances of ANRL, ASNE, SEANO, Planetoid, SEANO, GCN and GAT degrade in different levels with the noisy attributes (e.g., SEANO: -40.63% on the average Micro-F1 and -55.90% on the average Macro-F1). Thirdly, in these challenging tasks, the performances of CDNE and CGDM+GCN are highly deteriorated (e.g., CGDM+GCN: -50.07% on the average Micro-F1 and -57.89% on the average Macro-F1). Finally, the performances of CW-GCN are slightly influenced by the noise. In particular, the performances of CW-GCN on $D7 \rightarrow A9$ and $A9 \rightarrow C1$ are a little better than those in the basic cases respectively. A possible explanation is that after the elimination of noisy nodes, the more informative data play more important roles in training, leading to improved knowledge transfer. In summary, our method achieves much higher Micro-F1 and Macro-F1 results than the other competitors on all the datasets, which illustrates the great robustness of CW-GCN in handling a variety of unpredicted noisy environments.

4) Experimental Results on Social Graphs: We emphasize that the social graphs have higher average degrees than the citation graphs (i.e., nodes have more features and neighbors). In this section, we compare CW-GCN with the graph adaptation methods CDNE, ASN and CGDM+GCN on these difficult datasets in two noisy conditions. (a) Noisy edges: 10 percent of the source nodes are corrupted by deleting the original edges and adding 100 random edges. (b) Noisy features: 10 percent of the source nodes are corrupted by replacing the features with i.i.d samples from a Cauchy distribution centered at 0 with the scale parameter $S = 1$.

Fig. 2 summarizes the results and some advantages can be concluded. ASN and CGDM-GCN have clear underperformances compared with CDNE and CW-GCN in both noisy conditions. As mentioned in the above section, ASN utilizes the reconstructed adjacency matrix to model the structure in graph adaptation and this strategy may have special difficulty in dealing with higher average degrees. Meanwhile, the limited capability of CGDM-GCN reveals that designing loss functions regarding graphs is important in graph adaptation, especially in complex noisy cases. CDNE and CW-GCN obtain comparable performances with noisy edges. The superiority of CDNE may come from using PPMI based model to measure the structural proximities. Note that our correntropy-induced Wasserstein distance can be easily integrated to various GNN models and we leave this possible extension in future. However, in the noisy-attribute tasks, CDNE shows greatly degraded performances. The relative performance gains (Macro) of CW-GCN over CDNE are 37.85% and 33.96% in $B1 \rightarrow B2$ and $B2 \rightarrow B1$, respectively. In summary, our correntropy-induced Wasserstein distance is an effective measure to suppress various kinds of complicated noises, resulting in reliable graph knowledge transfer.

C. Empirical Analysis

1) Robustness of C-GCN Under Different Levels of Noises: We further evaluate the robustness to three different levels of noises. The noise has the following form: p percent of training nodes are randomly chosen whose original correct edges are deleted, and then q random edges are added on each chosen node; p percent of training nodes are randomly chosen as well whose features are replaced with Cauchy distribution. Level I (as shown in Table IV): $p = 30\%$, $q = 15$. Level II: $p = 40\%$, $q = 15$. Level III: $p = 30\%$, $q = 30$. The experiments are randomly repeated 20 times and the classification accuracy is shown in Table IX. With the increasing of noise levels, the performances of C-GCN are consistently better than GCN. These results demonstrate the effectiveness and the robustness of our C-loss function in all kinds of difficult cases.

2) Robustness of CW-GCN Under Different Levels of Noises: We consider two variants of CW-GCN to give an insight into its performance. W-GCN employs the cross-entropy loss in Eq. (15) instead of C-loss in CW-GCN. CW-GCN ($b = 1$) treats each source node equally in the

TABLE X
TRAINING TIME COMPARISONS BETWEEN MODELS. (UNIT: SECONDS)

	C1 → *D7*	*D7* → *A9*	*A9* → *C1*
CDNE [18]	2042	2094	3670
ASN [22]	433	409	562
CW-GCN (ours)	84	88	149

process of minimizing the Wasserstein distance, i.e., $b = 1$ in Eq. (14). Fig. 3 shows the robustness of the variants to different levels of noise on the *D7* → *A9* dataset (the hardest of the six datasets). Specifically, p percent of the labeled source nodes are randomly chosen whose original correct edges are deleted, and then 100 random edges are added on each chosen node, where $p = 10\%, 13\%, 15\%, 17\%, 20\%$. (1) W-GCN achieves worse performances than other methods across all the noise levels, demonstrating that the graph quality greatly influences the embedding results and a robust cost function is desirable to ameliorate the negative influence. (2) CW-GCN ($b = 1$) always performs worse than CW-GCN, which reflects that robust knowledge extraction and robust knowledge transfer are both essential in cross-graph node classification. In this term, our proposed novel Wasserstein distance shows its robustness at transferring knowledge in all kinds of difficult cases.

3) Complexity Analysis: We investigate the computation complexity of the graph adaptation models CDNE, ASN and CW-GCN. Table X illustrates the running time on tasks *C1* → *D7*, *D7* → *A9* and *A9* → *C1*. We found that the two-step WC-GCN achieves highest computational efficiency, which reduces the training time of ASN over 70% on *D7* → *A9* as an example. On this *D7* → *A9* task, the running time of CW-GCN's first step is 30 seconds. Specifically, in each epoch, 0.6 seconds are spent on updating and evaluating the model, and 1.0 seconds are spent on saving the node embedding files. The second step of CW-GCN takes 58 seconds. Note that CDNE suffers the high computation cost since the source code is performed on the CPU platform.

V. CONCLUSION

In graph embedding, knowledge transfer between two different but related graphs is promising and challenging. This paper proposes CW-GCN method to uncover a common latent feature space for source and target graphs under severely noisy environments. Specifically, CW-GCN is implemented in a two-step learning paradigm, allowing independent source and target mappings. Inspired from correntropy, the first step is distinguished with integrating three important sources of information (graph structure, node attribute and node labels) from the noisy source graph. Next, our method aims at efficiently and reliably adapting marginal distributions between graphs with the development of the existing Wasserstein distance. To the best of our knowledge, it is the first attempt in graph embedding field to explain the robustness from the correntropy perspective. Extensive experiments are conducted on five real-world graph datasets. The proposed method generalizes well across a variety of noisy cases, and establishes a new state-of-the-art for single-graph and cross-graph classification.

In the future, we plan to robustly adapt both marginal and conditional distributions between graphs in our architecture. In the challenging applications where source and target graphs are both from noisy environments, we will further investigate how to reduce the impact of the contaminated target nodes on knowledge transfer.

REFERENCES

[1] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, Mar. 2020.

[2] H. Cai, V. W. Zheng, and K. C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[3] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with multi-stream adaptive graph convolutional networks," *IEEE Trans. Image Process.*, vol. 29, pp. 9532–9545, 2020.

[4] C. Yan, Q. Zheng, X. Chang, M. Luo, C. Yeh, and A. G. Hauptman, "Semantics-preserving graph propagation for zero-shot object detection," *IEEE Trans. Image Process.*, vol. 29, pp. 8163–8176, 2020.

[5] B. Zhang, Q. Qiang, F. Wang, and F. Nie, "Flexible multi-view unsupervised graph embedding," *IEEE Trans. Image Process.*, vol. 30, pp. 4143–4156, 2021.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2016, pp. 1–14.

[7] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," in *Proc. IJCAI*, 2016, pp. 1895–1901.

[8] L. Liao, X. He, H. Zhang, and T. Chua, "Attributed social network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2257–2270, Dec. 2018.

[9] J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy, "Semi-supervised embedding in attributed networks with outliers," in *Proc. SDM*, 2018, pp. 153–161.

[10] Z. Yang, W. Cohen, and R. Salakhudinov, "Revisiting semi-supervised learning with graph embeddings," in *Proc. ICML*, 2016, pp. 40–48.

[11] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017, pp. 1024–1034.

[12] X. Wang et al., "Heterogeneous graph attention network," in *Proc. WWW*, 2019, pp. 2022–2032.

[13] G. Qi, C. C. Aggarwal, and T. Huang, "Link prediction across networks by biased cross-network sampling," in *Proc. IEEE 29th Int. Conf. Data Eng. (ICDE)*, Apr. 2013, pp. 793–804.

[14] M. Fang, J. Yin, and X. Zhu, "Transfer learning across networks for collective classification," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, pp. 161–170.

[15] J. Zhang, C. Xia, C. Zhang, L. Cui, Y. Fu, and P. S. Yu, "BL-MNE: Emerging heterogeneous social network embedding through broad learning with aligned autoencoder," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 605–614.

[16] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 119–128.

[17] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *Proc. IJCAI*, 2016, pp. 1823–1829.

[18] X. Shen, Q. Dai, S. Mao, F. Chung, and K. Choi, "Network together: Node classification via cross-network deep network embedding," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 1935–1948, May 2021.

[19] X. Shen, Q. Dai, F.-L. Chung, W. Lu, and K.-S. Choi, "Adversarial deep network embedding for cross-network node classification," in *Proc. AAAI*, 2020, pp. 2991–2999.

[20] Q. Dai, X.-M. Wu, J. Xiao, X. Shen, and D. Wang, "Graph transfer learning via adversarial domain adaptation with graph convolution," 2019, *arXiv:1909.01541*.

[21] Q. Zhu, N. Ponomareva, J. Han, and B. Perozzi, "Shift-robust GNNs: Overcoming the limitations of localized graph training data," in *Proc. NIPS*, 2021, pp. 27965–27977.

[22] X. Zhang, Y. Du, R. Xie, and C. Wang, "Adversarial separation network for cross-network node classification," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 2618–2626.

[23] M. Wu, S. Pan, and X. Zhu, "Attraction and repulsion: Unsupervised domain adaptive graph contrastive learning network," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 5, pp. 1079–1091, Oct. 2022.

[24] L. Yang et al., "HackGAN: Harmonious cross-network mapping using CycleGAN with Wasserstein–Procrustes learning for unsupervised network alignment," *IEEE Trans. Computat. Social Syst.*, vol. 10, no. 2, pp. 746–759, Apr. 2023.

[25] J. Xiao, Q. Dai, X. Xie, Q. Dou, K. Kwok, and J. Lam, "Domain adaptive graph infomax via conditional adversarial networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 1, pp. 35–52, Jan. 2023.

[26] T. Huang, K. Xu, and D. Wang, "GDA-HIN: A generalized domain adaptive model across heterogeneous information networks," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2022, pp. 4054–4058.

[27] Y. Zhuang, C. Shi, C. Yang, F. Zhuang, and Y. Song, "Semantic-specific hierarchical alignment network for heterogeneous graph adaptation," in *Proc. ECML PKDD*, 2021, pp. 335–350.

[28] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proc. AAAI*, 2016, pp. 1145–1152.

[29] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, Jul. 2006.

[30] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. ICML*, 2017, pp. 214–223.

[31] W. Liu, P. P. Pokharel, and J. C. Principe, "Correntropy: Properties and applications in non-Gaussian signal processing," *IEEE Trans. Signal Process.*, vol. 55, no. 11, pp. 5286–5298, Nov. 2007.

[32] B. Chen, X. Liu, H. Zhao, and J. C. Principe, "Maximum correntropy Kalman filter," *Automatica*, vol. 76, pp. 70–77, Feb. 2017.

[33] D. Geman and G. Reynolds, "Constrained restoration and the recovery of discontinuities," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 3, pp. 367–383, Mar. 1992.

[34] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710.

[35] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.

[36] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 1067–1077.

[37] D. Zhu, P. Cui, Z. Zhang, J. Pei, and W. Zhu, "High-order proximity preserved embedding for dynamic networks," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 11, pp. 2134–2144, Nov. 2018.

[38] O. Stretcu, K. Viswanathan, D. Movshovitz-Attias, E. Platanios, S. Ravi, and A. Tomkins, "Graph agreement models for semi-supervised learning," in *Proc. NeurIPS*, 2019, pp. 8713–8723.

[39] S. Pan, R. Hu, S. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2475–2487, Jun. 2020.

[40] X. Zhou et al., "Graph convolutional network hashing," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1460–1472, Apr. 2020.

[41] Y. Ganin et al., "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2030–2096, 2016.

[42] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. NIPS*, 2018, pp. 1–12.

[43] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (CMD) for domain-invariant representation learning," 2017, *arXiv:1702.08811*.

[44] D. Luo et al., "Learning to drop: Robust graph neural network via topological denoising," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 779–787.

[45] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards deep graph convolutional networks on node classification," in *Proc. ICLR*, 2020, pp. 1–18.

[46] S. Liu et al., "Local augmentation for graph neural networks," in *Proc. ICML*, 2022, pp. 14054–14072.

[47] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 66–74.

[48] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Nov. 2010.

[49] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2962–2971.

[50] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *Proc. AAAI*, 2018, pp. 4058–4065.

[51] J. Huang, D. Guan, A. Xiao, S. Lu, and L. Shao, "Category contrast for unsupervised domain adaptation in visual tasks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 1193–1204.

[52] Z. Du, J. Li, H. Su, L. Zhu, and K. Lu, "Cross-domain gradient discrepancy minimization for unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 3936–3945.

[53] H. Tang, K. Chen, and K. Jia, "Unsupervised domain adaptation via structurally regularized deep clustering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8722–8732.

[54] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4888–4897.

[55] L. Luo, L. Chen, and S. Hu, "Attention regularized Laplace graph for domain adaptation," *IEEE Trans. Image Process.*, vol. 31, pp. 7322–7337, 2022.

[56] X. Chen, S. Wang, M. Long, and J. Wang, "Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation," in *Proc. ICML*, 2019, pp. 1081–1090.

[57] H. Tang, X. Zhu, K. Chen, K. Jia, and C. L. P. Chen, "Towards uncovering the intrinsic data structures for unsupervised domain adaptation using structurally regularized deep clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6517–6533, Oct. 2022.

[58] H. Tang, Y. Wang, and K. Jia, "Unsupervised domain adaptation via distilled discriminative clustering," *Pattern Recognit.*, vol. 127, Jul. 2022, Art. no. 108638.

[59] I. Redko, A. Habrard, and M. Sebban, "Theoretical analysis of domain adaptation with optimal transport," in *Proc. ECML-PKDD*, 2017, pp. 737–753.

[60] C. Villani, *Optimal Transport: Old and New*. Cham, Switzerland: Springer, 2008.

[61] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. NIPS*, 2017, pp. 5767–5777.

[62] P. J. Huber, *Robust Statistics*, vol. 523. Hoboken, NJ, USA: Wiley, 2004.

[63] A. Singh, R. Pokharel, and J. Principe, "The C-loss function for pattern classification," *Pattern Recognit.*, vol. 47, no. 1, pp. 441–453, Jan. 2014.

[64] B. Chen, L. Xing, B. Xu, H. Zhao, and J. C. Príncipe, "Insights into the robustness of minimum error entropy estimation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 3, pp. 731–737, Mar. 2018.

[65] B. Chen, L. Xing, X. Wang, J. Qin, and N. Zheng, "Robust learning with kernel mean p-power error loss," *IEEE Trans. Cybern.*, vol. 48, no. 7, pp. 2101–2113, Mar. 2018.

[66] R. He, W. Zheng, and B. Hu, "Maximum correntropy criterion for robust face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1561–1576, Aug. 2011.

[67] K. Xiong, H. H. C. Iu, and S. Wang, "Kernel correntropy conjugate gradient algorithms based on half-quadratic optimization," *IEEE Trans. Cybern.*, vol. 51, no. 11, pp. 5497–5510, Nov. 2021.

[68] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[69] J. Idier, "Convex half-quadratic criteria and interacting auxiliary variables for image restoration," *IEEE Trans. Image Process.*, vol. 10, no. 7, pp. 1001–1009, Jul. 2001.

[70] R. He, X. Wu, Z. Sun, and T. Tan, "Wasserstein CNN: Learning invariant features for NIR-VIS face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1761–1773, Jul. 2019.

[71] D. Berthelot, T. Schumm, and L. Metz, "BEGAN: Boundary equilibrium generative adversarial networks," 2017, *arXiv:1703.10717*.

[72] M. Galassi et al., "Gnu scientific library—Reference manual: Weighted samples," Tech. Rep., 2011.

[73] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," in *Proc. NIPS*, 2017, pp. 3730–3739.

[74] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, Sep. 2008.

[75] Z. Zhang et al., "Anrl: Attributed network representation learning via deep neural networks," in *Proc. IJCAI*, 2018, pp. 3155–3161.

[76] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: Extraction and mining of academic social networks," in *Proc. SIGKDD*, 2008, pp. 990–998.

[77] J. Li, X. Hu, J. Tang, and H. Liu, "Unsupervised streaming feature selection in social media," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2015, pp. 1041–1050.
[78] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. ICLR*, 2013, pp. 1–9.
[79] C. D. Manning, H. Schütze, and P. Raghavan, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge Univ. Press, 2008.
[80] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 1–12, 2010.

Wei Wang received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2009 and 2014, respectively. During the Ph.D. degree, she did research with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, China. She is currently an Associate Professor with the Beijing University of Posts and Telecommunications, China. Her research interests include information theory learning, machine learning, and artificial intelligence.

Gaowei Zhang received the B.E. degree from the Nanjing University of Information Science and Technology in 2009, and the M.Sc. and Ph.D. degrees in computer science from Shanghai University, in 2014 and 2018, respectively. He is currently an Assistant Professor with the Beijing University of Posts and Telecommunications, China. His research interests include machine learning, artificial intelligence, and remote sensing.

Hongyong Han received the B.S. degree in computer science from the Shandong University of Science and Technology, Jinan, China, in 2021. He is currently pursuing the M.S. degree in intelligence science and technology with the Beijing University of Posts and Telecommunications. His current research interests include domain adaptation and artificial intelligence.

Chi Zhang received the B.Sc. degree in computer science from Southwest Jiaotong University in 2007, the M.Sc. degree in traffic control and information engineering from Lanzhou Jiaotong University in 2010, and the Ph.D. degree in computer science from the Institute of Automation, Chinese Academy of Sciences (CASIA), in 2016. In March 2016, he joined CASIA, where he is currently an Associate Professor. His research interests include biometric recognition, biology-inspired visual perception, and understanding for autonomous robots.