A Dual-channel Semi-supervised Learning Framework on Graphs via Knowledge Transfer and Meta-learning

ZIYUE QIAO, Jiangmen Laboratory of Carbon Science and Technology, China; Computer Network Information Center, Chinese Academy of Sciences; and Guangzhou HKUST Fok Ying Tung Research Institute
PENGYANG WANG, SKL-IOTSC, University of Macau, China
PENGFEI WANG and ZHIYUAN NING, Computer Network Information Center, CAS, China
YANJIE FU, Department of Computer Science, University of Central Florida, USA
YI DU and YUANCHUN ZHOU, Computer Network Information Center, CAS, China
JIANQIANG HUANG and XIAN-SHENG HUA, Damo Academy, Alibaba Group, China
HUI XIONG, Hong Kong University of Science and Technology (Guangzhou) and Guangzhou HKUST Fok Ying Tung Research Institute, China

This article studies the problem of semi-supervised learning on graphs, which aims to incorporate ubiquitous unlabeled knowledge (e.g., graph topology, node attributes) with few-available labeled knowledge (e.g., node class) to alleviate the scarcity issue of supervised information on node classification. While promising results are achieved, existing works for this problem usually suffer from the poor balance of generalization and fitting ability due to the heavy reliance on labels or task-agnostic unsupervised information. To address the challenge, we propose a dual-channel framework for semi-supervised learning on **G**raphs via **K**nowledge **T**ransfer between independent supervised and unsupervised embedding spaces, namely, GKT. Specifically, we devise a dual-channel framework including a supervised model for learning the label probability of nodes and an unsupervised model for extracting information from massive unlabeled graph data. A knowledge transfer head is proposed to bridge the gap between the generalization and fitting capability of the two models. We use the unsupervised information to reconstruct batch-graphs to smooth the label probability distribution on the graphs to improve the generalization of prediction. We also adaptively adjust the reconstructed graphs by encouraging the label-related connections to solidify the fitting ability. Since the optimization of the supervised channel with knowledge transfer contains that of the unsupervised channel as a constraint and vice versa, we then propose a meta-learning-based method to solve the bi-level optimization problem, which avoids the negative transfer and further improves the model's performance. Finally, extensive experiments validate the effectiveness of our proposed framework by comparing state-of-the-art algorithms.

This research was supported by the Foshan HKUST Projects (FSUST21-FYTRI01A, FSUST21-FYTRI02A), the Natural Science Foundation of China under Grant No. 61836013, and the Strategic Priority Research Program of CAS XDB38030300.
Authors' addresses: Z. Qiao, Jiangmen Laboratory of Carbon Science and Technology, China; Computer Network Information Center, Chinese Academy of Sciences; and Guangzhou HKUST Fok Ying Tung Research Institute; email: qiaoziyue@cnic.cn; P. Wang, SKL-IOTSC, University of Macau, China; email: pywang@um.edu.mo; P. Wang, Z. Ning, Yi Du, and Y. Zhou (corresponding author), Computer Network Information Center, CAS, China; emails: wpf2106@gmail.com, ningzhiyuan@cnic.cn, duyi@cnic.cn, zyc@cnic.cn; Y. Fu, Department of Computer Science, University of Central Florida, USA; email: yanjie.fu@ucf.edu; J. Huang and X.-S. Hua, Damo Academy, Alibaba Group, China; emails: jianqiang.hjq@alibaba-inc.com, xiansheng.hxs@alibaba-inc.com; H. Xiong (corresponding author), Hong Kong University of Science and Technology (Guangzhou) and Guangzhou HKUST Fok Ying Tung Research Institute, China; email: huixiong@ust.hk.

1559-1131/2024/01-ART18 $15.00
https://doi.org/10.1145/3577033

CCS Concepts: • **Theory of computation** → **Semi-supervised learning**; • **Computing methodologies** → *Unsupervised learning*; *Transfer learning;*

Additional Key Words and Phrases: Social network, meta learning

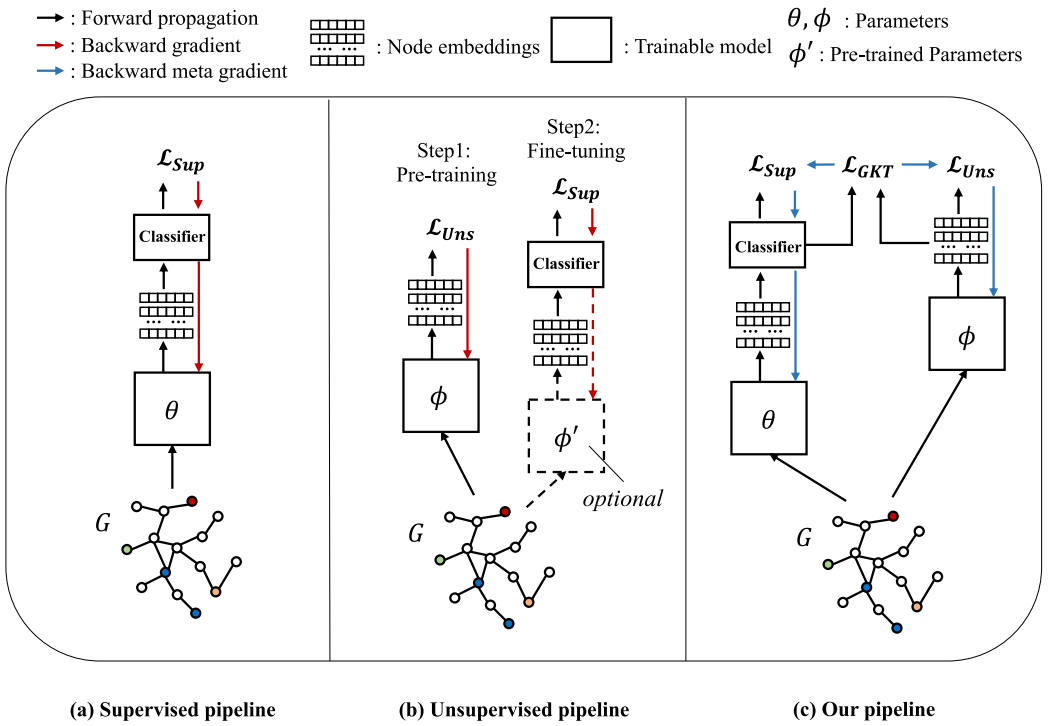
ACM Reference format:
Ziyue Qiao, Pengyang Wang, Pengfei Wang, Zhiyuan Ning, Yanjie Fu, Yi Du, Yuanchun Zhou, Jianqiang Huang, Xian-Sheng Hua, and Hui Xiong. 2024. A Dual-channel Semi-supervised Learning Framework on Graphs via Knowledge Transfer and Meta-learning. *ACM Trans. Web* 18, 2, Article 18 (January 2024), 26 pages.
https://doi.org/10.1145/3577033

1 INTRODUCTION

Semi-Supervised Learning on Graphs (SSL-G) is a fundamental task on graph data, which usually refers to the node classification problem on graphs, where labels are only available for a small subset of nodes. The problem of SSL-G exists in various real-world applications, such as image/video recognition [5, 25, 63], anomaly detection [1, 49], event detection [30, 61], biological sequence analysis [47]. Research on the SSL-G model has drawn extensive attention during past decades. Generally, the acquisition of labeled nodes is usually quite expensive and time-consuming, especially involving manual effort. The deficiency of labeled nodes makes supervised models easily over-fitting. At the same time, the unlabeled graph data is usually easily and cheaply collected. Therefore, mainstream methods incorporate ubiquitous unlabeled knowledge (e.g., graph topology, node attributes) with few-available labeled knowledge (e.g., node classes) to alleviate the scarcity issue of supervised information. They mainly focus on exploiting the intrinsic distribution disclosed by the unlabeled data to facilitate generalizing the learned models.

Basically, based on the learning pipeline, recently proposed methods can be divided into two categories—supervised methods and unsupervised methods, as shown in Figure 1. Supervised methods formulate SSL-G as a unified supervised learning problem end-to-end. They usually encode the nodes' local neighborhood with unlabeled information into hidden node embeddings and feed them into the classifier [20, 29], which is inherently a data augmentation that exploits proximity between nodes to estimate labels for unlabeled data. Unsupervised methods break SSL-G into two stages—pre-training & fine-tuning. Pre-training aims to optimize self-supervised learning objectives over the whole unlabeled graph and obtains a pre-trained model [17, 35, 37] or unsupervised node embeddings [13, 36, 53, 55]; fine-tuning takes the knowledge learned by pre-training as a good initialization and further trains a lightweight classifier on it for supervised learning tasks.

Although these methods use different techniques to alleviate the impact of label scarcity, they still face certain limitations: (i) **Supervised methods suffer from heavy label reliance, weak robustness, and low generalization.** For example, the label propagation-based model [67] explores the pair-wise affinity between nodes to infer the labels of the given unlabeled nodes, and the graph neural networks [20, 29] predict the node classes via a message-passing mechanism on graph topology. However, these methods are only optimized on the given limited labeled data and heavily rely on the goodness-of-fit of the model. Some methods [22, 39, 45, 48] design self-supervised objectives as the explicit graph regularization term and train it along with the supervised objectives as an auxiliary task. However, the hard parameter sharing for two tasks limits the representation ability of node embeddings. It may also lead to one of the tasks dominating training to dampen the classification due to different scales of losses. (ii) **Unsupervised methods with target-agnostic optimization objectives have a poor fitting ability.** For example, the contrastive learning-based models usually extract graph structure information such as node



(a) Supervised pipeline **(b) Unsupervised pipeline** **(c) Our pipeline**

Fig. 1. The learning pipelines of supervised methods, unsupervised methods, and our method for semi-supervised learning on graphs. Notably, the labeled nodes are colored in the graph G, where each color represents one class. (a) Supervised methods train the node classifier in an end-to-end schema; (b) Unsupervised methods first pre-train a graph encoder without labels and fine-tune the encoder or node representations for classification; (c) Our article proposes a new training pipeline that co-trains two channels of pipelines and uses meta gradient to upgrade their parameters.

contexts [33, 34] and local sub-graph [53] as the pseudo labels to train the graph encoder to learn the underlying patterns of the graph data. However, because self-supervised learning objectives are task-agnostic, the unsupervised models may learn generalized node embeddings but are naturally sub-optimal for the downstream tasks, and the fine-tuning as supervised learning with limited labels still faces the same over-fitting problem as the supervised models.

To address these limitations, we design a novel semi-supervised graph learning framework by co-training two independent channels of the supervised and unsupervised methods and transferring knowledge between them to balance their fitting and generalization abilities. Specifically, to incorporate knowledge from one channel to another, instead of hard sharing or transferring the parameters/embeddings between two channels, we suppose the dual-channel outputs should favor different semantic spaces and adopt a knowledge transfer head above two channels to transfer the soft distributions-based knowledge. We consider the correlation between supervised and unsupervised information. In the knowledge transferring, we reconstruct a batch graph via the output node embeddings of the unsupervised channel to smooth the classification probability of the supervised channel between the connected nodes. Meanwhile, the reconstructed graph is adaptively adjusted with the guide of supervised information to be related to the high-confidence label distributions, which would conversely further improve the performance of the supervised channel. During such processes, we bridge the gap between the generalization and fitting capability of the

two channels. The above objective can be formulated as a bi-level optimization problem and can be solved by meta-learning. We propose a meta-learning-based dual-channel training algorithm to better transfer the learned knowledge of two channels to each other, in which channel losses are viewed as the objective of the inner-learner and the knowledge transfer loss is put into the objective of the outer-learner to transfer the knowledge between two channels. The main contribution of our article is summarized as follows:

- We propose a novel semi-supervised learning framework on graphs, namely, GKT, to incorporate the unlabeled knowledge to alleviate the scarcity of supervised information. Different with the traditional supervised and unsupervised methods specialized in fitting and generalization capability, GKT stacks two independent channels of both and a knowledge transfer head that can well balance fitting and generalization in SSL-G.
- We propose a graph knowledge transfer model that utilizes unlabeled graph data to reconstruct graphs to smooth label distributions to improve the model generalization, while it adaptively adjusts the reconstructed graph by encouraging label-related connections.
- We propose a meta-learning-based training algorithm to handle the bi-level optimization problem on the dual-channel model, which avoids negative transferring and further improves the model's robustness and performance.
- We compare GKT with state-of-the-art supervised and unsupervised methods on various datasets and conduct extensive experiments to analyze it. Various experimental results verify the effectiveness of GKT.

2 RELATED WORKS

In this section, we introduce the related works of semi-supervised learning on graphs, knowledge transfer, and meta-learning.

2.1 Semi-supervised Learning on Graphs

SSL-G can also be named as graph-based semi-supervised learning, which has a large number of successful applications across, such as computer vision [28, 41, 50], natural language processing [26, 44], social networks [3, 12, 42], and biomedical science [24]. As introduced in Section 1, the SSL-G methods can be divided into two categories based on their learning pipelines—supervised methods and unsupervised methods. Supervised methods are based on the assumption that nearby nodes are likely to have the same label. Among those, label propagation is a representative method. The basic idea is to transfer the labels of labeled nodes to unlabeled ones through the graph topology based on the similarity of each node pair. One popular technology is regularizing a supervised classifier with a Laplacian regularizer or an embedding-based regularizer, such as deep semi-supervised embedding [54] and Planetoid [58]. Our proposed knowledge transfer model can be viewed as a label propagation model to some extent, but the graph topology we constructed is adaptively optimized in training. Another widely used technology is **graph neural networks (GNNs)**, which have demonstrated the superiority of modeling graph structure and node attributes. In the propagation of GNNs, the node embeddings are improved for classification by some forms of the weighted average of their neighborhood, such as GCN [20], GraphSAGE [13], GAT [52], SGC [56], GIN [57]. However, GNNs without explicit optimization on the generalization of node embeddings are also easily over-fitting under limited label information. Unsupervised methods usually pre-train a pretext task to capture the underlying patterns of the graph data into node embeddings, the pretext task is usually designed to predict the central nodes given node context and sub-graph contexts, such as DeepWalk [33], S^2GRL [31], and Subg-Con [19], or maximize the mutual information between local and global graph, such as DGI [53] and GMI [32]. Despite the

noticeable achievements of these two categories of methods, how to adaptively balance the generalization and fitting capability of models has not been well addressed. Recently, some methods also introduced meta-learning into SSL-G to alleviate the over-fitting problem on limited label information. Meta-GNN [64] proposes a novel graph meta-learning framework. It obtains the prior knowledge of classifiers by training on many similar few-shot learning tasks and then classifies the nodes from new classes with only a few labeled samples. G-Meta [18] uses local sub-graphs to transfer sub-graph-specific information and learn transferable knowledge faster via meta gradients. Compared with them, we are the first to explicitly incorporate an unsupervised model into meta-learning-based training in SSL-G.

2.2 Knowledge Transfer

In a broad sense, knowledge transfer, also is transfer learning, is a range of methods that exploit some forms of learned knowledge (such as soft distributions) from the source model as additional supervision for training the target model. In semi-supervised learning scenarios, works usually treat the supervised task as the target task and the unsupervised learning task as the source task, which aims to transfer the intrinsic data distribution information from a large amount of unlabeled data to improve the performance of the classification task. Graph pre-training methods [17, 35, 37] can be viewed as a traditional transfer learning pipeline on graph data. Usually, they pre-train a generalized model on the massive unsupervised graph data and then transfer the model to the target graph for the supervised task via a few fine-tuning steps. The problem with graph pre-training methods is that they usually need to prepare a large number of extra graph data and need expensive training costs. In our article, we conveniently use the target graph data to transfer the knowledge of unsupervised graph patterns to improve the generalization of supervised learning. Moreover, traditional semi-supervised learning models draw both the labeled and unlabeled samples on the same space. In contrast, in transfer learning, the data distributions of the source and the target domains are usually different. Thus, this article sets two channels for individually extracting supervised and unsupervised information and proposes a transfer learning-based model for semi-supervised learning, which avoids the interference of different information on the distribution and improves the quality of classification probability distribution. Recent work has proposed some knowledge transfer models related to semi-supervised learning on graphs. For example, GAKT [7] develops a novel graph adaptive knowledge transfer model for unsupervised domain adaptation task to jointly optimize target labels and domain-free features in a unified framework. GFL [59] proposes a graph few-shot learning algorithm that incorporates prior knowledge learned from auxiliary graphs to improve classification accuracy on the target graph. In our article, we actually use the proximity between unsupervised node embeddings and classification probability in their own semantic spaces as knowledge to transfer between two channels. And we train the knowledge transfer model in a co-training schema, i.e., we train dual-channel models at the same time and transfer their information to each other.

2.3 Meta-learning

Meta-learning, also named *learning to learn*, has seen a rising interest in recent research. Meta-learning can be viewed as an optimization technique that can greatly improve the generalization ability of the model via meta-gradient parameter updating. It has been widely applied for improving data efficiency, unsupervised learning, and knowledge transfer [15]. Typically, MAML [8] formulates the meta-learning problem in a nested optimization format, where the inner loop imitates the process of adaptation, while the outer loop focuses on optimizing the meta-objective. The inner optimization is further replaced by a single SGD step so the meta-objective can be optimized in an end-to-end manner. Recently, some work also introduced meta-learning to improve the

optimization of knowledge transfer objectives, where meta-learning is usually used to solve the bilevel optimization problem in the framework. For example, Franceschi et al. [10] propose a framework based on bi-level programming that optimizes both the hyper-parameters and the parameters of ground models. Han et al. [14] propose a new transfer learning paradigm on GNNs that could effectively leverage self-supervised tasks as auxiliary tasks to adaptively transfer the knowledge from unlabeled information to the target task. Zhou et al. [65] introduce meta-learning optimization into knowledge distillation and transfer the knowledge of the teacher network to the student network. It introduces a pilot update mechanism to improve the alignment between the inner-learner and meta-learner in meta-learning algorithms that focus on an improved inner-learner. TAdaNet [46] proposes a task-adaptive network via meta-learning optimization that makes use of a domain-knowledge graph to enrich data representations and provide task-specific customization for the classification task. In this work, we develop a meta-learning optimization algorithm for the knowledge transfer between the node classification distribution and the unsupervised reconstructed graph, which basically is to optimize both the parameters of the supervised channel and the unsupervised channel.

3 PRELIMINARIES

In this section, we introduce the important definitions and notations of this article. The notations and their description used in this article are presented in Table 1.

3.1 Graph Data

The graph data can be expressed as $G = \{A, X\}$, where $A \in \mathbb{R}^{N \times N}$ is the symmetric adjacency matrix with N nodes, $A_{ij} = 1$ represents that there is an edge between nodes n_i and n_j, otherwise, $A_{ij} = 0$, where $X \in \mathbb{R}^{N \times d}$ is the node feature matrix, and d is the dimension of node features.

3.2 Unsupervised Learning on Graphs

Give the graph data $G = \{A, X\}$. The unsupervised learning on graphs, also named graph representation learning and graph embedding, refers to embedding the original graph topology and node attribute information, i.e., A and X into a low-dimension node representations matrix, represented as $Z \in \mathbb{R}^{N \times h}$, where h is the hidden dimension, each row z_i of Z represents the unsupervised node embedding of the corresponding node n_i.

3.3 Semi-supervised Learning on Graphs

Give the graph data $G = \{A, X\}$. A few nodes in G are labeled, and the rest are unlabeled. Suppose the index of labeled nodes is $\mathcal{I} \in \mathbb{R}^{N_l}$, and the label set of these nodes is $Y \in \mathbb{R}^{N_l \times C}$ on G, where N_l is the number of labeled nodes, and C is the number of classes, the semi-supervised learning on graphs in our article refers to the node classification problem, which aims to assign all unlabeled nodes one out of C classes.

3.4 Bi-level Optimization via Meta-learning

The bi-level optimization problem refers to the optimization on one set of parameters θ contains the optimization on another set of parameters ϕ as a constraint. A typical example is the bi-level optimization on hyperparameters and target parameters. The hyperparameter can be regularization strength [10, 27], task-relatedness in multi-task learning [9, 14], sample weights [43], and so on.

The bi-level optimization is usually addressed by the meta-learning training strategy. Meta-learning typically involves a hierarchical optimization process, which can greatly improve the

Table 1. Table of Notation

Categories	Notation	Description
Given data	G	graph data
	A	adjacency matrix
	X	node feature matrix
	N	number of nodes
	d	dimension of node feature
	N_l	number of labeled nodes
	C	number of classes
	$\mathcal{I}$	index set of labeled nodes
	Y	label set of labeled nodes
Framework	f_θ	supervised model
	g_ϕ	unsupervised model
	θ	parameters of the supervised model
	ϕ	parameters of the unsupervised model
	$\mathcal{L}_{Sup}$	loss of the supervised channel
	$\mathcal{L}_{Uns}$	loss of the unsupervised channel
	$\mathcal{L}_{GKT}$	loss of the knowledge transfer model
Knowledge Transfer	B	batch size
	$\{n_i\}_{i=1}^B$	a batch of nodes
	$\{\widehat{y}_i\}_{i=1}^B$	classification probability distributions of nodes
	$\{z_i^{(u)}\}_{i=1}^B$	unsupervised embeddings of nodes
	$\widehat{A}$	weighted adjacency matrix of the reconstructed batch-graph
	L	Laplacian matrix of the reconstructed batch-graph
	δ	threshold that controls the density of the reconstructed graph
	$\mathcal{I}_B$	index set of labeled nodes in the batch
	Y_B	label set of labeled nodes in the batch
Bi-level optimization	λ_1, λ_2	loss weights
	$\{\mathcal{B}_i^{inner}\}_{i=1}^M$	M batches of nodes for the inner-learner
	$\{\mathcal{B}_i^{meta}\}_{i=1}^M$	M batches of nodes for the meta learner
	$\mathbf{g}_{\theta_t}$	gradient of θ in the tth training step
	$\mathbf{g}_{\phi_t}$	gradient of ϕ in the tth training step

generalization ability of the model via meta-gradient parameter updating. The core idea of meta-learning is *learning to learn*, which aims to obtain an optimized outer-learner, while the optimization of the inner-learner is mainly used to provide learning signals and transfer across-task knowledge for the meta-optimization process [15]. There are many works involving bi-level optimization on two sets of parameters. They usually optimize one set of parameters in the inner-learner and use the optimized parameters in the inner-learner as initialization to optimize another set of parameters in the outer-learner. For example, in the bi-level optimization on hyperparameters and target parameters, the hyperparameters are usually optimized first in the inner-learner, and then the learned theses are used to guide the optimization of the target parameters.

Formally, meta-learning for bi-level optimization consists of an inner-learner and an outer-learner; the update rule of two sets of parameters can be formalized as follows:

$$\begin{aligned} \theta^* &= \arg\min_\theta \sum_{i=1}^{M} \mathcal{L}_{outer}\left(\mathcal{B}_i^{outer}, \theta, \phi^*(\theta)\right) \\ \text{s.t. } \phi^*(\theta) &= \arg\min_\phi \sum_{i=1}^{M} \mathcal{L}_{inner}\left(\mathcal{B}_i^{inner}, \theta, \phi\right), \end{aligned} \tag{1}$$

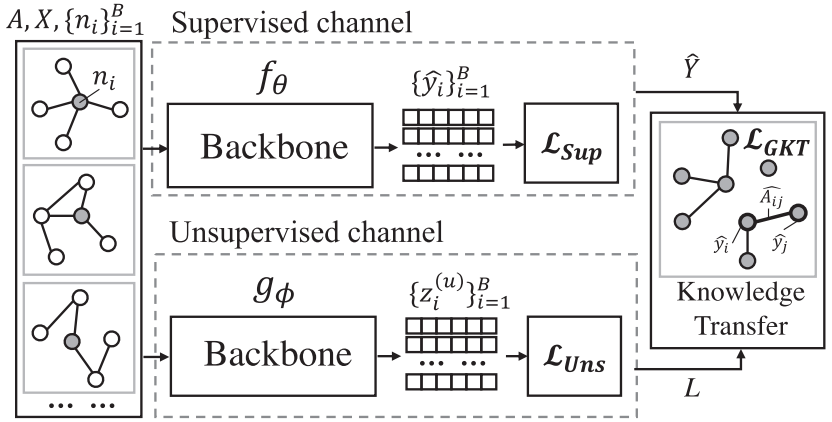


Fig. 2. The dual-channel graph knowledge transfer module. The graph data and batch of nodes are fed into two channels of backbones (which can be a graph/sub-graph encoder) to output classification probability and unsupervised embeddings of nodes. Three losses, including $\mathcal{L}_{Sup}$, $\mathcal{L}_{Uns}$, and $\mathcal{L}_{GKT}$, are conducted on the dual channels to perform knowledge transfer.

where $\mathcal{L}_{outer}$ and $\mathcal{L}_{inner}$ represent the outer and inner objectives and also are the optimization objectives of θ and ϕ, respectively. $\mathcal{B}_i^{outer}$ and $\mathcal{B}_i^{inner}$ represent the training data for the inner-learner and outer-learner. M is the training steps. From 1, ϕ is optimized by the inner-learner, while the θ is not changed in training, and the outer-learner optimizes θ to produce models with $\phi^*(\theta)$ that perform well on the outer objective after training. With such a procedure, one can obtain the optimized outer-learner for the target task, while the optimization of inner-learner aims to provide meta-knowledge to benefit the target task.

4 METHODOLOGY

In this section, we first introduce the proposed knowledge transfer model for integrating unsupervised information into semi-supervised learning via a reconstructed graph, as shown in Figure 2. We also explain that the knowledge transfer objective can adaptively adjust the reconstructed graph by optimizing the unsupervised node embeddings in the guide of classification probabilities. Then, we introduce our meta-learning-based optimizing strategy to address the bi-level optimization problem on the transfer learning above two-channel, as shown in Figure 3. Finally, we introduce the detailed implementations of our model.

4.1 Model Overview

In our article, we aim to train the SSL-G model under the observed label set Y for node classification and also under designed self-supervised signals (pseudo labels) extracted from G for improving generalization. We suppose the distribution of hidden outputs under the true and pseudo labels favors different semantic spaces and should be individually encoded. Thus, we set two channels of models, including a supervised model $f_\theta = f(\theta; A, X)$ and an unsupervised model $g_\phi = g(\phi; A, X)$ on the graph data, where θ and ϕ are the parameters, f_θ is trained with the loss $\mathcal{L}_{Sup}(f_\theta; Y)$ under the observed label set Y, and g_ϕ is trained with the loss $\mathcal{L}_{Uns}(g_\phi; G)$ under the designed self-supervised signals extracted from G. We propose to transfer the generalization ability of the unsupervised channel to the supervised channel to improve the semi-supervised node classification. Thus, we propose a knowledge transfer head with the loss $\mathcal{L}_{GKT}(f_\theta, g_\phi; Y, G)$ to bridge the two models. We optimize the model parameters under the above three losses in an end-to-end meta-learning-based training pipeline.

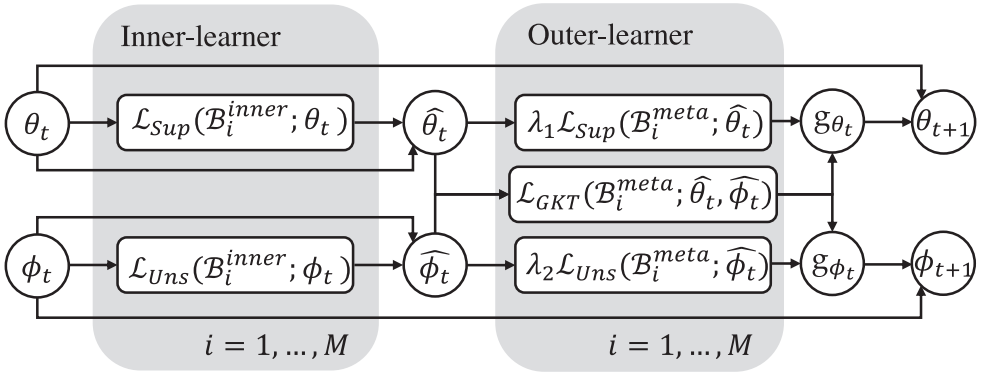


Fig. 3. Illustration of our meta-learning optimization. The parameter gradients are calculated by two stages—inner-learner and outer-learner with different objectives. For each stage, M batches of nodes is sampled for the learners.

4.2 Dual-channel Graph Knowledge Transfer

Inspired by the label propagation models, we utilize unsupervised node embeddings to reconstruct graphs and use the topology of the reconstructed graphs to smooth the classification probabilities of connected nodes. The assumption is that the nodes with higher similarities are more likely to share similar classification probability distributions. The label smoothing can be seen as a regularization of the classification probability output of the supervised model, which avoids the supervised model over-fitting on a few labeled nodes and improves the generalization of the prediction.

In our article, the unsupervised model is also trained in the guidance of the supervised model via the knowledge transfer head; that is, the nodes with higher classification probability similarity are encouraged to have higher connection weights on the reconstructed graph, meaning that they are closer with each other in the unsupervised spaces. It helps to adaptively trim the edges not related to the distribution of labels, enhance the connection of nodes with similar classification probability on the reconstructed graph in training, and eventually further improve the distribution of classification probability in the supervised model for node classification.

Formally, considering the scalability of our model, we reconstruct batch-wise graphs instead of the complete graph. Specifically, given a batch of nodes $\{n_i\}_{i=1}^B$ where B is the batch size, we first obtain the classification probability distributions $\{\widehat{y}_i\}_{i=1}^B$ and the unsupervised node embeddings $\{z_i^{(u)}\}_{i=1}^B$ of the given batch nodes from f_θ and g_ϕ, respectively. Then, we use the unsupervised node embedding to reconstruct a batch-graph, whose weighted adjacency matrix $\widehat{A} \in \mathbb{R}^{B\times B}$ is calculated by:

$$\widehat{A}_{ij} = s\left(z_i^{(u)}, z_j^{(u)}\right) * \mathbb{1}\left(s\left(z_i^{(u)}, z_j^{(u)}\right) > \delta\right), \tag{2}$$

where $\widehat{A}_{ij}$ is the ith row and j-column of $\widehat{A}$, $\mathbb{1}(\cdot) \to \{0, 1\}$ is the indicator function, and $s(z_i^{(u)}, z_j^{(u)})$ can be any kernel similarity function, we use the cosine function for simplicity. δ is the threshold to control the density of the reconstructed graph. Then, the loss of knowledge transfer is defined as:

$$\begin{aligned} \mathcal{L}_{GKT} &= \sum_{i,j=1}^{B} \widehat{A}_{ij} ||\widehat{y}_i - \widehat{y}_j||_2^2, \\ &\text{s.t. } \widehat{y}_i = y_i, \end{aligned} \tag{3}$$

where $\widehat{y}_i \in \mathbb{R}^C$, representing the classification probability distribution of node n_i output by the supervised model, we replace the soft classification probability $\widehat{y}_i$ with the hard label y_i if n_i is labeled. Suppose $\widehat{Y} = \{\widehat{y}_i\}_{i=1}^B$ is the probability distribution matrix; we can rewrite the above function with matrix notation:

$$\begin{aligned}\sum_{i,j=1}^{B} \widehat{A}_{ij}||\widehat{y}_i - \widehat{y}_j||_2^2 &= \sum_{i=1}^{B}\sum_{j=1}^{B} \widehat{A}_{ij}\left(\widehat{y}_i^2 + \widehat{y}_j^2 - 2\widehat{y}_i\widehat{y}_j\right) \\ &= 2\sum_{i=1}^{B}\widehat{y}_i^2 \sum_{j=1}^{B}\widehat{A}_{ij} - 2\sum_{i,j=1}^{B}\widehat{A}_{ij}\widehat{y}_i\widehat{y}_j \\ &= 2(\widehat{Y}^T D\widehat{Y} - 2\widehat{Y}^T\widehat{A}\widehat{Y}) \\ &= 2\widehat{Y}^T(D-\widehat{A})\widehat{Y} \\ &= 2\widehat{Y}^T L\widehat{Y},\end{aligned} \tag{4}$$

where L is the Laplacian of the reconstructed batch-graph, which is defined as $L = D - \widehat{A}$ and $D_{ii} = \sum_{j=i}^{B}\widehat{A}_{ij}$ is a diagonal matrix of $\widehat{A}$. Then, we remove the constant in Equation (3) and rewrite it as follows:

$$\begin{aligned}&\mathcal{L}_{GKT} = \widehat{Y}^T L\widehat{Y} \\ &\text{s.t. } \widehat{Y}[\mathcal{I}_B] = Y_B,\end{aligned} \tag{5}$$

where $\mathcal{I}_B$ is the index set of labeled nodes in the batch, and $Y_B \subseteq Y$ is the corresponding label set. As the equation shows, the knowledge transfer is very efficient, as it is matrix operations on the batch-wise outputs of two channels, and it introduces no extra parameters.

In training, both the supervised channel f_θ and the unsupervised channel g_ϕ are trained via $\mathcal{L}_{GKT}$. According to the above equations, we can easily reach that, for f_θ, nodes with higher unsupervised similarities are encouraged to have less mean absolute error on classification probabilities. Thus, the generalization ability is incorporated into the supervised channel via such a label-smoothing process. Meanwhile, for g_ϕ, nodes with a higher difference in classification probabilities would be led to have fewer similarities in unsupervised space. Thus, fitting capability for label information is possessed in the unsupervised channel. Eventually, through the above, the knowledge transfer between the two channels is realized.

4.3 Bi-level Optimization via Meta-learning

As discussed in the last section, training the generalized classification probability of nodes depends on the optimized unsupervised node embeddings, while training unsupervised node embeddings to adaptively adjust the reconstructed graph requires the guidance of optimized classification probability. Thus, the above objective becomes a typical bi-level optimization problem, where one optimization contains another optimization as a constraint [16]. The learning objective is formulated as follows:

$$\begin{aligned}&\theta^*(\phi) = \arg\min_{\theta}(\lambda_1\mathcal{L}_{Sup}(f_\theta; Y) + \mathcal{L}_{GKT}(f_\theta, g_{\phi^*(\theta)}; Y, G)) \\ &\text{s.t. } \phi^*(\theta) = \arg\min_{\phi}(\lambda_2\mathcal{L}_{Uns}(g_\phi; G) + \mathcal{L}_{GKT}(f_{\theta^*(\phi)}, g_\phi; Y, G)),\end{aligned} \tag{6}$$

where θ^* and ϕ^* represent the optimized parameters and λ_1 and λ_2 are the loss weights. Simply training the above three losses in a multi-task manner may cause a negative transfer. For example, the over-fitting at a bad local minimum on the supervised channel may dampen the node embeddings on the unsupervised channel.

Thus, we design a meta-learning objective to solve the bi-level optimization problem on two channels of parameters in Equation (6). The strategy is that, for each channel, we first train one model in the inner-learner and then transfer the learned knowledge of the model to train the other model in the outer-learner. Thus, there are two stages in each training step. In the inner-learning stage, we first optimize the two channels of the model under their own losses with a few training steps individually. Then, in the meta-learning stage, the properly optimized parameters are trained to obtain the final gradient of parameters under their own losses as well as the knowledge transfer loss. Finally, we use the learned gradient to upgrade the parameters of the two channels.

Specifically, in the inner-learning stage of the tth training step, suppose θ_t and ϕ_t are the parameters of two channels, we sample M batches of nodes, expressed as $\{\mathcal{B}_i^{inner}\}_{i=1}^{M}$ where $\mathcal{B}_i^{inner} = \{n_j\}_{j=1}^{B}$, and separately train the two channels of parameters, which is updated as follows:

$$\begin{aligned} \widehat{\theta}_t &= \arg\min_{\theta} \sum_{i=1}^{M} \mathcal{L}_{Sup}\left(\mathcal{B}_i^{inner}; \theta_t\right) \\ \widehat{\phi}_t &= \arg\min_{\phi} \sum_{i=1}^{M} \mathcal{L}_{Uns}\left(\mathcal{B}_i^{inner}; \phi_t\right). \end{aligned} \tag{7}$$

Then, in the outer-learner stage, We sample M additional batches of nodes, denoted as $\{\mathcal{B}_i^{meta}\}_{i=1}^{M}$, and input each batch into the trained model parameters. The final meta-gradients of the parameters are computed by:

$$\begin{aligned} \mathbf{g}_{\theta_t} &= \sum_{i=1}^{M} \nabla_{\theta}\left(\lambda_1 \mathcal{L}_{Sup}\left(\mathcal{B}_i^{meta}; \widehat{\theta}_t\right) + \mathcal{L}_{GKT}\left(\mathcal{B}_i^{meta}; \widehat{\theta}_t, \widehat{\phi}_t\right)\right) \\ \mathbf{g}_{\phi_t} &= \sum_{i=1}^{M} \nabla_{\phi}\left(\lambda_2 \mathcal{L}_{Uns}\left(\mathcal{B}_i^{meta}; \widehat{\phi}_t\right) + \mathcal{L}_{GKT}\left(\mathcal{B}_i^{meta}; \widehat{\theta}_t, \widehat{\phi}_t\right)\right). \end{aligned} \tag{8}$$

Finally, the updating parameters θ_{t+1} and ϕ_{t+1} are given as $\theta_{t+1} \leftarrow \theta_t - \alpha \mathbf{g}_{\theta_t}$ and $\phi_{t+1} \leftarrow \phi_t - \beta \mathbf{g}_{\phi_t}$, where α and β are the learning rates.

4.3.1 Connection with Existing Work. Actually, our meta-learning optimization can be seen as a variant of MAML [8] dividedly conducted on two channels. The difference is that we introduce an extra loss $\mathcal{L}_{GKT}$ in the task of outer-learner for knowledge transfer. The meta-learning, which involves an inner-learner and an outer-leaner, aims to train the supervised channel to generate node embeddings that are both label-related and correlated with the node embeddings produced by the unsupervised channel. Similarly, the same holds for the unsupervised channel. Thus, this training strategy can avoid one of the tasks dominating training and transfer the knowledge learned and optimized from two channels to each other.

4.3.2 Difference with Multi-task Learning. Both multi-task learning and meta-learning aim to leverage the correlation between different tasks to enable better generalization to multiple tasks. The difference is that multi-task learning involves several related tasks jointly to improve the generalization of the model. However, in addition to the joint training scheme, meta-learning improves generalization by training the model's parameters learned by one task to fast adapt to another task.

In our framework, the problem of multi-task learning is that in the joint training schema, different tasks may have different scales of losses and need different convergence times. For our task, the supervised model may converge faster than the unsupervised model. However, due to the limited labeled data, the supervised model may easily over-fitting in the early epoch, making the knowledge transfer model tend to transfer negative knowledge to the unsupervised model, and conversely, the dampened unsupervised model would further negatively affect the supervised

ALGORITHM 1: The tth Training Step of GKT

Input: The graph data $G = \{A, X\}$;
The label set of these node Y;
The index of labeled nodes $\mathcal{I}$;
The parameters of supervised channel θ_t;
The parameters of unsupervised channel ϕ_t.
Output: Updated parameters θ_{t+1} and ϕ_{t+1}

$\widehat{\mathbf{g}_{\theta_t}} = 0, \mathbf{g}_{\theta_t} = 0$;
$\widehat{\mathbf{g}_{\phi_t}} = 0, \mathbf{g}_{\phi_t} = 0$;
/* Inner-Learner */
for $b = 1, \ldots, M$ **do**
  Sample a batch of node $\{n_i\}_{i=1}^{B}$;
  Obtain the classification probability and unsupervised node embeddings $\{\widehat{y}_i\}_{i=1}^{B}$ and $\{z_i^{(u)}\}_{i=1}^{B}$ by f_{θ_t} and g_{ϕ_t};
  $\widehat{\mathbf{g}_{\theta_t}} += \nabla_\theta \mathcal{L}_{Sup}(\{\widehat{y}_i\}_{i=1}^{B}; \theta_t)$;
  $\widehat{\mathbf{g}_{\phi_t}} += \nabla_\phi \mathcal{L}_{Uns}(\{z_i^{(u)}\}_{i=1}^{B}; \phi_t)$;
end
$\widehat{\theta}_t \leftarrow \theta_t - \alpha \widehat{\mathbf{g}_{\theta_t}}$;
$\widehat{\phi}_t \leftarrow \phi_t - \beta \widehat{\mathbf{g}_{\phi_t}}$;
/* outer-learner */
for $b = 1, \ldots, M$ **do**
  Sample a batch of node $\{n_i\}_{i=1}^{B}$;
  Obtain the classification probability and unsupervised node embeddings $\{\widehat{y}_i\}_{i=1}^{B}$ and $\{z_i^{(u)}\}_{i=1}^{B}$ by $f_{\widehat{\theta}_t}$ and $g_{\widehat{\phi}_t}$;
  $\mathbf{g}_{\theta_t} += \lambda_1 \nabla_\theta \mathcal{L}_{Sup}(\{\widehat{y}_i\}_{i=1}^{B}; \widehat{\theta}_t) + \nabla_\theta \mathcal{L}_{GKT}(\{\widehat{y}_i, z_i^{(u)}\}_{i=1}^{B}; \widehat{\theta}_t, \widehat{\phi}_t)$;
  $\mathbf{g}_{\phi_t} += \lambda_2 \nabla_\phi \mathcal{L}_{Uns}(\{z_i^{(u)}\}_{i=1}^{B}; \widehat{\phi}_t) + \nabla_\phi \mathcal{L}_{GKT}(\{\widehat{y}_i, z_i^{(u)}\}_{i=1}^{B}; \widehat{\theta}_t, \widehat{\phi}_t)$;
end
$\theta_{t+1} \leftarrow \theta_t - \alpha \mathbf{g}_{\theta_t}$;
$\phi_{t+1} \leftarrow \phi_t - \beta \mathbf{g}_{\phi_t}$;
Return θ_{t+1} and ϕ_{t+1}

model. To conquer this, we can reduce the weight of the supervised model, making the convergence speed of the supervised model comparable with the unsupervised model. But the weight proportion is unknown to us and is very difficult to determine. Nevertheless, our solution—meta-learning can well solve the negative transfer problem with an adaptive optimization strategy. We first train one model in the inner-learner and then transfer the learned knowledge of the model to train the other model in the outer-learner. The meta-learning makes supervised model to generate node embeddings that are both label-related and correlated with the node embeddings produced by the unsupervised channel. Similarly, the same holds for the unsupervised model. Thus, this training strategy can avoid one of the tasks dominating training and transfer the knowledge learned and optimized from two channels to each other. Thus, meta-learning is more effective than multi-task learning in solving bi-level optimization problems. The pseudocode of our model can be found in Algorithm 1.

4.4 Implementation of Dual-channel Models

In the above sections, we introduce our proposed GKT framework. Note that the supervised channel f_θ and the unsupervised model g_ϕ of the GKT framework are feasible with most supervised

models and unsupervised models of graph data. Without loss of generality, we use popular techniques to implement our framework. The detailed description is as follows:

4.4.1 Backbone Model. In our model, for graph data extracting, we conveniently adopt the most classic and widely used GNN–**Graph Convolutional Network (GCN)** [20] as the backbone of the graph encoder in two channels. We employ a two-layer GCN for each channel, which is expressed as

$$Z = \overline{A}\sigma(\overline{A}XW^{(0)})W^{(1)}, \tag{9}$$

where Z is the node embedding matrix, $\overline{A} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$, $\widetilde{A} = A + I_N$, and $\widetilde{D} = \sum_j \widetilde{A}_{ij}$. $\sigma(\cdot)$ denotes the ReLU function. $W^{(0)}, W^{(1)}$ denote the trainable weight matrices.

Then, we add a supervised head on one channel to compose the supervised channel model f_θ. For unsupervised model g_ϕ, we use the unsupervised head based on deep graph infomax model from DGI [53]. The detailed setting of supervised channel loss and unsupervised loss in GKD is described as follows:

4.4.2 Supervised Channel Loss. Given the input node batch $\{n_i\}_{i=1}^B$, we can obtain the node embeddings $\{z_i^{(s)}\}_{i=1}^B$ of the supervised channel. For training the supervised channel, We first feed $\{z_i^{(s)}\}_{i=1}^B$ into a linear transformation $Linear(\cdot)$ and softmax functions $softmax(\cdot)$ to obtain the classification probability of each node, written as:

$$\widehat{Y} = \{\widehat{y}_i\}_{i=1}^B = softmax\left(Linear\left(\left\{z_i^{(s)}\right\}_{i=1}^B\right)\right). \tag{10}$$

The supervised channel loss $\mathcal{L}_{Sup}$ on the batch is defined as the cross-entropy between $\widehat{y}_i$ and the ground-truth label:

$$\mathcal{L}_{Sup} = \sum_{i=1}^{B} \mathbb{1}(i \in \mathcal{I}_B) H(y_i, \widehat{y}_i), \tag{11}$$

where $\mathcal{I}_B$ is the index set of labeled nodes in the batch, $H(y, p)$ denotes the cross-entropy between the probability distribution p, and one-hot label y, $y_i \in Y$ is the ground truth label of n_i.

4.4.3 Unsupervised Channel Loss. DGI [53] proposed the **Mutual Information Maximization(MIM)**-based unsupervised objective for graph embedding. It considers the strong correlation between the central node and all its neighbor nodes in the local neighborhood and aims to maximize the mutual information between their embeddings.

In our model, given the node batch $\{n_i\}_{i=1}^B$, suppose $\{z_i^{(u)}\}_{i=1}^B$ is the unsupervised node embeddings and $\{u_i^{(u)}\}_{i=1}^B$ is the corresponding graph-level embeddings of the nodes' local neighborhood. Then, the unsupervised loss can be written as:

$$\mathcal{L}_{Uns} = \sum_{i=1}^{B}\left(\log \mathcal{D}(z_i^{(u)}, u_i^{(u)}) + \log(1 - \mathcal{D}(\widehat{z_i^{(u)}}, u_i^{(u)}))\right), \tag{12}$$

where $\mathcal{D}(z, u) : \mathbb{R}^h \times \mathbb{R}^h \to \mathbb{R}$ is a discriminator; we follow the DGI by applying a simple bilinear scoring function:

$$\mathcal{D}(z, u) = \sigma(z^T W^M u), \tag{13}$$

where $\sigma(\cdot)$ is the logistic sigmoid nonlinearity and W^M is trainable weight matrices. To obtain the neighborhood embeddings, we use the personalized pagerank algorithm-based sampling strategy [19, 62] to sample a batch of neighborhood sub-graphs $\{G_i\}_{i=1}^B$ for the given node batch $\{n_i\}_{i=1}^B$,

then summarize the unsupervised node embeddings on each sub-graph into a graph-level embedding via a *Readout function*:

$$u_i^{(u)} = \mathcal{R}\left(\left\{z_j^{(u)}\right\}_{n_j \in G_i}, G_i\right). \tag{14}$$

We defined $\mathcal{R}(\cdot)$ as a self-attention graph pooling operation similar with Reference [21]. Then, the negative embeddings $\widehat{z_i^{(u)}}$ for contrasitve learning is obtained by a *corruption function* defined as:

$$\left\{\widehat{z_i^{(u)}}\right\}_{i=1}^{B} = C\left(\left\{z_i^{(u)}\right\}_{i=1}^{B}\right). \tag{15}$$

For efficiency, we define the corrupting function $C(\cdot)$ as a randomly shuffling operation on the batch-wise inputs, based on the assumption that neighbor nodes see its central node as the positive sample and the central nodes in other sub-graphs as negative samples. The advantage is that there is no need to sample new negative samples or an extra generation process to increase additional calculations.

4.5 Complexity Analysis

To analyze the complexity of GKT, we can first divide the computing of GKT into three parts—the two channels of supervised and unsupervised models, a knowledge transfer head, and the meta-learning optimization strategy. Below, we prove that the running time of GKT mainly depends on the time complexity of the backbone model, while the computing time of its other modules is negligible.

First, the running time of supervised and unsupervised models depends on the chosen backbones. Second, the complexity of knowledge transferring is consisted of computing L and Equation (5), the time complexity of computing L is $O(BBf)$, which depends on computing the cosine similarity matrix of B samples in the input batch, and f is the hidden dimension. The time complexity of Equation (5) is $O(BBB)$. Thus, the complexity of the knowledge transfer module is $O(BBf + BBB)$. We choose the most widely used model, GCN, as the backbone to compare with the knowledge transfer module. According to Reference [6], suppose N is the number of nodes and d is the average node degree in the graph, the time complexity of a L-layer GCN on the complete graph can be bounded by $O(LNdf + LNff)$, where $O(LNdf)$ is the cost of feature propagation, i.e., the sparse-dense matrix multiplication on the normalized adjacency matrix and the node hidden embedding matrix, and $O(LNff)$ is the cost of the feature transformation by applying weight matrix. As usually $B < f$, $B << N$, and B is comparable with Ld, $O(LNff) >> O(BBf)$ and $O(LNdf) >> O(BBB)$. Thus, the time complexity of the knowledge transfer module is far less than that of the GCN backbone, which proves that it is very efficient. Third, for the meta-learning optimization, we do not add any extra parameters other than two channels of models, and these two channels are individual and can be computed in parallel. Thus, the computing cost of back-propagation is equal to that of the backbone. Thus, meta-learning barely increases computing. However, according to the training curves reported in Section 5.7, GKT convergences in about the 20th epochs on the Cora and Citeseer datasets and in about the 50th epochs on the Pubmed dataset. As reported in Reference [38], our model achieves convergence with a similar number of epochs with baselines, proving that meta-learning brings no training gain.

To sum up, the running time of GKT mainly depends on the backbone model and barely depends on the knowledge transfer module and meta-learning optimization strategy. In our experiments, the actual running time of GKT on five datasets is less than 10% higher than that of baseline models. The training efficiency on large data sets may not be the primary focus of this paper. Still, we totally consider it an important factor so our methods can achieve better performance and better robustness at the expense of very limited extra computations.

Table 2. The Statistics of the Datasets

Dataset	Nodes	Edges	Classes	Attribute	Label rate
Cora	2,708	5,429	7	1,433	0.052
Citeseer	3,327	4,732	6	3,703	0.036
Pubmed	19,717	44,338	3	500	0.003
Flickr	89,250	899,756	7	500	0.50
Reddit	232,965	11,606,919	41	602	0.66

5 EXPERIMENTS

In this section, we empirically evaluate the effectiveness of the proposed GKT on semi-supervised learning on real-world graph datasets and answer the following **research questions** (**RQ**):

- **RQ1:** Is the performance of GKT superior to existing unsupervised and supervised baselines on the semi-supervised node classification task?
- **RQ2:** Does the proposed technology, such as dual-channel backbone, knowledge transfer, and meta-learning of GKT, have the claimed positive effects on the semi-supervised learning?
- **RQ3:** How is the distribution of node embeddings learned by GKT compared with the node embedding learned by its two channels of models?
- **RQ4:** Can the ability of GKT to balance the generalization and fitting capability via the reconstructing and label-smoothing-based knowledge transfer be proved?
- **RQ5:** What is the effect of choosing different backbones, batch sizes, and sub-graph sizes on the model?

5.1 Datasets

We use five benchmark datasets of semi-supervised node classification, including three citation networks named Cora, Citeseer, and Pubmed [40] for classifying research papers into topics, a social network named Flickr [60] for categorizing types of images based on the descriptions and common properties of online images, and a social network named Reddit [13] for predicting communities of online posts based on user comments. There are significant variations in the number of nodes in these datasets, which can well reflect the scalability of our model. All datasets are exported from the torch geometric library[1] and follow the "fixed-partition" splits of train/valid/test set in the original papers. The statistics of these datasets are presented in Table 2.

5.2 Baselines

We compare our method with both the supervised and unsupervised methods. We choose five unsupervised methods based on different self-supervised objectives: DeepWalk [33] is based on skip-gram, EP-B [11] is based on margin-based ranking loss, GraphSAGE(Uns) [13] in unsupervised version is a GNN model encoder with a skip-gram-based loss, DGI [53] and GMI [32] is based on mutual information maximization. In detail,

- **DeepWalk** is a classic graph embedding method that randomly initializes the node embeddings and uses a random walk strategy to generate paths and leverage a skip-gram loss on these paths to optimize the node embeddings.
- **EP-B** is an unsupervised learning method for graphs, which learns node representations by passing two types of messages between neighboring nodes.

[1] https://pytorch-geometric.readthedocs.io.

Table 3. The Hyper-parameter Settings on Five Datasets

Parameter	Cora	Citeseer	Pubmed	Flickr	Reddit
batch size B	64	64	512	64	64
number of batches M	8	16	16	2	2
sub-graph size k	20	8	20	20	20
density threshold δ	0.75	0.85	0.85	0.85	0.85
learning rate α	1e-4	1e-4	1e-3	1e-3	1e-4
learning rate β	1e-4	3e-4	5e-4	5e-4	1e-3

- **GraphSAGE(Uns)** is the unsupervised version of the graph neural network–GraphSAGE, which encodes nodes with its subgraph neighborhood into node embeddings and uses a skip-gram loss function to optimize the node embeddings.
- **DGI** is a deep graph representation learning method that leverages local mutual information maximization between the graph's patch representation and node representations to capture structural properties.
- **GMI** inherits the idea of mutual information maximization on the graph by directly maximizing the mutual information between the input and output of a graph neural encoder in terms of node features and topological structure.

We also choose six supervised models with the different supervised settings, including **Label Propagation (LP)** [68], Planetoid [58], and four state-of-the-art GNN models that encode the complete unlabeled graph data to perform semi-supervised classification: GCN [20], GAT [52], FastGCN [4], and SGC [56]. In detail,

- **LP** applies a regularization term based on the unlabelled graph topology that encourages a node's predicted label probability distribution to be equal to a weighted average of its neighbors' distributions.
- **Planetoid** performs semi-supervised learning by applying a supervised loss and a regularization term that depends on the unsupervised skip-gram representation of the graph.
- **GCN** performs semi-supervised node classification on node embeddings incorporated with multi-level neighborhood information. It implements the graph convolution in each layer by conducting Laplacian smoothing.
- **GAT** improves GCN by leveraging the attention mechanism on each layer to calculate the edge weights for neighbor aggregation.
- **FastGCN** enhances GCN with importance sampling, which subsamples vertices in a bootstrapping manner in each layer to approximate the convolution.
- **SGC** improves GCN by removing the weight matrices and nonlinearities between layers via a graph-based pre-processing step.

5.3 Experimental Setting

We adopt the mean classification accuracy as the metric on five datasets. All experiments are conducted on an Nvidia Tesla V100 GPU (32 GB GPU Memory). We implemented GKT in Pytorch 1.7.1 with Python 3.7. We set the loss weights λ_1 and λ_2 both as 1, the weight decay as 1e-3, and the dropout rate as 0.9. Other hyper-parameter settings of GKT on five datasets are presented in Table 3.

5.4 Node Classification (RQ1)

We report the mean and confidence interval of the classification results of our method in Table 4, which are measured by three runs under the same hyperparameters. Notably, we take the best

Table 4. Results of Node Classification

Category	Methods	Training data A	X	Y	Cora	Citeseer	Pubmed	Flickr	Reddit
Unsupervised	DeepWalk	✓			67.2%	43.2%	65.3%	27.9%	32.4%
	EP-B	✓	✓		78.1 ± 1.5%	71.0 ± 1.4%	79.6 ± 2.1%	-	-
	GraphSAGE	✓	✓		75.2 ± 1.5%	59.4 ± 0.9%	70.1 ± 1.4%	36.5 ± 1.0%	90.8 ± 1.1%
	DGI	✓	✓		82.3 ± 0.6%	71.8 ± 0.7%	76.8 ± 0.6%	42.9 ± 0.1%	94.0 ± 0.1%
	GMI	✓	✓		83.0 ± 0.3%	72.4 ± 0.1%	79.9 ± 0.2%	44.5 ± 0.2%	95.0 ± 0.0%
Supervised	LP	✓		✓	68.8%	43.9%	66.4%	-	-
	Planetoid	✓	✓	✓	75.7%	64.7%	77.2%	-	-
	GCN	✓	✓	✓	81.4 ± 0.6%	70.3 ± 0.7%	76.8 ± 0.6%	48.7 ± 0.3%	93.3 ± 0.1%
	GAT	✓	✓	✓	83.0 ± 0.7%	72.5 ± 0.7%	79.0 ± 0.3%	OOM	OOM
	FastGCN	✓	✓	✓	78.0 ± 2.1%	63.5 ± 1.8%	74.4 ± 0.8%	48.1 ± 0.5%	89.5 ± 1.2%
	SGC	✓	✓	✓	81.0 ± 0.0%	71.9 ± 0.1%	78.9 ± 0.0%	–	94.9%
Dual-channel	GKT	✓	✓	✓	**84.4 ± 0.3%**	**74.0 ± 0.4%**	**80.8 ± 0.6%**	**50.2 ± 0.5%**	**95.5 ± 0.2%**

OOM: out of memory.

results of baselines that are already reported in their papers and other existing papers [2, 19, 23, 60]. Clearly, GKT generally achieves the best performance on all datasets, showing the superiority of our method in improving the semi-supervised node classification. We can observe that the unsupervised methods achieve competitive results with supervised methods and, in some cases, are better than them; that is because when there are limited labels, the generalization ability of the supervision methods is weaker than unsupervised methods, and it is easier to overfit. But the unsupervised methods are limited without introducing labels in training, and their learned node embeddings are independent of downstream classification tasks, which restricts their performance. Although most supervised baselines incorporate unsupervised information in training, they are unable to automatically select useful unsupervised information and may introduce label-unrelated ones to dampen the fitting ability. However, our model can adaptively adjust the reconstructed graph to label-smoothing for classification, which well balances the generalization and fitting ability. Plus, the meta-learning strategy optimization of the two channels, separately, making one channel's optimization would not dominate the other channel's optimization. Thus, the proposed GKT model can achieve better results than baselines.

5.5 Ablation Study (RQ2)

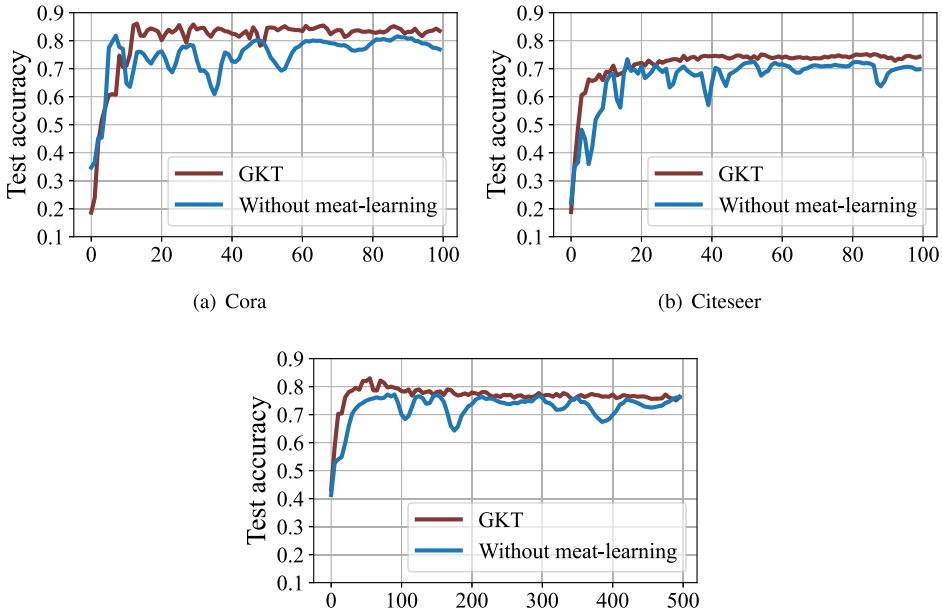
To verify the properties and effectiveness of the knowledge transfer on dual channels and the meta-learning optimization, we design four variant models to conduct an ablation study on GKT:

- **1C&U.** We only train the unsupervised channel with the unsupervised loss and input the learned node embeddings into a logistic regression classifier to give the prediction results of test nodes.
- **1C&S.** We only train the supervised channel with the supervised loss and use the learned model to generate the classification probability of nodes for prediction.
- **1C&SU.** We only set one channel with the graph encoder same as GKT and conduct both the supervised and unsupervised heads above. Thus, there is no need to transfer knowledge, and we sum the two losses for optimization.
- **2C&KT.** We use the same model architecture and loss function as GKT but train the model in a multi-task manner without meta-learning. We directly sum up the three losses of GKT.

We report the classification results of the variant models and GKT in Table 5. We can observe that, as shown in the results of 1C&U, 1C&S, and 1C&SU, simply combining the supervised loss and unsupervised loss to train the graph model with shared parameters may be unable to improve the

Table 5. Results of Ablation Study

Models	Brief Description	Cora	Citeseer	Pubmed	Flickr	Reddit
1C&U	One channel with unsupervised head	79.9%	67.9%	75.3%	47.9%	93.1%
1C&S	One channel with supervised head	80.2%	70.3%	75.1%	41.0%	93.3%
1C&SU	One channel with the both above heads	80.0%	73.5%	77.1%	41.2%	93.3%
2C&KT	Two channels with knowledge transfer	80.6%	73.6%	78.9%	45.0%	94.7%
GKT	–	**84.4%**	**74.0%**	**80.8%**	**50.2%**	**95.5%**



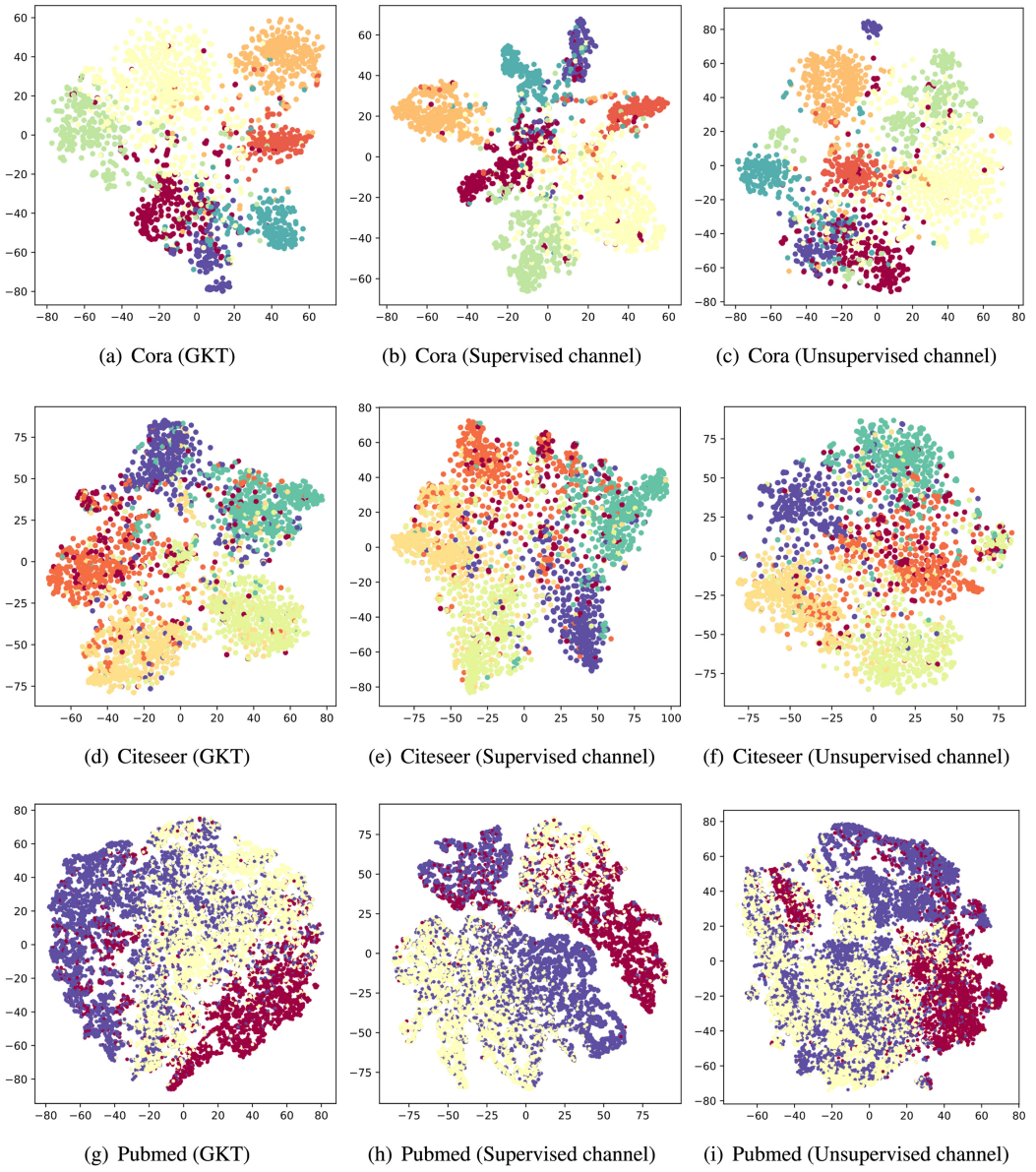
(a) Cora (b) Citeseer

(c) Pubmed

Fig. 4. Test accuracy curves of GKT and GKT without meta-learning in training.

performance, as different optimization objectives favor different semantic spaces and may dampen with each other in training. GKT and 2C&KT lead a clear margin in comparison with other models, and 2C&KT outperforms 1C&SU, showing the effectiveness of our proposed knowledge transfer model in integrating the generalization of unsupervised information for node classification. While, plus the meta-learning, GKT performs much better 2C&KT, verifying the indispensability of meta-learning-based optimization in improving the framework.

To further verify the effectiveness of meta-learning in training, we also report test accuracy curves of GKT and GKT without meta-learning in training in Figure 4, where the x-axis is the number of epochs. We report the results of the first 100 batches for Cora and Citeseer and 500 batches for Pubmed. We can observe that the two methods begin to converge after 20 batches on Cora and Citeseer and after 100 batches on Pubmed. In contrast, GKT performs better and is more stable with meta-learning-based optimization; that is because meta-learning can avoid one of the tasks dominating training and the negative transfer, thus further improving the generalization ability of the model. In short, the ablation study proves the effectiveness of the dual-channel knowledge transfer and meta-learning of GKT.



(a) Cora (GKT) (b) Cora (Supervised channel) (c) Cora (Unsupervised channel)

(d) Citeseer (GKT) (e) Citeseer (Supervised channel) (f) Citeseer (Unsupervised channel)

(g) Pubmed (GKT) (h) Pubmed (Supervised channel) (i) Pubmed (Unsupervised channel)

Fig. 5. The t-SNE visualization of node embeddings learned by GKT, its supervised channel, and its unsupervised channel.

5.6 Visualization of Node Embeddings (RQ3)

To evaluate the effectiveness of the proposed method in improving the distributions of node embeddings, we project the node embeddings learned by GKT, the node embeddings learned only by its supervised channel, i.e., GCN, and its unsupervised channel, i.e., DGI, into the two-dimension space via the t-SNE [51] technique and visualize them in Figure 5. We can observe that the node embeddings learned by both GKT and the supervised have clearer margins between different classes than the unsupervised channel; that is because they are incorporated with label information in

training. In contrast, the unsupervised channel can only learn label-agnostic node embeddings. Also, we can observe that the area of each class in the supervised channel is smaller than that in GKT, and the nodes with different classes more easily overlap with each other, while the node embeddings learned by the unsupervised channel have more discrete distribution and the boundaries between different classes are ambiguous. These phenomena are because the supervised channel is easily over-fitting, and the unsupervised channel can learn generalized node embeddings but not the label information. The visualization results of GKT show that the node embeddings learned by GKT are better than these two channels in both the generalization and the label-fitting; that is because the proposed GKT can adaptively balance the fitting ability of the supervised channel and the generalization ability of the unsupervised channel in training so that it can learn better node embeddings and achieve better performance in node classification.

5.7 Adaptive Ability Analysis of Reconstructed Graph (RQ4)

We have introduced in Section 4 that we reconstruct a bath-graph via the unsupervised node embeddings to help the knowledge transfer between the supervised channel and the unsupervised channel. Also, the reconstructed graph can be adaptively adjusted in training with the guide of supervised information so nodes with similar classification probability would be suggested to be connected in the reconstructed graph. In this section, to evaluate the adaptive adjusting ability of the reconstructed graph, we calculate the *Homophily* [66] of the reconstructed graph during training. Homophily follows the principle of "birds of a feather flock together," which can reflect the correlation between the topology of the graph and node category. Homophily is often calculated as the fraction of intra-class edges in a graph. Formally, it can be defined as follows:

Definition 1 (Homophily). The homophily of a graph $G = \{A, X\}$ with node label vector Y is given by

$$h(G, Y) = \frac{1}{\sum_{i,j \in N} A_{ij}} \sum_{i,j \in N} A_{ij} * \mathbb{1}(y_i == y_j), \tag{16}$$

where A_{ij} is the ith row and jth column of A, and $y_i \in Y$ indicates the label of the ith node.

We calculate the homophily for the reconstructed graphs on whole nodes given all labels in each training step and report the changing curves in Figure 6. We can observe that the values of homophily on three datasets all increase with the training steps, indicating that the topology of the reconstructed graph becomes more correlated with the node labels in training. This proves that our model can adaptively adjust the reconstructed graph to help the label smoothing for the classification. The reason why our model has such ability is that through knowledge transfer and meta-learning, it can transfer the supervised information to the training of node embeddings in the unsupervised channel, which makes the embeddings of nodes with high similarity close and increases the label-correlated edges in the reconstructed graph. Thus, the label-smoothing operation on the reconstructed graph not only improves the generalization ability of the model but also avoids introducing too much unrelated unsupervised information to affect the fitting ability for the classification.

5.8 Comparison of Various Backbones (RQ5)

The most fundamental module in our framework is the graph encoder, which is to extract the graph data for generating the classification probability of nodes in the supervised model and the node embeddings in the unsupervised model. Considering that the graph neural networks have been verified as the most effective graph extractors, we choose four widely used graph neural networks as the sub-graph encoder to substitute the GCN in our framework, including GAT [52], GraphSAGE [13], SGC [56], GIN [57]. The results are shown in Figure 7. We can observe that (1) their

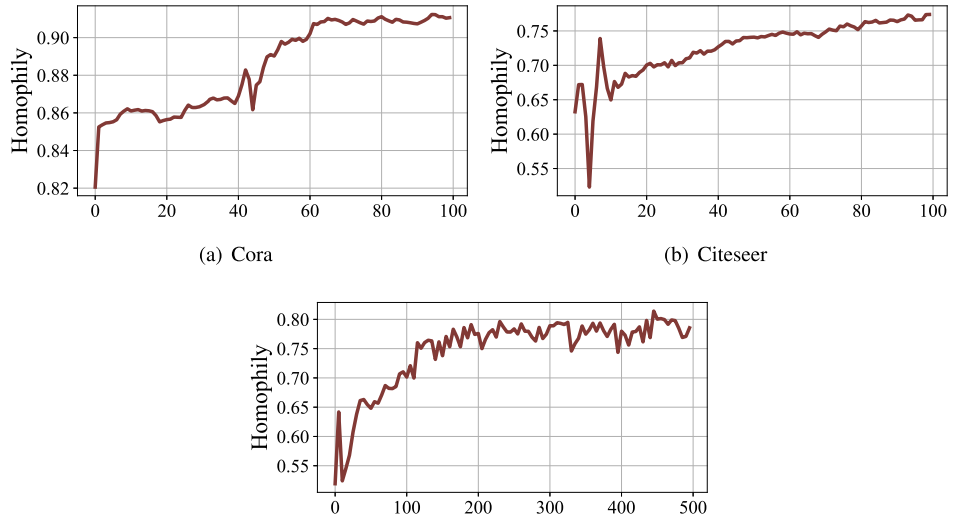


Fig. 6. Homophily curves of the reconstructed graph in training.

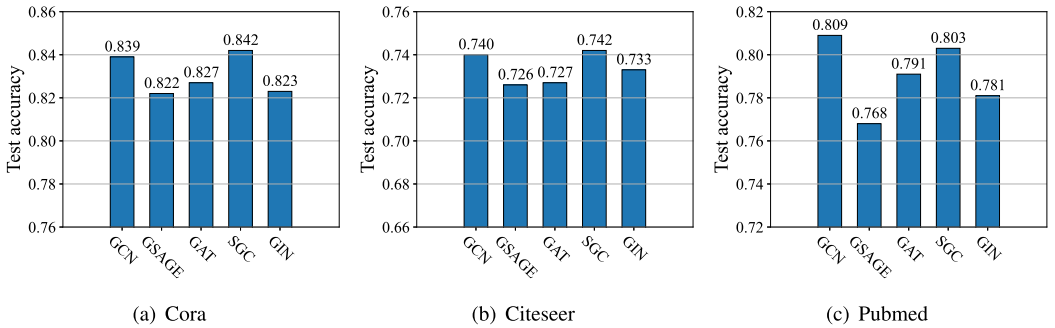
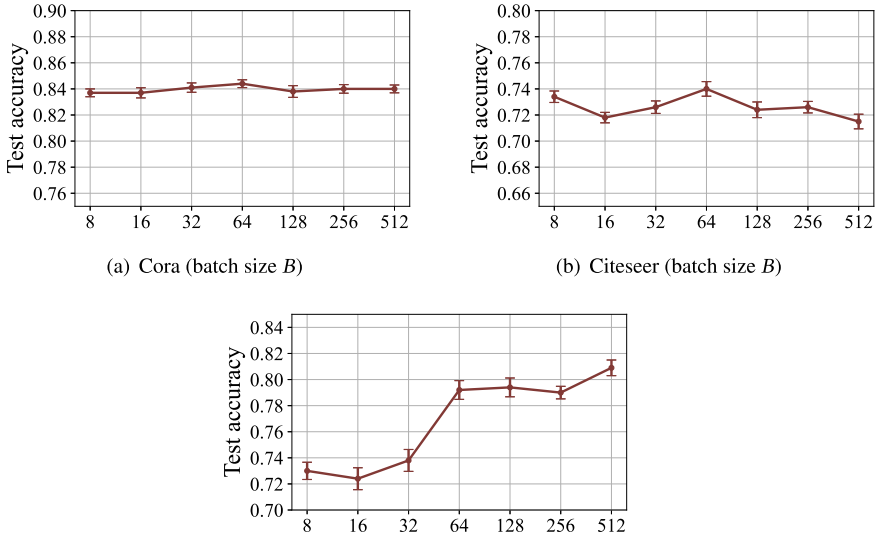


Fig. 7. Comparison of various backbones.

results are close and comparable with the GCN, showing our GKT framework is adaptable to various GNNs. (2) GraphSAGE and SGC mostly achieve better results than their original methods in Table 4, and GAT achieves comparable results, verifying the effectiveness of our framework in semi-supervised learning. (3) our model may achieve better improvement with parameter-lightweight backbones such as GCN and SGC, which may be because too many trainable parameters may aggravate the possibility of overfitting and dampen the effectiveness of meta-learning optimization.

5.9 Analysis of the Training Batch Size (RQ5)

We first analyze the effect of the training batch size B. Notably, the batch size is the number of input nodes in each training step. It also controls the size of the reconstructed graph in the knowledge transfer in our model. We set the batch size as $B = \{8, 16, 32, 64, 128, 256, 512\}$ and report the results in Figure 8. We can observe that GKT is robust to the batch size on the Cora and Citeseer datasets and achieves better results with a larger batch size on Pubmed. Even with small batch sizes, GKT can achieve relatively state-of-the-art performance on three datasets. The results show the batch



(a) Cora (batch size B)

(b) Citeseer (batch size B)

(c) Pubmed (batch size B)

Fig. 8. Analysis of the training batch size B.

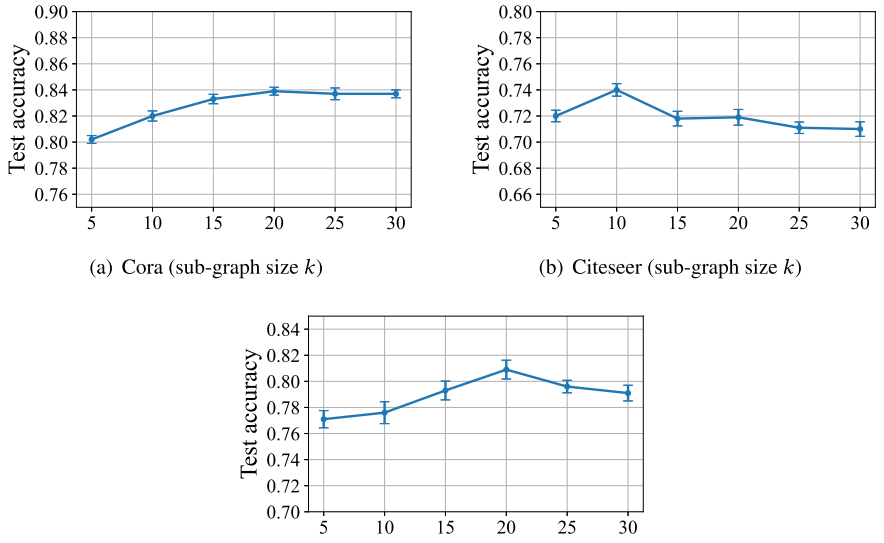
size (the size of the reconstructed graph) has little impact on the model performance, and we may adopt a relatively large batch size on large-scale graph data.

5.10 Analysis of the Sub-graph Size (RQ5)

In the unsupervised channel, we sample neighborhoods of nodes and maximize the mutual information between nodes and their neighborhoods to learn unsupervised node embeddings. Thus, we analyze the effect of the sub-graph size k, which is to control the size of sampled neighborhoods of nodes. The results of $k = \{5, 10, 15, 20, 25, 30\}$ are reported in Figure 9. We can observe that even with small sub-graph sizes, GKT can achieve competitive performance with other baselines, showing that GKT can learn high-quality embeddings from restrained regional structure information. Also, the curves are stable, showing that GKT is generally low-sensitive to sub-graph sizes, and the best size varies among different datasets.

6 CONCLUSIONS

In this article, we propose a semi-supervised learning framework on graphs via dual-channel knowledge transfer and meta-learning. We use the dual-channel models to encode classification probability and unsupervised embeddings of nodes, respectively. We use unsupervised node embeddings to reconstruct batch-wise graphs as the distance metric to smooth the probability distributions, which improves the generalization of the classifier. To avoid introducing extra label-unrelated unsupervised information, we also use the probability distributions to guide the training of unsupervised node embeddings, which makes the reconstructed graph adjusted towards being related to the labels. Thus, the proposed knowledge transfer model can adaptively balance the generalization and fitting capability for classification. The meta-learning optimization on the channel losses and knowledge transfer loss can improve the effectiveness of the knowledge transfer via the meta-gradient optimization to avoid the negative transfer between the two channels. Finally, the experimental results demonstrate the effectiveness of our model. In future work, we plan to use this framework on other related real-world applications related to semi-supervised learning.



(a) Cora (sub-graph size k)

(b) Citeseer (sub-graph size k)

(c) Pubmed (sub-graph size k)

Fig. 9. Analysis of the sub-graph size k.

However, since the backbone of GKT is graph convolutional networks, it may suffer from the common problem of GCN, such as the over-smoothing problem, which means that when stacking too many layers of GCNs, the output node embeddings would be converged to the same point in the embedding space. However, the GKT framework is feasible with most graph neural networks, including those deep GNN models. This article focuses on proposing a general pipeline for semi-supervised learning on graphs, and we leave the above problem for future work.

In general, we believe this article may open up the investigation of semi-supervised learning via knowledge transfer and meta-learning and have a possible impact on the research community.

REFERENCES

[1] Firoj Alam, Shafiq Joty, and Muhammad Imran. 2018. Graph-based semi-supervised learning with convolution neural networks to classify crisis related tweets. In *12th International AAAI Conference on Web and Social Media*.

[2] Jiyang Bai, Yuxiang Ren, and Jiawei Zhang. 2020. Ripple walk training: A subgraph-based training framework for large and deep graph neural network. *arXiv preprint arXiv:2002.07206* (2020).

[3] Muthu Balaanand, N. Karthikeyan, S. Karthik, R. Varatharajan, Gunasekaran Manogaran, and C. B. Sivaparthipan. 2019. An enhanced graph-based semi-supervised learning algorithm to detect fake users on Twitter. *J. Supercomput.* 75, 9 (2019), 6085–6105.

[4] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*.

[5] Kaixuan Chen, Lina Yao, Dalin Zhang, Xianzhi Wang, Xiaojun Chang, and Feiping Nie. 2019. A semisupervised recurrent convolutional attention model for human activity recognition. *IEEE Trans. Neural Netw. Learn. Syst.* 31, 5 (2019), 1747–1756.

[6] Ming Chen, Zhewei Wei, Bolin Ding, Yaliang Li, Ye Yuan, Xiaoyong Du, and Ji-Rong Wen. 2020. Scalable graph neural networks via bidirectional propagation. *Adv. Neural Inf. Process. Syst.* 33 (2020), 14556–14566.

[7] Zhengming Ding, Sheng Li, Ming Shao, and Yun Fu. 2018. Graph adaptive knowledge transfer for unsupervised domain adaptation. In *European Conference on Computer Vision (ECCV)*. 37–52.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*. PMLR, 1126–1135.

[9] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. 2017. Forward and reverse gradient-based hyperparameter optimization. In *International Conference on Machine Learning*. PMLR, 1165–1173.

[10] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*. PMLR, 1568–1577.

[11] Alberto Garcia Duran and Mathias Niepert. 2017. Learning graph representations with embedding propagation. *Adv. Neural Inf. Process. Syst.* 30 (2017), 5119–5130.

[12] Sujatha Das Gollapalli, Cornelia Caragea, Prasenjit Mitra, and C. Lee Giles. 2015. Improving researcher homepage classification with unlabeled data. *ACM Trans. Web* 9, 4 (2015), 1–32.

[13] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *31st International Conference on Neural Information Processing Systems*. 1025–1035.

[14] Xueting Han, Zhenhuan Huang, Bang An, and Jing Bai. 2021. Adaptive transfer learning on graph neural networks. In *27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 565–574.

[15] Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. 2021. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* (2021). DOI: https://doi.org/10.1109/TPAMI.2021.3079209

[16] Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. 2021. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 9 (2021), 5149–5169.

[17] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. GPT-GNN: Generative pre-training of graph neural networks. In *26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1857–1867.

[18] Kexin Huang and Marinka Zitnik. 2020. Graph meta learning via local subgraphs. *Adv. Neural Inf. Process. Syst.* 33 (2020), 5862–5874.

[19] Yizhu Jiao, Yun Xiong, Jiawei Zhang, Yao Zhang, Tianqi Zhang, and Yangyong Zhu. 2020. Sub-graph contrast for scalable self-supervised graph representation learning. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 222–231.

[20] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[21] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. In *International Conference on Machine Learning*. PMLR, 3734–3743.

[22] Junnan Li, Caiming Xiong, and Steven C. H. Hoi. 2021. CoMatch: Semi-supervised learning with contrastive graph regularization. In *IEEE/CVF International Conference on Computer Vision*. 9475–9484.

[23] Qimai Li, Xiao-Ming Wu, Han Liu, Xiaotong Zhang, and Zhichao Guan. 2019. Label efficient semi-supervised learning via graph filtering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9582–9591.

[24] Jiawei Luo, Pingjian Ding, Cheng Liang, and Xiangtao Chen. 2018. Semi-supervised prediction of human miRNA-disease association based on graph regularization framework in heterogeneous networks. *Neurocomputing* 294 (2018), 29–38.

[25] Minnan Luo, Xiaojun Chang, Liqiang Nie, Yi Yang, Alexander G. Hauptmann, and Qinghua Zheng. 2017. An adaptive semisupervised feature analysis for video semantic recognition. *IEEE Trans. Cyber.* 48, 2 (2017), 648–660.

[26] Qiaozhu Mei, Duo Zhang, and ChengXiang Zhai. 2008. A general optimization framework for smoothing language models on graph structures. In *31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 611–618.

[27] Paul Micaelli and Amos Storkey. 2020. Non-greedy gradient-based hyperparameter optimization over long horizons. *arXiv preprint arXiv:2007.07869* (2020).

[28] Daniel Carlos Guimarães Pedronette, Ying Weng, Alexandro Baldassin, and Chaohuan Hou. 2019. Semi-supervised and active learning through manifold reciprocal kNN graph for image retrieval. *Neurocomputing* 340 (2019), 19–31.

[29] Hao Peng, Ruitong Zhang, Yingtong Dou, Renyu Yang, Jingyi Zhang, and Philip S. Yu. 2021. Reinforced neighborhood selection guided multi-relational graph neural networks. *ACM Trans. Inf. Syst.* 40, 4 (2021), 1–46.

[30] Hao Peng, Ruitong Zhang, Shaoning Li, Yuwei Cao, Shirui Pan, and Philip Yu. 2022. Reinforced, incremental and cross-lingual event detection from social messages. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 1 (2022), 980–998.

[31] Zhen Peng, Yixiang Dong, Minnan Luo, Xiao-Ming Wu, and Qinghua Zheng. 2020. Self-supervised graph representation learning via global context prediction. *arXiv preprint arXiv:2003.01604* (2020).

[32] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph representation learning via graphical mutual information maximization. In *the Web Conference*. 259–270.

[33] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 701–710.

[34] Ziyue Qiao, Yi Du, Yanjie Fu, Pengfei Wang, and Yuanchun Zhou. 2019. Unsupervised author disambiguation using heterogeneous graph convolutional network embedding. In *IEEE International Conference on Big Data (Big Data)*. IEEE, 910–919.

[35] Ziyue Qiao, Yanjie Fu, Pengyang Wang, Meng Xiao, Zhiyuan Ning, Denghui Zhang, Yi Du, and Yuanchun Zhou. 2022. RPT: Toward transferable model on heterogeneous researcher data via pre-training. *IEEE Trans. Big Data* 9, 1 (2022), 186–199.

[36] Ziyue Qiao, Pengyang Wang, Yanjie Fu, Yi Du, Pengfei Wang, and Yuanchun Zhou. 2020. Tree structure-aware graph representation learning via integrated hierarchical aggregation and relational metric learning. In *IEEE International Conference on Data Mining (ICDM)*. IEEE, 432–441.

[37] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph contrastive coding for graph neural network pre-training. In *26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1150–1160.

[38] Asmaa Rassil, Hiba Chougrad, and Hamid Zouaki. 2020. The importance of local labels distribution and dominance for node classification in graph neural networks. In *19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 1505–1511.

[39] Yuxiang Ren, Jiyang Bai, and Jiawei Zhang. 2021. Label contrastive coding based graph neural network for graph classification. In *Database Systems for Advanced Applications*, Christian S. Jensen, Ee-Peng Lim, De-Nian Yang, Wang-Chien Lee, Vincent S. Tseng, Vana Kalogeraki, Jen-Wei Huang, and Chih-Ya Shen (Eds.). Springer International Publishing, Cham, 123–140.

[40] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Mag.* 29, 3 (2008), 93–93.

[41] Yuanjie Shao, Nong Sang, Changxin Gao, and Li Ma. 2017. Probabilistic class structure regularized sparse representation graph for semi-supervised hyperspectral image classification. *Pattern Recog.* 63 (2017), 102–114.

[42] Min Shi, Yufei Tang, Xingquan Zhu, and Jianxun Liu. 2020. Topic-aware web service representation learning. *ACM Trans. Web* 14, 2 (2020), 1–23.

[43] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. *Adv. Neural Inf. Process. Syst.* 32 (2019), 1919–1930.

[44] Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Conference on Empirical Methods in Natural Language Processing*. 167–176.

[45] Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang. 2020. InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=r1lfF2NYvH.

[46] Qiuling Suo, Jingyuan Chou, Weida Zhong, and Aidong Zhang. 2020. TAdaNet: Task-adaptive network for graph-enriched meta-learning. In *26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1789–1799.

[47] Ioannis A. Tamposis, Konstantinos D. Tsirigos, Margarita C. Theodoropoulou, Panagiota I. Kontou, and Pantelis G. Bagos. 2019. Semi-supervised learning of hidden Markov models for biological sequence analysis. *Bioinformatics* 35, 13 (2019), 2208–2215.

[48] Zhengzheng Tang, Ziyue Qiao, Xuehai Hong, Yang Wang, Fayaz Ali Dharejo, Yuanchun Zhou, and Yi Du. 2021. Data augmentation for graph convolutional network on semi-supervised classification. In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*. Springer, 33–48.

[49] Yingjie Tian, Mahboubeh Mirzabagheri, Peyman Tirandazi, and Seyed Mojtaba Hosseini Bamakan. 2020. A non-convex semi-supervised approach to opinion spam detection by ramp-one class SVM. *Inf. Process. Manag.* 57, 6 (2020), 102381.

[50] Ashwini Tonge and Cornelia Caragea. 2020. Image privacy prediction using deep neural networks. *ACM Trans. Web* 14, 2 (2020), 1–32.

[51] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *J. Mach. Learn. Res.* 9, 11 (2008).

[52] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

[53] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. 2018. Deep graph infomax. In *International Conference on Learning Representations*.

[54] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. 2012. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*. Springer, 639–655.

[55] Daqing Wu, Xiangyang Guo, Xiao Luo, Ziyue Qiao, and Jinwen Ma. 2022. Adaptive harmony learning and optimization for attributed graph clustering. In *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[56] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International Conference on Machine Learning*. PMLR, 6861–6871.

[57] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[58] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*. PMLR, 40–48.

[59] Huaxiu Yao, Chuxu Zhang, Ying Wei, Meng Jiang, Suhang Wang, Junzhou Huang, Nitesh Chawla, and Zhenhui Li. 2020. Graph few-shot learning via knowledge transfer. In *AAAI Conference on Artificial Intelligence*. 6656–6663.

[60] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. 2020. GraphSAINT: Graph sampling-based inductive learning method. In *International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=BJe8pkHFwS.

[61] Dalin Zhang, Lina Yao, Kaixuan Chen, Sen Wang, Xiaojun Chang, and Yunhao Liu. 2019. Making sense of spatio-temporal preserving representations for EEG-based human intention recognition. *IEEE Trans. Cyber.* 50, 7 (2019), 3033–3044.

[62] Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. 2020. Graph-Bert: Only attention is needed for learning graph representations. *arXiv preprint arXiv:2001.05140* (2020).

[63] Yabin Zhang, Bin Deng, Kui Jia, and Lei Zhang. 2020. Label propagation with augmented anchors: A simple semi-supervised learning baseline for unsupervised domain adaptation. In *European Conference on Computer Vision*. Springer, 781–797.

[64] Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. 2019. Meta-GNN: On few-shot node classification in graph meta-learning. In *28th ACM International Conference on Information and Knowledge Management*. 2357–2360.

[65] Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2021. Meta learning for knowledge distillation. *arXiv preprint arXiv:2106.04570* (2021).

[66] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. *Adv. Neural Inf. Process. Syst.* 33 (2020).

[67] Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02–107, Carnegie Mellon University.

[68] Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In *20th International Conference on Machine Learning (ICML'03)*. 912–919.

Received 31 January 2022; revised 16 August 2022; accepted 20 October 2022