Augmenting Knowledge Transfer across Graphs

Yuzhen Mao[§]
Simon Fraser University
Greater Vancouver, BC, Canada
yuzhenm@sfu.ca

Jianhui Sun
University of Virginia
Charlottesville, VA, USA
js9gu@virginia.edu

Dawei Zhou
Virginia Tech
Blacksburg, VA, USA
zhoud@vt.edu

2022 IEEE International Conference on Data Mining (ICDM) | 978-1-6654-5099-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICDM54844.2022.00138

***Abstract*—Given a *resource-rich* source graph and a *resource-scarce* target graph, how can we effectively transfer knowledge across graphs and ensure a good generalization performance? In many high-impact domains (e.g., brain networks and molecular graphs), collecting and annotating data is prohibitively expensive and time-consuming, which makes domain adaptation an attractive option to alleviate the label scarcity issue. In light of this, the state-of-the-art methods focus on deriving domain-invariant graph representation that minimizes the domain discrepancy. However, it has recently been shown that a small domain discrepancy loss may not always guarantee a good generalization performance, especially in the presence of disparate graph structures and label distribution shifts. In this paper, we present TRANSNET, a generic learning framework for augmenting knowledge transfer across graphs. In particular, we introduce a novel notion named trinity signal that can naturally formulate various graph signals at different granularity (e.g., node attributes, edges, and subgraphs). With that, we further propose a domain unification module together with a trinity-signal mixup scheme to jointly minimize the domain discrepancy and augment the knowledge transfer across graphs. Finally, comprehensive empirical results show that TRANSNET outperforms all existing approaches on seven benchmark datasets by a significant margin.**

***Index Terms*—Domain Adaptation, Data Augmentation, Graph Pre-training Strategies**

I. INTRODUCTION

Graph provides a pivotal data structure and a fundamental abstraction for modeling many complex systems, ranging from social science to material science, from financial fraud detection to traffic prediction and many more. The success of convolutional neural networks (CNNs) [11] for grid data has inspired the recent development of graph neural networks (GNNs), which have achieved superior performance on a variety of graph mining tasks such as node classification, link prediction, subgraph matching, and network alignment. Despite the remarkable success, the performance of GNNs is largely attributed to the abundant and high-quality training data. However, in many high-impact domains (e.g., brain networks and molecular graphs), there exist only scarce labels as the data annotation process is prohibitively expensive and time-consuming. Therefore, a fundamental problem is how to transfer knowledge from the resource-rich source graph to the resource-scarce target graph and ensure a good generalization performance.

[§]This work is done as an undergraduate research assistant in Virginia Tech.

Domain adaptation is an attractive solution to tackle this problem, which has received a surge of attention [7, 22] in the graph mining community. The general philosophy is to learn *domain-invariant representations* that do not only achieve satisfactory source domain performances, but also generalize well to the label-scarce target domain. Abundant algorithms [5] and statistical guarantees [1, 2] have been proposed specifically for the independent and identically distributed (i.i.d.) data. However, how to generalize these algorithms and theoretical results to the graph-structured data (i.e., instances are apparently non-iid due to the interconnecting nodes and edges) with heterogeneous graph signals (e.g., node, edges, motifs) is under-explored. Moreover, recent studies [23] have shown that domain-invariant representation may not be able to guarantee a good generalization performance, especially in the presence of disparate graph structures and label distribution shifts, which motivates us to propose novel approach with rigorous guarantees to improve the generalization performance of GNNs across graphs.

Towards this goal, we identify the following two challenges: *C1. Graph Discrepancy*: how to eliminate negative transfer when the source graph and target graph exhibit disparate structures and feature spaces? *C2. Signal Heterogeneity*: how to effectively characterize and leverage graph signals which are heterogeneous (e.g., node, edges, motifs) in both source and target graphs to improve the generalization performance?

In this paper, we propose a generic learning framework named TRANSNET for augmenting knowledge transfer across graphs and show that our proposed approach achieves superior performances universally on all backbone GNNs. The main idea behind our method is a principled way to unify the heterogeneous signals on disparate graphs. To address C1, we develop bi-level gradient reversal layers that learn invariant representations to unify the structure and feature space of the source and target graphs. To address C2, we firstly introduce a novel notion named trinity signal that can naturally formulate various graph signals (e.g., node attributes, edges, and subgraphs). That is to say, we can transform heterogeneous graph signals into a unified format. Building upon this, we propose a data augmentation scheme that automatically conducts interpolation and mixup upon trinity signals to regularize the backbone GNNs with a smooth decision boundary. In general, our contributions are summarized as follows.

- **Problem.** We formalize the *graph signal domain adaptation* problem and identify multiple unique challenges

978-1-6654-5099-7/22/$31.00 ©2022 IEEE
DOI 10.1109/ICDM54844.2022.00138

inspired by the real applications.

- **Algorithm.** We propose a novel method named TRANSNET that (1) unifies the heterogeneous graph signals and dissipate feature spaces and (2) automatically augments the knowledge transfer via trinity-signal mixup.
- **Evaluation.** We systematically evaluate the performance of TRANSNET on seven real graphs by comparing them with eleven baseline models, which verifies the efficacy of TRANSNET. We find that TRANSNET largely alleviates the negative transfer issue and leads up to 9.45% precision improvement over the state-of-the-art methods.
- **Reproducibility.** We publish our data and code at https://github.com/yuzhenmao/TransNet

The rest of our paper is structured as follows. The problem definition is introduced in Section II, followed by the discussion of TRANSNET in Section III. Experimental results are reported in Section IV. In Section V, we review the existing literature before we conclude the paper in Section VI.

II. PROBLEM DEFINITION

In the setting of domain adaptation across graphs, we denote the source graph $\mathcal{G}_s$ and the target graph $\mathcal{G}_t$ in the form of triplets, i.e. $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, \mathbf{X}_s)$ and $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$, where $\mathcal{V}_s$ ($\mathcal{V}_t$) represents the set of nodes, $\mathcal{E}_s$ ($\mathcal{E}_t$) represents the set of edges, and $\mathbf{X}_s$ ($\mathbf{X}_t$) represents the node features in $\mathcal{G}_s$ ($\mathcal{G}_t$). Moreover, we denote the adjacency matrices of $\mathcal{G}_s$ and $\mathcal{G}_t$ as $\mathbf{A}_s$ and $\mathbf{A}_t$ correspondingly. The goal of this paper is to translate the relevant and complementary information from the source graph to the target graph, by addressing graph discrepancy and signal heterogeneity.

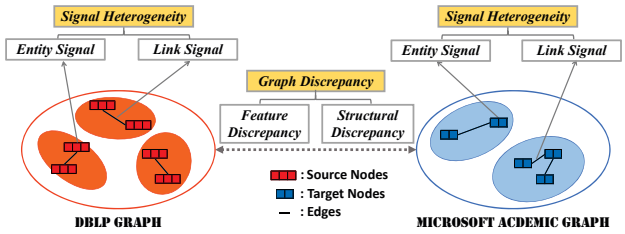

Figure 1. An illustrative example of domain adaptation across DBLP graph and Microsoft Academic Graph.

Problem Definition We consider transferring knowledge learned from the source domain(s) to a target domain with limited labels. Fig 1 presents an illustrative example, which visualizes knowledge transfer from the DBLP Graph ($\mathcal{G}_s$) to the Microsoft Academic Graph ($\mathcal{G}_t$). Here, both source and target domain data could be modeled as graphs. As shown in Fig 1, there are two obstacles, including graph discrepancy and signal heterogeneity during graph domain adaptation. On the one hand, real-world graphs are complex and composed of heterogeneous signals, including entity signals (e.g., nodes, subgraphs) and the corresponding link signals between them. On the other hand, graphs across different domains naturally exhibit disparate distribution in feature representations (e.g., different feature

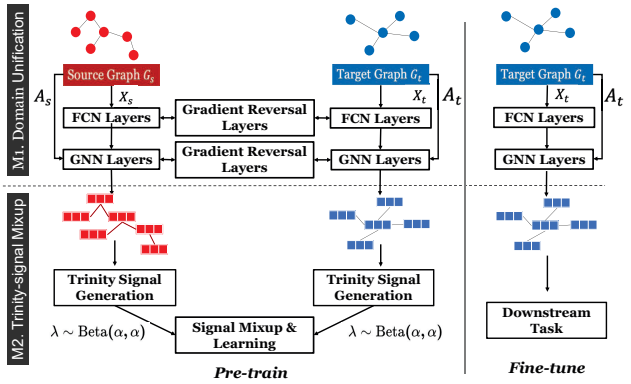

Figure 2. The proposed TRANSNET framework.

dimensions in $\mathcal{G}_s$ and $\mathcal{G}_t$) and structural organizations (e.g., three clusters in $\mathcal{G}_s$ while two clusters in $\mathcal{G}_t$). Given that, we formally define our problem as follows:

Problem 1. Knowledge Transfer across Graphs.
Given: The source graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, \mathbf{X}_s)$ with rich node labels $\mathcal{Y}_s$, the target graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$ with few-shot node labels $\tilde{\mathcal{Y}}_t \in \mathcal{Y}_t$.
Find: Accurate predictions $\hat{\mathcal{Y}}_t$ of unlabeled examples in the target graph $\mathcal{G}_t$.

III. METHODOLOGY

We first review graph pre-training strategies and a theoretical model for domain adaptation before diving into our model.

Graph Pre-training. Graph pre-training strategies [8, 9, 10, 28] provide a powerful tool to parameterize GNNs without label information by predicting easily-accessible graph signals (e.g., node/edge features, context information [8], distance2clusters [10]) extracted from the input graph. In general, the learning objective of existing graph pre-training strategies can be formulated as follows

$$\text{argmax}_\theta \mathbb{E}_{\mathbf{s} \in \mathcal{G}} \log h_\theta(\mathbf{s}|\hat{\mathcal{G}}, \theta) \tag{1}$$

where $\mathcal{G}$ is the input graph, $\hat{\mathcal{G}}$ is the corrupted graph with some masked graph signals $\mathbf{s}$, $h(\cdot)$ is a GNN model with hidden parameters θ. Open research questions lie in how to effectively pre-train GNNs in the presence of heterogeneous graph signals.

Domain Adaptation. A domain consists of a distribution $\mathcal{D}$ on space $\mathcal{X}$ and a labeling function $f : \mathcal{X} \to [0, 1]$. Given two domains, a source domain $\langle \mathcal{D}_s, f_s \rangle$ and a target domain $\langle \mathcal{D}_t, f_t \rangle$, as well as a hypothesis $h : \mathcal{X} \to \{0, 1\}$, we define the risk of the hypothesis $h(\cdot)$ w.r.t. a true labeling function $f(\cdot)$ under distribution $\mathcal{D}$ as $\epsilon(h) = \mathrm{E}_{\mathbf{x} \sim \mathcal{D}}[|h(\mathbf{x}) - f(\mathbf{x})|]$. As a common notion, the empirical risk of a function $h(\cdot)$ on the source domain is defined as $\hat{\epsilon}_s(h)$. Similarly, for the target domain, we use the parallel notation $\epsilon_t(h)$, and $\hat{\epsilon}_t(h)$. In [1] and [2], the generalization bound on the target risk in terms of the empirical source risk and the discrepancy between the source and target domains is derived as follows

Theorem 1 ([2]). With probability at least $1 - \delta$, for every $h \in \mathcal{H}$,

$$\varepsilon_t(h) \leq \widehat{\varepsilon}_s(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}\left(\widehat{\mathcal{D}}_s, \widehat{\mathcal{D}}_t\right) + \lambda + O\left(\sqrt{\frac{d\log(m/d) + \log(1/\delta)}{m}}\right) \tag{2}$$

where $\widehat{\mathcal{D}}_s(\widehat{\mathcal{D}}_t)$ denotes the empirical distribution induced by m samples drawn from $\mathcal{D}_s(\mathcal{D}_t)$; $\mathcal{H}$ denotes a hypothesis class; $d_{\mathcal{H}\Delta\mathcal{H}}$ denotes the distance on $(\widehat{\mathcal{D}}_s, \widehat{\mathcal{D}}_t)$ induced by the symmetric difference hypothesis space; λ denotes the combined risk of the optimal hypothesis; and the last term is a constant which does not depend on any particular $h(\cdot)$.

A. A Generic Learning Framework

In the rest of this section, we propose TRANSNET, a generic learning framework that aims to augment knowledge transfer from the source graph to the target graph. An overview of TRANSNET is presented in Fig 2, which consists of two major modules: *M1. Domain Unification* and *M2. Trinity-signal Mixup*. These two modules are designed to address C1 and C2, correspondingly. In particular, to address the graph discrepancy challenge (C1), M1 automatically unifies the disparate structure and feature distributions of $\mathcal{G}_s$ and $\mathcal{G}_t$ into a domain-invariant hidden space; to address the signal heterogeneity challenge (C2), M2 further unifies the formats of heterogeneous graph signals and conducts manifold mixup [20] operation to achieve a smooth decision boundary. We will further rationale the significance of these two modules with ablation studies (Section IV-B). In the following subsections, we dive into the two modules of TRANSNET in detail.

M1. Domain Unification. Learning invariant representations is crucial for efficient knowledge transfer. One of the standard adversarial approaches is minimizing the distribution discrepancy between domains by Gradient Reversal Layer (GRL) [5]. However, domain adaptation on graph-structured data naturally exhibits the bi-level discrepancy (i.e., feature discrepancy and structural discrepancy), which is illustrated in Figure 1. Different from the previous methods [15, 16, 4], here we propose a bi-level GRL scheme (shown in M1 of Figure 2) to unify the structure and feature space discrepancy of different domains. Firstly, given raw nodes representations $\mathbf{x}_s \in \mathbf{X}_s$ and $\mathbf{x}_t \in \mathbf{X}_t$, we develop domain-specific feature encoder functions that transform $\mathbf{x}_s$ and $\mathbf{x}_t$ to a small domain-invariant hidden space. To eliminate the feature discrepancy, we implement the feature encoder function via Multi-Layer Perceptron (MLP) regularized by GRL. Next, by obtaining the unified node feature representations, we feed them forward to a shared Graph Neural Network (GNN) for extracting domain-invariant structural information, which is also regularized by GRL. By regularizing feature discrepancy and structural discrepancy via M1, we are able to encode $\mathbf{x}_s$ and $\mathbf{x}_t$ into a domain-invariant space. In particular, we formulate the loss function $\mathcal{L}_{\text{domain}}$ of M1 as follow

$$\begin{aligned}\mathcal{L}_{\text{domain}} =& \texttt{Unif}_f + \texttt{Unif}_s \\ =& \underbrace{\texttt{GRL}(\texttt{MLP}(\mathbf{x}_s), \texttt{MLP}(\mathbf{x}_t))}_{\texttt{Unif}_f\text{: feature discrepancy loss}} \\ &\underbrace{+\texttt{GRL}(\texttt{GNN}(\texttt{MLP}(\mathbf{x}_s), \mathbf{A}_s), \texttt{GNN}(\texttt{MLP}(\mathbf{x}_t), \mathbf{A}_t))}_{\texttt{Unif}_s\text{: structural discrepancy loss}}\end{aligned} \tag{3}$$

where $\texttt{Unif}_f$ denotes the feature discrepancy loss, $\texttt{Unif}_s$ denotes the structure discrepancy loss. Without M1, downstream trinity-signal mixup module would potentially blend in unnecessarily redundant signals and thus result in negative transfer [5]. In general, M1 disentangles the domain-specific information by utilizing bi-level GRL and only keeps the domain invariant information, which paves the way for trinity-signal mixup in M2.

M2. Trinity-signal Mixup. As graph-structured data is complex and hierarchical, it naturally exhibits heterogeneous signals. To utilize the information encoded in different signals, existing graph pre-training and domain adaptation approaches treat each signal separately, e.g., [8] and [9] design different pre-train tasks for different signals, while [22] applies an attention scheme to capture the significances of different signals. This could lead to high learning complexity and limit the usage of several useful techniques (e.g., mixup [25] and data poisoning). Here, inspiring from multi-label learning, we propose a generic data structure named trinity signal to unify the representation of heterogeneous graph signals with multi-labels as follows

Definition 1 (Trinity Signal)**.** Given a pair of connected signals $\{\mathbf{s}_i, \mathbf{s}_j\}$ in graph $\mathcal{G}$ together with their representations $\{\mathbf{e}_i, \mathbf{e}_j\}$, the corresponding node labels $\{y_i, y_j\}$ and connection property p_{ij}, the trinity signal representation of $\{\mathbf{s}_i, \mathbf{s}_j\}$ is defined as: $\mathbf{t}_{ij} = \texttt{MLP}([\mathbf{e}_i, \mathbf{e}_j])$ with multi-labels $\mathbf{y}_{ij} = \{y_i, y_j, p_{ij}\}$, where $[\cdot]$ denotes the concatenation operation.

In practice, the trinity signals can be generalized to various graph signals. For instance, when $\mathbf{s}_i$ and $\mathbf{s}_j$ represents a pair of nodes, then $\mathbf{e}_i$ ($\mathbf{e}_j$) denotes the node representation, y_i (y_j) denotes the node label, p_{ij} denotes the weight or proximity score between $\mathbf{s}_i$ and $\mathbf{s}_j$ (e.g., edge existence and personalized PageRank); when $\mathbf{s}_i$ ($\mathbf{s}_j$) denotes a (sub)graph [8], similarly, $\mathbf{e}_i$ ($\mathbf{e}_j$) denotes a (sub)graph representation, y_i (y_j) denotes a (sub)graph label, p_{ij} denotes the (sub)graph distance between $\mathbf{s}_i$ and $\mathbf{s}_j$ (e.g., graph similarity or graph edit distance). In general, trinity signals simultaneously encode entity signals (e.g., nodes, subgraphs) and the corresponding link signals in a principled way.

However, after unifying heterogeneous graph signals, discreteness and non-differentiability still exist in the generated trinity signals, which leads to sub-optimal performance of the model [12]. Mixup [25], a widely adopted data augmentation technique, is a potential approach which has been shown to improve both generalizability and robustness in various domains [26]. Motivated by this, we propose a novel graph mixup strategy named trinity-signal mixup that could be

conducted upon the trinity graph signals. Formally, given two trinity signals $\mathbf{t}$ and $\mathbf{t}'$ with labels $\mathbf{y} = \{y_1, y_2, p\}$ and $\mathbf{y}' = \{y_1', y_2', p'\}$ respectively, we firstly map the trinity signals to a latent space by one linear fully connected layer. Then, a mixup function $\texttt{Mixup}_\lambda(\mathbf{t}, \mathbf{t}')$ generates a new interpolated trinity signal $\tilde{\mathbf{t}}$, where $\lambda \sim \text{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \infty)$ [25]:

$$\tilde{\mathbf{t}} = \texttt{Mixup}_\lambda(\mathbf{t}, \mathbf{t}') = \lambda * \mathbf{t} + (1 - \lambda) * \mathbf{t}' \tag{4}$$

with labels defined $\tilde{\mathbf{y}}$ as:

$$\begin{aligned}\tilde{\mathbf{y}} = \texttt{Mixup}_\lambda(\mathbf{y}, \mathbf{y}') = \{&\lambda * y_1 + (1 - \lambda) * y_1', \\ &\lambda * y_2 + (1 - \lambda) * y_2', \\ &\lambda * p + (1 - \lambda) * p'\}\end{aligned} \tag{5}$$

We also train a multi-label classifier $g(\cdot)$ which outputs the label of trinity signals in $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} = \{\hat{y_1}, \hat{y_2}, \hat{p}\} = g\left(\texttt{Mixup}_\lambda(\mathbf{t}, \mathbf{t}')\right) \tag{6}$$

We define the loss function of trinity-signal mixup as follows

$$\begin{aligned}\mathcal{L}_{signal}(\mathcal{D}, \alpha) = &\mathop{\mathbb{E}}_{(\mathbf{t},\mathbf{y})\sim\mathcal{D}(\mathbf{t}',\mathbf{y}')\sim\mathcal{D}\lambda\sim\text{Beta}(\alpha,\alpha)} \\ &\ell\left(g\left(\texttt{Mixup}_\lambda(\mathbf{t}, \mathbf{t}')\right), \texttt{Mixup}_\lambda(\mathbf{y}, \mathbf{y}')\right)\end{aligned} \tag{7}$$

where $\mathcal{D}$ is a specific data distribution, $(\mathbf{t}, \mathbf{y})$ and $(\mathbf{t}', \mathbf{y}')$ is a pair of labeled examples sampled from distribution $\mathcal{D}$, ℓ is a composite loss function including cross-entropy loss for node classification and mean squared loss for distance regression. In general, trinity signals provide high flexibility for the end users to handle various graph signals at different granularities (e.g., node-level, edge-level, subgraph-level).

B. Algorithm

The overall objective function is defined as follows

$$\mathcal{L}_{total} = \mathcal{L}_{\text{domain}} + \gamma * \mathcal{L}_{\text{signal}} \tag{8}$$

where $\mathcal{L}_{\text{domain}}$ denotes the bi-level GRL loss, $\mathcal{L}_{\text{signal}}$ denotes the trinity-signal loss, and γ is the hyper-parameter that balances the contributions of the two terms.

The procedure for TRANSNET training is presented in Algorithm 1, with Adam as the optimizer. Given the source graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, \mathbf{X}_s)$ with rich labels $\mathcal{Y}_s$; the target graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$ with limited labels $\tilde{\mathcal{Y}}_t \in \mathcal{Y}_t$, we hope to learn a model predicting the node labels of the target graphs.

Algorithm 1 The TRANSNET Learning Framework

Require:
(i) a source graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, \mathbf{X}_s)$ with rich labels $\mathcal{Y}_s$; (ii) a target graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$ with few-shot labels $\tilde{\mathcal{Y}}_t$; (iii) parameter k.

Ensure:
Predictions $\hat{\mathcal{Y}}_t$ of unlabeled examples in $\mathcal{G}_t$

1: Initialize the domain unification model, the trinity-signal classifier $g(\cdot)$, and the classifier $h(\cdot)$ for the downstream task in $\mathcal{G}_t$.
2: **while** not convergent **do**
3: Compute domain-invariant representations of both $\mathcal{G}_s$ and $\mathcal{G}_t$ via domain unification.
4: Generate k trinity signals and apply manifold mixup based on Eq. 4&5.
5: Update the hidden parameters of the domain unification model and the trinity-signal classifier $g(\cdot)$ by minimizing the overall loss function in Eq. 8.
6: **end while**
7: **while** not convergent **do**
8: Fine-tune `MLP` of the target domain, the `GNN` and the classifier $h(\cdot)$ for the downstream task.
9: **end while**

IV. EXPERIMENT

In this section, we demonstrate the performance of our proposed model TRANSNET on seven benchmark datasets by comparing with eleven state-of-the-art baselines.

A. Experiment Setup

Datasets: We evaluate TRANSNET on seven real-world undirected graphs, including five paper citation graphs: Microsoft Academic Graph [15], DBLPv7 [15], DBLPv8 [22], ACMv9_1 [22], ACMv9_2 [15], where nodes represent papers, edges represent a citation relation between two linked nodes; and two co-purchase graphs [14]: Amazon Computers, Amazon Photo, where nodes represent goods, edges represent that two linked goods are frequently bought together. All these seven datasets use bag-of-words encoded features, and each node is associated with one label only. In this paper, we use A1, D1, A2, M2, D2, Comp, Photo to denote ACMv9_1, DBLPv8, ACMv9_2, Microsoft Academic Graph, DBLPv7, Amazon Computers, Amazon Photo, respectively.

Comparison Baselines: We compare TRANSNET with five GNNs, two graph pre-train methods, and four graph transfer learning methods.

GNNs: **GCN** [11], **GAT** [18], **GIN** [24], **GraphSAGE** [6] are four standard graph representation benchmark architectures. **GraphMix** [19] is one of the most popular graph mixup model.

Graph Pre-train: **GPT** [9] pre-trains a GNN by introducing a self-supervised attributed graph generation task. **SelfTask** [10] builds advanced pretext tasks to pre-train the GNN.

Transfer Learning on Graphs: **GPA** [7] is a transferable active learning model. **DANN** [5] is a classical domain adaptation method with GRL. In our experiment, we use GCN as its feature extractor. **UDAGCN** [22] and **ACDNE** [15] are two domain adaptation methods for graph structured data.

For a fair comparison, all baselines contain two GNN hidden layers with $d_1 = 64$ and $d_2 = 32$ for the first and second layers, respectively. The output dimension of GNN is 16. We conduct experiments with only five labeled samples in each class of the target dataset and test based on the rest unlabeled nodes. For UDAGCN and ACDNE having the constraints of shared input features, we follow the instruction from the original papers [22, 15] to build a union set for input features between the source and target domains by setting zeros for unshared features. For classical GNNs (GCN, GAT, GIN, GraphSage), we directly train each model on the target domain for 2000 epochs. For

domain adaptation models (DANN, UDAGCN, ACDNE), after training from the source datasets, they are fine-tuned on the target datasets for 1000 epochs.

For TRANSNET, it is firstly pre-trained on the source dataset for 2000 epochs; then it is fine-tuned on the target dataset for 800 epochs using limited labeled data in each class. We use Adam optimizer with learning rate 3e-3. α in the beta-distribution of trinity-signal mixup is set to 1.0. The output dimension of `MLP` in domain unification module is set to 100. Precision is used as the evaluation metric. We run the experiments with 100 random seeds. The experiments are performed on a Ubuntu20 machine with 16 3.8GHz AMD Cores and a single 24GB NVIDIA GeForce RTX3090.

B. Effectiveness

Comparison Results. We compare TRANSNET with eleven baseline methods across seven real-world undirected graphs. We show the precision of different methods in Table I. In general, we have the following observations: (1) Our proposed TRANSNET consistently outperforms all the baselines on seven datasets, which demonstrates the generalizability and effectiveness of our model. Especially, when adapting knowledge from DBLPv8 to Microsoft Academic Graph with five labeled samples per class, the improvement is more than 10% comparing with the second best model (DANN). (2) Classical GNNs have good performance in several datasets including DBLPv7 and Amazon Computers; but in most instances, they have relatively lower precision. For example, in dataset ACMv9_2, with five labeled samples per class, the best precision is 50.18% achieved by GNN, which is 5% lower than GPT and 14% lower than TRANSNET. The reason is that these models don't make use of the additional knowledge from the source graph, which leads to relatively worse performance especially when the labeled samples are limited. (3) Graph pre-train models sometimes achieve significant improvement: SelfTask and GPA outperform all classical GNNs in dataset ACMv9_2 and DBLPv7 respectively. But compared with TRANSNET, they have relatively poor generalization performance since these pre-train models do not consider the graph discrepancy so that they cannot make use of the knowledge from the resource-rich source domains. (4) Graph transfer learning models such as DANN and UDAGCN could achieve better performance than classical GNNs and graph pre-train models. Particularly, DANN outperforms all the models except TRANSNET in datasets ACMv9_2, Microsoft Academic Graph, and DBLPv7 with both three or five labeled samples per class. However, TRANSNET could still beat graph transfer learning models in every dataset. For example, in datasets ACMv9_2, Microsoft Academic Graph, and DBLPv7, TRANSNET outperforms all listed graph transfer learning models by at least 5% precision. Comparing with graph transfer learning models, the key advantage of TRANSNET lies in the trinity-signal mixup that could handle signal heterogeneity and reduce the learning complexity simultaneously.

Ablation Study. Considering that TRANSNET consists of various components, we set up the experiments to study the effect of different components by removing one component from TRANSNET at a time. The ablation results are presented in Table II. From the results, we have several interesting observations. (1) Adding node signals could make a huge improvement to label prediction precision. (2) Although adding link signals does not help much in the node classification task (which is reasonable since link signals have no direct connection with node signals), it does not reduce the precision either, which means our model could encode those two signals well simultaneously. (3) Although removing the target domain label information could still transfer knowledge, adding target domain influence during the pre-training does make knowledge adaptation even better. (4) Both two domain losses help the model better adapt knowledge from the source to the target domain, which proves the effectiveness of bi-level GRL in alleviating the graph discrepancy. (5) Trinity-signal Mixup also helps the model to adapt knowledge better by at most 4% (DBLPv8 $\rightarrow$ Microsoft Academic Graph).

V. RELATED WORK

Pre-Training for Graphs. Graph pre-training generalizes knowledge to downstream tasks by capturing the structural and semantic properties of input graphs. The current graph pre-training strategies can be summarized into two different categories: 1) Using mutual information maximization between different graph structures which are generated from various corruption functions [17]; 2) Utilizing feature generation or edge generation by masking [9]. Besides, [8] pre-trains a graph at the level of both individual nodes and the entire graph. However, these existing methods cannot transfer knowledge from other domains.

Domain Adaptation. Domain adaptation methods provide potential approach to efficiently transfer knowledge from the source graph to the target graph with disparate structures and label distributions. There are majorly three techniques used for realizing the Domain Adaptation algorithm: 1) Divergence based [13, 29]; 2) Adversarial based [5]; 3) Reconstruction based [3]. Recent researches which apply domain adaptation techniques to graph dataset [22, 15, 16, 4] only focus on the setting of shared input feature. To the best of our knowledge, graph domain adaptation based on two different input spaces and two output label-sets has received little attention in the machine learning community.

Mixup for Data Augmentation. Mixup and its variants [25, 20] are interpolation-based and widely-adopted data augmentation techniques for regularizing neural networks. More recently, mixup is applied to graph dataset. [19] proposes to train an auxiliary Fully-Connected Network which uses the node features to implement Manifold Mixup. [27] aims to train an edge generator through the task of adjacency matrix reconstruction. [21] mixes the receptive field subgraphs for the paired nodes. These previous works ignore the mixup in the link level or need to use additional networks , which is far less elegant, efficient and accurate.

Table I
COMPARISON OF DIFFERENT METHODS USING 5 LABELED SAMPLES PER CLASS (% TEST PRECISION).

Source	Target	GCN	GAT	GIN	GraphSAGE	GraphMix	GPT-GNN	SelfTask	GPA	DANN	UDAGCN	ACDNE	TRANSNET
Photo	Comp	67.24	65.81	66.37	71.26	42.13	62.75	63.18	60.22	71.74	73.13	24.55	**76.54**
Comp	Photo	79.17	71.58	75.32	84.56	74.36	75.63	76.80	71.36	83.75	81.24	33.38	**87.67**
A1	A2	50.18	46.90	43.56	45.63	48.64	55.04	46.60	52.02	55.34	39.33	33.14	**64.11**
D1									51.60	56.35	38.66	31.71	**65.34**
A1	M2	59.60	51.86	47.88	53.17	55.67	64.27	54.75	62.53	65.75	45.90	43.77	**73.53**
D1									61.63	64.63	45.20	43.11	**74.20**
A1	D2	58.30	53.39	45.07	52.16	51.59	51.84	59.05	57.50	60.01	42.36	40.93	**66.75**
D1									56.89	61.87	42.26	40.17	**67.95**
A2	A1	63.36	62.23	46.82	57.70	60.17	58.53	56.73	59.05	62.22	61.23	42.64	**65.99**
M2									56.56	61.63	59.27	44.75	**64.74**
D2									58.31	61.58	60.21	43.10	**64.46**
A2	D1	94.74	97.33	96.81	94.75	91.15	64.64	91.67	68.78	95.01	91.72	29.17	**97.95**
M2									69.99	95.10	93.87	26.70	**97.71**
D2									71.27	95.44	93.54	33.45	**97.91**

Table II
ABLATION STUDY USING 5 LABELED SAMPLES PER CLASS. MEAN AND STANDARD DEVIATION ARE REPORTED OVER FIFTY RANDOM TRIALS.

Ablation	A1 $\rightarrow$ M2	D1 $\rightarrow$ M2	A1 $\rightarrow$ A2	D1 $\rightarrow$ A2
Without node signals in source & target domain	60.06 ± 5.91	58.14 ± 6.05	49.20 ± 4.94	49.10 ± 5.86
Without link signals in source & target domain	73.51 ± 3.93	73.72 ± 3.79	63.78 ± 5.08	65.28 ± 4.79
Without target domain node and link signals	70.43 ± 3.91	70.55 ± 3.91	60.29 ± 3.76	59.95 ± 3.78
Without $\texttt{Unif}_f$	50.07 ± 9.49	55.49 ± 13.51	47.01 ± 8.89	41.36 ± 10.01
Without $\texttt{Unif}_s$	67.80 ± 4.05	67.50 ± 3.21	58.69 ± 4.68	57.46 ± 4.18
Without $\texttt{Unif}_f$ & $\texttt{Unif}_s$	61.86 ± 9.20	54.51 ± 9.36	49.31 ± 10.03	49.72 ± 7.83
Without Trinity-signal Mixup	70.54 ± 4.15	70.83 ± 3.53	62.57 ± 4.27	61.22 ± 5.31
TRANSNET	**73.53** ± 4.13	**74.20** ± 3.64	**64.11** ± 4.75	**65.34** ± 5.26

VI. CONCLUSION

In this paper, we present TRANSNET, a generic learning framework for augmenting knowledge transfer across different graphs via multi-scale graph signal mixup. It consists of two major parts: Domain Unification and Trinity-signal Mixup, which give potential approaches to two challenges: *C1. Graph Discrepancy* and *C2. Signal Heterogeneity* respectively. Extensive experimental results demonstrate the efficacy of our method for knowledge transfer across graphs.

REFERENCES

[1] S. Ben-David et al. "A theory of learning from different domains". In: *Machine learning*. 2010.

[2] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. "Learning bounds for domain adaptation". In: *NeurIPS*. 2007.

[3] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. "Domain separation networks". In: *NeurIPS*. 2016.

[4] Q. Dai, X.-M. Wu, J. Xiao, X. Shen, and D. Wang. "Graph Transfer Learning via Adversarial Domain Adaptation with Graph Convolution". In: *TKDE*. 2022.

[5] Y. Ganin et al. "Domain-adversarial training of neural networks". In: *JMLR*. 2016.

[6] W. Hamilton, Z. Ying, and J. Leskovec. "Inductive representation learning on large graphs". In: *NeurIPS*. 2017.

[7] S. Hu et al. "Graph policy network for transferable active learning on graphs". In: *NeurIPS*. 2020.

[8] W. Hu et al. "Strategies for pre-training graph neural networks". In: *ICLR*. 2020.

[9] Z. Hu, Y. Dong, K. Wang, K.-W. Chang, and Y. Sun. "Gpt-gnn: Generative pre-training of graph neural networks". In: *SIGKDD*. 2020.

[10] W. Jin et al. "Self-supervised learning on graphs: Deep insights and new direction". In: *arXiv preprint arXiv:2006.10141*. 2020.

[11] T. N. Kipf and M. Welling. "Semi-supervised classification with graph convolutional networks". In: *ICLR*. 2017.

[12] L. Liu, M. Wang, and J. Deng. "A unified framework of surrogate loss by refactoring and interpolation". In: *ECCV*. 2020.

[13] M. Long, Y. Cao, J. Wang, and M. Jordan. "Learning transferable features with deep adaptation networks". In: *ICML*. 2015.

[14] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann. "Pitfalls of graph neural network evaluation". In: *arXiv preprint arXiv:1811.05868*. 2018.

[15] X. Shen, Q. Dai, F.-l. Chung, W. Lu, and K.-S. Choi. "Adversarial deep network embedding for cross-network node classification". In: *AAAI*. 2020.

[16] X. Shen, Q. Dai, S. Mao, F.-l. Chung, and K.-S. Choi. "Network together: Node classification via cross-network deep network embedding". In: *IEEE Trans. Neural Networks Learn. Syst.* 2020.

[17] P. Velickovic et al. "Deep Graph Infomax." In: *ICLR*. 2019.

[18] P. Veličković et al. "Graph attention networks". In: *ICLR*. 2018.

[19] V. Verma et al. "GraphMix: Improved Training of GNNs for Semi-Supervised Learning". In: *AAAI*. 2021.

[20] V. Verma et al. "Manifold mixup: Better representations by interpolating hidden states". In: *ICML*. 2019.

[21] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi. "Mixup for node and graph classification". In: *WWW*. 2021.

[22] M. Wu, S. Pan, C. Zhou, X. Chang, and X. Zhu. "Unsupervised domain adaptive graph convolutional networks". In: *WWW*. 2020.

[23] Y. Wu, E. Winston, D. Kaushik, and Z. Lipton. "Domain adaptation with asymmetrically-relaxed distribution alignment". In: *ICML*. 2019.

[24] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. "How powerful are graph neural networks?" In: *ICLR*. 2019.

[25] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. "mixup: Beyond empirical risk minimization". In: *ICLR*. 2018.

[26] L. Zhang, Z. Deng, K. Kawaguchi, A. Ghorbani, and J. Zou. "How Does Mixup Help With Robustness and Generalization?" In: *ICLR*. 2021.

[27] T. Zhao, X. Zhang, and S. Wang. "GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks". In: *WSDM*. 2021.

[28] D. Zhou, L. Zheng, D. Fu, J. Han, and J. He. "MentorGNN: Deriving Curriculum for Pre-Training GNNs". In: *CIKM* (2022).

[29] D. Zhou, L. Zheng, Y. Zhu, J. Li, and J. He. "Domain adaptive multi-modality neural attention network for financial forecasting". In: *WWW*. 2020, pp. 2230–2240.