Pengfei Ding Macquarie University Sydney, Australia pengfei.ding2@students.mq.edu.au Yan Wang* Macquarie University Sydney, Australia yan.wang@mq.edu.au Guanfeng Liu Macquarie University Sydney, Australia guanfeng.liu@mq.edu.au

ABSTRACT

In recent years, heterogeneous graph few-shot learning has been proposed to address the label sparsity issue in heterogeneous graphs (HGs), which contain various types of nodes and edges. The existing methods have achieved good performance by transferring generalized knowledge extracted from rich-labeled classes in source HG(s) to few-labeled classes in a target HG. However, these methods only consider the single-heterogeneity scenario where the source and target HGs share a fixed set of node/edge types, ignoring the more general scenario of cross-heterogeneity, where each HG can have a different and non-fixed set of node/edge types. To this end, we focus on the unexplored cross-heterogeneity scenario and propose a novel model for Cross-heterogeneity Graph Few-shot Learning, namely CGFL. In CGFL, we first extract meta-patterns to capture heterogeneous information and propose a multi-view heterogeneous graph neural network (MHGN) to learn meta-patterns across HGs. Then, we propose a score module to measure the informativeness of labeled samples and determine the transferability of each source HG. Finally, by integrating MHGN and the score module into a meta-learning mechanism, CGFL can effectively transfer generalized knowledge to predict new classes with few-labeled data. Extensive experiments on four real-world datasets have demonstrated the superior performance of CGFL over the state-of-the-art methods.

CCS CONCEPTS

Computing methodologies → Neural networks; Learning latent representations.

KEYWORDS

Heterogeneous graphs; Few-shot learning; Graph neural networks

ACM Reference Format:

CIKM '23, October 21-25, 2023, Birmingham, United Kingdom.

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM. ACM ISBN 979-8-4007-0124-5/23/10...\$15.00 https://doi.org/10.1145/XXXXXXXXXXXX



Figure 1: Scenarios of heterogeneous graph few-shot learning (HGFL).

1 INTRODUCTION

Heterogeneous Graphs (HGs) consisting of diverse node types and diverse edge types, are pervasive in a wide variety of real-world systems, such as social networks, citation networks, and e-commerce networks [15]. When learning the representations of nodes and edges in HGs, label scarcity is a common issue due to the expense and difficulties of data annotation in practice [26]. To address this issue, heterogeneous graph few-shot learning (HGFL) has been developed recently, which combines few-shot learning methods [27] with heterogeneous graph neural networks [22]. The aim of HGFL is to transfer the generalized knowledge extracted from rich-labeled classes in source HG(s), to learn few-labeled classes in a target HG. HGFL has been successfully applied to various applications, such as few-shot node classification [28], few-shot link prediction [1], and few-shot graph classification [8].

Existing studies on HGFL mainly focus on the scenario of *singleheterogeneity*, where the source and target HGs share a fixed set of node/edge types. In such a case, the knowledge of common node/edge types can be extracted from source HGs and transferred to the target HG to predict classes with few-labeled data. For instance, in Fig. 1, the source HG G_1 and target HG G_2 both contain node types of "paper", "workshop" and "author". The knowledge of the "paper" node type can be learned from G_1 and transferred to G_2 to predict the class of the "paper P_3 " (*i.e.*, *biology*). Similarly, the knowledge of "movie" node type can be transferred from G_3 to G_4 to predict the class of the "movie M_3 " (*i.e.*, *cartoon*).

However, in real-world scenarios, source HGs with rich-labeled classes may not always have the single-heterogeneity. In such a case, existing HGFL methods will be ineffective because they are unable

^{*}Corresponding author

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

to extract generalized knowledge across HGs through common node/edge types. For example, as shown in Fig. 1, if there are only two source HGs G_1 and G_3 , existing HGFL methods cannot extract knowledge of "user" and "organization" from G_1 and G_3 , thereby limiting their capability to predict the class of "users" in G_5 .

Based on the above discussion, a question arises: is it possible to extract generalized knowledge from other source HGs with different heterogeneities? In fact, different HGs may share relations that do not rely on specific node/edge types [4, 13]. These shared relations can facilitate the extraction of generalized knowledge across HGs with different heterogeneities. One typical relation is the affiliation relation [13, 16], which represents the association between nodes with individual properties and nodes with set properties. For instance, despite the disparate node types in G_1 , G_3 , and G_5 , they all contain affiliation relations such as "paper-workshop", "moviedirector", and "user-organization". By leveraging the similarities and patterns derived from the affiliation relations existing in source HGs G_1 and G_3 (e.g., "papers" published in the same "workshop" tend to have similar research topic labels, and "movies" directed by the same "director" tend to have similar theme labels), we can extract generalized knowledge that nodes with the same affiliation relation tend to have similar labels. This knowledge can be transferred to the target HG G_5 to classify "users" based on their work "organizations", e.g., U_1 and U_2 working at the same "organization" (*i.e.*, *lab*) may have similar professions (*i.e.*, *researchers*).

In addition to the affiliation relation, another important relation is the *interaction relation* [13, 16], which represents the connection between nodes with individual properties. For instance, in Fig. 1, G_1 , G_3 , and G_5 all contain interaction relations such as "authorauthor", "audience-audience" and "user-user". Specifically, "authors" A_1 and A_2 in G_1 "collaborate" with each other, "audiences" C_1 and C_2 in G_3 "attend screenings" together, and U_1 and U_3 in G_5 "play tennis" together. By analyzing the labels of these nodes (*e.g.*, both A_1 and A_2 focus on *data mining*, while both C_1 and C_2 are *sci-fi fans*), and the semantics of these connections (*e.g.*, "collaborate" and "attend screenings" both represent forms of partnership), we can infer that nodes connected through interaction relations, which imply partnerships, are likely to share similar personal interests and preferences. This inference can be applied to the target HG G_5 to identify "users" (*i.e.*, U_1 and U_3) who potentially share hobbies.

The above-mentioned example illustrates the existence of common relations in HGs with different heterogeneities. These relations can be considered as general patterns that reflect the underlying connections among these HGs. We refer to these general patterns as *meta-patterns*. Meta-patterns in real-world HGs may exhibit more complex structures and encompass a wider range of categories. Therefore, by further exploring various meta-patterns, we can discover generalized knowledge across HGs in the cross-heterogeneity scenario.

In light of the above discussion, the generalizability across HGs with different heterogeneities should be explored to handle a more general scenario of *cross-heterogeneity graph few-shot learning*, namely, transferring generalized knowledge extracted from source HGs with multiple heterogeneities, to make predictions on the target HG with a different heterogeneity and few-labeled data. This is a novel problem, which, however, has the following key challenges.

CH1. How to normalize information from HGs with different heterogeneities to extract generalized knowledge implied in these HGs? HGs with different heterogeneities typically exhibit significant variations in terms of node/edge features, node-node interactions, and graph structures. Therefore, in order to enable the extraction of generalized knowledge implied in these HGs, it is essential to develop a comprehensive framework that can normalize the heterogeneous information derived from HGs with various heterogeneities.

CH2. *How to achieve effective knowledge transfer from source HGs to the target HG*? Since source HGs and the target HG differ in their structural similarities and heterogeneity relevance, not all source HGs contain sufficient generalized knowledge that can be transferred to the target HG. Additionally, even within the same HG, different samples may exhibit varying degrees of generalized knowledge due to their distinct surrounding heterogeneous environments. Therefore, to achieve effective knowledge transfer, it is crucial to select samples and HGs that encompass generalized knowledge across various HGs.

CH3. Given few-labeled samples from a new heterogeneity, how to effectively learn the classes associated with these samples? In the target HG, since the few-labeled samples interact with various types of nodes and edges, each sample plays a distinct role in characterizing its respective class. However, measuring the importance of these samples from the new heterogeneity is challenging, because the knowledge of these interacted node/edge types cannot be transferred from source HGs. Consequently, to mitigate the influence of noise and outliers in these few-labeled samples, a robust and effective few-shot learning model is required.

To address the above three challenges, we propose a Crossheterogeneity Graph Few-shot Learning model, namely CGFL. In order to address CH1, we propose a general approach to extract meta-patterns across HGs with different heterogeneities, and adopt a novel multi-view heterogeneous graph neural network (MHGN) module to effectively generalize the information of these metapatterns. To address CH2, we propose a three-level score module to evaluate (1) the transferability of source HGs, (2) the consistency of few-shot tasks, and (3) the informativeness of labeled samples. This module allows CGFL to perform hierarchical and preferential learning from the source HG data, facilitating the extraction of generalized knowledge in a stable manner and achieving effective knowledge transfer. To address CH3, we propose a novel metalearning module that transfers the knowledge of measuring node informativeness from source HGs to the target HG. This module can effectively estimate the importance of few-labeled samples and generate highly robust class representations for accurate prediction in the target HG.

To the best of our knowledge, our work is the first to propose the novel problem of cross-heterogeneity graph few-shot learning. Our contributions can be summarized as follows:

- We propose the CGFL model, which can learn transferable knowledge across HGs with multiple heterogeneities and adapt to predicting new classes with a different heterogeneity and few-labeled data;
- We propose a novel multi-view heterogeneous graph neural network module that can be generalized to HGs with different heterogeneities, and propose a novel three-level score

CIKM '23, October 21-25, 2023, Birmingham, United Kingdom.

module that can evaluate the source HG data to achieve effective knowledge transfer;

• We conduct extensive experiments on four real-world datasets. Since our approach is the first to address this novel problem, we compare it against 12 representative and state-of-the-art baselines from four categories. The experimental results illustrate that CGFL outperforms the best-performing baselines by an average of 5.56% in accuracy and 4.97% in macro-F1 score.

2 RELATED WORK

Heterogeneous Graph Neural Networks (HGNNs). HGNNs have shown promising results in learning representations of HGs. Some HGNNs directly model various types of nodes and edges. For example, HGT [10] calculates heterogeneous attention scores for 1-hop neighboring nodes w.r.t. edge types. Simple-HGN [14] incorporates learnable edge type embeddings in edge attention, while HetSANN [9] employs a type-aware attention layer. SGAT [12] extends GAT [20] to HGs using simplicial complexes and upper adjacencies. Other HGNNs focus on modeling meta-paths to extract hybrid semantics in HGs. For instance, HAN [23] designs node-level attention and semantic-level attention to hierarchically aggregate features from meta-path based neighbors. MAGNN [7] further considers intermediate nodes along the meta-paths on the basis of HAN. However, existing HGNNs focus on learning specific node types and meta-paths within a single heterogeneity, which limits their ability to handle HGs with multiple heterogeneities.

Graph Few-shot Learning. Most studies on graph few-shot learning focus on homogeneous graphs [27]. For example, Meta-GNN [30] applies the MAML [6] algorithm to address the low-resource learning problem on graphs. G-Meta [11] samples local subgraph surrounding the target node to transfer subgraph-specific information. TENT [21] proposes to reduce the variance among tasks for generalization performance. Some recent studies have extended few-shot learning paradigms to heterogeneous graphs. For instance, HINFShot [31] and HG-Meta [29] target few-shot problems on a single citation network. CrossHG-Meta [28] focuses on transferring knowledge across HGs with different graph structures but the same node/edge types. These approaches rely on the transfer of generalized knowledge through shared node/edge types between source and target HGs. Consequently, they are not applicable to the cross-heterogeneity scenario where the source and target HGs have completely different node/edge types.

Recently, MetaGS [4] transfers knowledge between two HGs to predict semantic relations, it focuses on capturing comprehensive relationships between two nodes and requires the existence of semantic relations in the two HGs. Therefore, MetaGS cannot be generalized to deal with cross-heterogeneity few-shot problems.

3 PRELIMINARIES

Heterogeneous Graph. A heterogeneous graph, denoted as $G = (\mathcal{V}, \mathcal{E}, \phi, \psi)$, consists of a node set \mathcal{V} , an edge set \mathcal{E} , a node type mapping function $\phi : \mathcal{V} \mapsto \mathcal{A}$, and an edge type mapping function $\psi : \mathcal{E} \mapsto \mathcal{R}$. \mathcal{A} and \mathcal{R} represent the set of node types and the set of edge types, respectively, where $|\mathcal{A}| + |\mathcal{R}| > 2$. The notations used in this paper are summarized in Table 1.

Table 1: Notations used in this paper.

Notation	Explanation					
G	heterogeneous graph					
\mathcal{V}	set of nodes					
З	set of edges					
ϕ	node type mapping function					
ψ	edge type mapping function					
${\mathcal A}$	set of node types					
${\mathcal R}$	set of edge types					
${\cal H}$	single heterogeneity					
T_i	set of few-shot tasks on \mathcal{H}_i					
\mathcal{G}_{src}	set of source HGs					
G_t	target HG					
S	set of few-shot tasks on source HGs					
${\mathcal T}$	set of few-shot tasks on the target HG					
$\tau = (\tau_{spt}, \tau_{qry})$	single few-shot task					
$\hat{\mathcal{P}}$	set of meta-patterns					
Ι	set of instances matching meta-patterns in ${\cal P}$					
proto _c	prototype embedding of class c					

Heterogeneity. The heterogeneity of G_i , denoted as $\mathcal{H}_i = (\mathcal{A}_i, \mathcal{R}_i)$, contains the node type set \mathcal{A}_i and the edge type set \mathcal{R}_i . G_i and G_j have different heterogeneities if $\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ and $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$. **Problem Formulation.** We focus on the problem of few-shot node classification across HGs with different heterogeneities. To mimic the few-shot scenario, the data of each heterogeneity \mathcal{H}_i is considered as a collection of *m* few-shot tasks $T_i = \{\tau_1, \tau_2, \dots, \tau_m\}$.

- Input: Source HGs $\mathcal{G}_{src} = \{G_1, G_2, \dots, G_n\}$ and their sets of fewshot tasks $\mathcal{S} = \{T_1, T_2, \dots, T_n\}$. Target HG G_t with a set of few-shot tasks \mathcal{T} . Each graph in \mathcal{G}_{src} has a different heterogeneity to G_t .
- **Output:** A model with good generalization ability to few-shot tasks on the target HG (*i.e.*, T), after training with few-shot tasks on source HGs (*i.e.*, S).

Few-shot Task Construction. Few-shot tasks $T = \{\tau_1, \tau_2, ..., \tau_m\}$ in *G* are constructed as follows. Firstly, under the *N*-way *K*-shot setting, each task $\tau = (\tau_{spt}, \tau_{qry})$ is sampled by randomly choosing *N* different classes $C_{\tau} = \{c_1, c_2, ..., c_N\}$ from the label space *C*. Specifically, the support set $\tau_{spt} = \{\tau_{c_1}, \tau_{c_2}, ..., \tau_{c_N}\}$ is created by sampling *K* labeled nodes per class, *i.e.*, $\tau_{c_i} = \{(v_1, c_i), (v_2, c_i), ..., (v_K, c_i)\}$. The query set $\tau_{qry} = \{\tilde{\tau}_{c_1}, \tilde{\tau}_{c_2}, ..., \tilde{\tau}_{c_N}\}$ is formed from the remaining data of each class, where $\tilde{\tau}_{c_i} = \{(\tilde{v}_1, c_i), (\tilde{v}_2, c_i), ..., (\tilde{v}_K, c_i)\}$. After sufficient training iterations over *S* with the proposed methodology, the obtained model is expected to conduct *N*-way classification in \mathcal{T} given only *K* labeled examples per class.

4 THE PROPOSED CGFL

Fig. 2(a) shows the structure of our proposed CGFL model, which contains three main steps: (1) *Meta-pattern Extraction: meta-patterns* are extracted and classified into several general categories to capture heterogeneous information across HGs; (2) *Multi-view Learning*: a new multi-view heterogeneous graph neural network (MHGN) module is proposed to aggregate meta-pattern information from three general views: *sum, max* and *mean*; (3) *Meta-learning*: In the meta-training process, a three-level score module is proposed to

assess the transferability of source HGs, the consistency of few-shot tasks, and the informativeness of labeled nodes. In the meta-testing process, the node-level score submodule is generalized to the target HG to evaluate the importance of each few-labeled node. Overall, CGFL can be well initialized from few-shot tasks in source HGs and adapt to the target HG, enabling accurate prediction of few-labeled classes from a different heterogeneity.

4.1 Meta-pattern Extraction

In existing methods, the extraction of heterogeneous information from HGs typically relies on the utilization of predefined *patterns* referred to as meta-paths [19] or meta-graphs [5]. These patterns consist of diverse node types and have specific semantics, such as "author-paper-author" indicating author collaboration. By matching the HG with multiple pre-defined patterns, various types of heterogeneous information embedded in the HG can be extracted for further analysis. However, the process of mining patterns for a single HG typically requires domain expertise [2], thereby presenting challenges in extracting general patterns in the cross-heterogeneity scenario. On the one hand, mining useful patterns for all source HGs is impractical due to the extensive domain knowledge required. On the other hand, mining valuable patterns in the target HG becomes arduous due to limited knowledge regarding the new heterogeneity.

To tackle this challenge, we propose a novel method for extracting heterogeneous information across HGs automatically. Specifically, we first extract important patterns in HGs and form *metapatterns*. These meta-patterns are then classified into four general categories, enabling the extraction of generalized knowledge across HGs with different heterogeneities.

Procedure for Extracting Meta-patterns. For a single HG *G*, we propose a random walk-based approach to capture prevalent local structures and form meta-patterns: (1) Starting from each node in *G*, N_{path} paths with a length of *l* are randomly chosen for each node, and these paths are aggregated into a set \mathbf{S}_{path} . (2) For every node type $\Phi_i \in \mathcal{A}$ (Φ_i denotes the *i*-th node type), subpaths that connect nodes with the same node type Φ_i are extracted from \mathbf{S}_{path} , forming the set \mathbf{S}_{path}^i . (3) \mathbf{S}_{path}^i is then partitioned into distinct groups based on the type patterns of the subpaths (*e.g.*, subpaths with the "author-paper-author" pattern and the "author-venue-author" pattern are assigned to different groups). (4) Meta-patterns for Φ_i are identified by selecting the patterns with the top- K_{mp} highest counts. Hence, the meta-patterns in *G* can be represented as $\mathcal{P} = \{\mathcal{P}_i | \Phi_i \in \mathcal{A}\}$.

By identifying and summarizing patterns in HGs to form metapatterns, we can effectively compress a large volume of diverse heterogeneous information into a more concise format. In addition, applying a uniform approach to deal with various HGs can facilitate extracting generalized knowledge from these HGs.

Categorization for Meta-patterns. Our categorization approach firstly divides meta-patterns into two main categories: *affiliation patterns* (APs) and *interaction patterns* (IPs). Within each category, we further classify the patterns into two subcategories based on the strength of the relationship they represent. For instance, APs can be further categorized as either *strong affiliation patterns* (SAPs) or *weak affiliation patterns* (WAPs). This categorization is inspired by existing studies that adopt two general types of relations to classify various relationships in HGs [13, 16]: the *affiliation relation* between

nodes with individual properties and nodes with set properties (*e.g.*, "paper-venue"), and the *interaction relation* between nodes with individual properties (*e.g.*, "author-author").

However, these relations can only capture generalized nodenode connections and are incapable of capturing generalized graph structures across HGs. Moreover, in the cross-heterogeneity scenario, each relation category may contain diverse information that requires further classification. To overcome these limitations, CGFL extends the two types of relations from node-level to a more intricate meta-pattern level and provides a more detailed categorization of meta-patterns. Specifically, we first categorize meta-patterns into APs and IPs as follows.

$$D_{mp}(p) = \max(\{D(\Phi_i, \Phi_j) | (\Phi_i, \Phi_j) \in p\}), \tag{1}$$

$$D(\Phi_i, \Phi_j) = \frac{\max(d_{\Phi_i}, d_{\Phi_j})}{\min(d_{\Phi_i}, d_{\Phi_i})},$$
(2)

where *p* is a single meta-pattern, d_{Φ_i} and d_{Φ_j} are average degrees of node types Φ_i and Φ_j respectively. $D(\Phi_i, \Phi_j)$ determines the type of relationship between Φ_i and Φ_j [16]. A large value of $D(\cdot)$ indicates an affiliation relation, while a low value suggests an interaction relation. Then, we set a threshold θ_{mp} to classify meta-patterns, with those having $D_{mp}(\cdot) \geq \theta_{mp}$ classified as affiliation patterns and those with $D_{mp}(\cdot) < \theta_{mp}$ classified as interaction patterns. The classification is based on whether the meta-pattern contains the affiliation relation or not. θ_{mp} is typically set to 10, which is in line with previous studies on the classification of affiliation relations and interaction relations [16].

Next, we categorize affiliation patterns (APs) and interaction patterns (IPs) based on the strength of the relationships they represent. For APs, we analyze whether the pattern is symmetric, as symmetrical patterns typically indicate stronger relationships [19]. For instance, the symmetric AP "user-company-user" indicates that two "users" are connected through the same type of affiliation relation (i.e., "user-company"), suggesting that these two "users" are likely to be colleagues working in the same "company". Conversely, the asymmetric AP "user-company-item-user" indicates that two "users" are connected through different types of affiliation relations (i.e., "user-company" and "company-item-user"). In this case, one "user" may be an employee of the "company" while the other could be a customer of its products. These two "users" may have distinct roles, which weakens the relationship between them compared to the "user-company-user" pattern. Consequently, a symmetrical AP is classified as a strong affiliation pattern (SAP), whereas an asymmetrical AP is categorized as a weak affiliation pattern (WAP).

Regarding IPs, since each IP does not contain affiliation relations, nodes in IPs interact with each other on an equal basis. Nodes exhibit strong interaction relationships when they are closely connected, either directly or through a few common neighbors. Therefore, IPs with shorter lengths represent strong interaction relationships and are considered *strong interaction patterns* (SIPs), while IPs with longer distances do not indicate a close interaction relationship and are classified as *weak interaction patterns* (WIPs). We set a length threshold θ_{lp} to specify SIPs and WIPs.

Categorizing APs and IPs allows for a more comprehensive study of diverse meta-patterns. This categorization helps discover latent

CIKM '23, October 21-25, 2023, Birmingham, United Kingdom.



Figure 2: (a) The overall architecture of CGFL. (b) The three-level score module.

correlations within meta-patterns from each category, thereby facilitating the extraction of generalized knowledge. The categorization of meta-patterns yields four distinct groups: \mathcal{P}^{SAP} , \mathcal{P}^{WAP} , \mathcal{P}^{SIP} , and \mathcal{P}^{WIP} , each with its own set of instances, e.g., I^{SAP} .

4.2 Multi-view Learning

Each meta-pattern category can be represented by aggregating the information from its individual instances. However, directly aggregating meta-pattern instances may result in a loss of valuable information due to the different structures and node types in each category. To address this issue, we propose a multi-view mechanism inspired by widely used graph aggregation methods [25]. This mechanism captures information from three perspectives: the *sumview*, *max-view*, and *mean-view*. These views have been proven to be general enough to encompass the capabilities of various graph aggregators [25], thereby enabling CGFL to extract comprehensive and generalized knowledge across HGs.

- In the sum-view, we capture the complete information of instances for each meta-pattern category, *i.e.*, $I^{sum-SIP} = I^{SIP}$, here we use SIP as an example for the three views.
- In the max-view, we focus on extracting information related to the most prevalent meta-pattern within each category:

$$\mathcal{I}^{max-SIP} = \{I | I \in \mathcal{I}^{SIP}, \mathcal{P}(I) = p_{N-max}^{SIP}\},$$
(3)

where $\mathcal{P}(I)$ represents the meta-pattern associated with instance I, $p_{N\text{-max}}^{SIP}$ is the meta-pattern in \mathcal{P}^{SIP} that has the maximum number of instances.

 In the mean-view, we capture the distributions of all meta-patterns by averagely sampling N_{mean} instances for each meta-pattern p ∈ P^{SIP} and forming the set I^{mean-SIP}.

Thus, we can create instance sets for the three views: I^{sum} , I^{max} and I^{mean} . Each set consists of instances corresponding to different meta-pattern categories, *e.g.*, $I^{sum-SAP}$, $I^{sum-WAP}$, $I^{sum-SIP}$, $I^{sum-WIP}$ }. By leveraging these views to learn meta-patterns from different categories, we can extract comprehensive and generalized knowledge across HGs. Next, we adopt the following three functions to aggregate the information of meta-patterns and compute node representations.

Meta-pattern Instance Aggregation. To begin with, we perform the aggregation of instances belonging to the same meta-pattern category and the same view (*e.g.*, $I^{sum-SIP}$). Let $I_v^{sum-SIP}$ represent a set of instances that start from node v and are extracted from $I^{sum-SIP}$. An instance encoder is employed to transform the node features of each instance into a single vector:

$$\mathbf{h}_{v-i}^{sum-SIP} = f_{\theta}^{SIP}(\{\mathbf{x}_u | u \in I_{v-i}^{sum-SIP}\}), \tag{4}$$

where \mathbf{x}_u is the feature of node u, $I_{v-i}^{sum-SIP}$ represents the *i*-th instance in $I_v^{sum-SIP}$, and $f_{\theta}^{SIP}(\cdot)$ denotes the shared encoder function for SIP instances. Since each instance contributes differently to the representation of the target node, we learn the importance weight for each instance and utilize a multi-head attention mechanism to

calculate weighted sums of the instance representations:

$$\alpha_{v-i}^{sum-SIP} = \frac{\exp\left(\sigma\left(\mathbf{a}_{sum-SIP}^{\mathsf{T}} \cdot \left[\mathbf{x}_{v} \| \mathbf{h}_{v-i}^{sum-SIP}\right]\right)\right)}{\sum_{j=1}^{|\mathcal{I}_{v}^{sum-SIP}|} \exp\left(\sigma\left(\mathbf{a}_{sum-SIP}^{\mathsf{T}} \cdot \left[\mathbf{x}_{v} \| \mathbf{h}_{v-j}^{sum-SIP}\right]\right)\right)},$$
 (5)

$$\mathbf{h}_{v}^{sum-SIP} = \prod_{k=1}^{K_{att}} \sigma\left(\sum_{i=1}^{|\mathcal{I}_{v}^{sum-SIP}|} \alpha_{v-i,k}^{sum-SIP} \cdot \mathbf{h}_{v-i}^{sum-SIP}\right), \tag{6}$$

where $\alpha_{v-i,k}^{sum-SIP}$ denotes the importance of *i*-th instance at the *k*-th attention head, \parallel denotes the concatenation operation, $\mathbf{a}_{sum-SIP}$ represents the trainable attention parameter shared among SIP instances obtained from the sum-view, K_{att} corresponds to the number of attention heads, and $\sigma(\cdot)$ denotes an activation function. **Meta-pattern Category Aggregation.** Next, we aggregate the information from the four meta-pattern categories under the same view. Specifically, we adopt an attention mechanism to aggregate information from these meta-pattern categories:

$$w_{sum-SIP} = \sigma \left(\left(\mathbf{W}_{sum} \cdot \mathbf{h}_{v}^{sum-SIP} + \mathbf{b}_{sum} \right) \cdot \mathbf{a}_{sum} \right), \tag{7}$$

$$\beta_{sum-SIP} = \exp(w_{sum-SIP}) / \sum_{i \in \mathbf{S}_{mp}} \exp(w_{sum-i}), \tag{8}$$

$$\mathbf{h}_{v}^{sum} = \sum_{i \in \mathbf{S}_{mp}} \beta_{sum \cdot i} \cdot \mathbf{h}_{v}^{sum \cdot i},\tag{9}$$

where $S_{mp} = \{SAP, WAP, SIP, WIP\}$ is the set of meta-pattern categories, W_{sum} and b_{sum} are trainable parameters, a_{sum} denotes the shared attention parameter for meta-pattern categories in the sum-view.

View Aggregation. Finally, we aggregate the information from the three views to obtain the final embedding:

$$z_{sum} = \sigma \left(\left(\mathbf{W}_{view} \cdot \mathbf{h}_{v}^{sum} + \mathbf{b}_{view} \right) \cdot \mathbf{a}_{view} \right), \qquad (10)$$

$$\gamma_{sum} = \exp(z_{sum}) / \sum_{i \in S_{view}} \exp(z_i), \tag{11}$$

$$\mathbf{h}_{v} = \sum_{i \in \mathbf{S}_{view}} \gamma_{i} \cdot \mathbf{h}_{v}^{i}, \tag{12}$$

where $S_{view} = \{sum, max, mean\}$ is the set of views, W_{view} and b_{view} are trainable parameters, and a_{view} denotes a view level attention parameter.

4.3 Meta-learning

The meta-learning module consists of two steps: meta-training and meta-testing. During meta-training, to achieve effective knowledge transfer from source HGs, we propose a three-level score module (as shown in Fig. 2(b)) that evaluates the source HGs' data from three perspectives: HG transferability, task consistency, and node informativeness. Subsequently, we adopt a prototypical network to transfer the generalized knowledge from source HGs to the target HG. In meta-testing, the node-level score submodule, which has been learned from the source HGs, is adapted to the target HG. This adaptation enables the evaluation of the importance of few-labeled nodes in the target HG, thereby facilitating the generation of more stable representations for classes in the target HG.

Graph-level Score Submodule. This submodule is proposed to evaluate the transferability of each source HG. By comparing the similarity of heterogeneous information between the source HG and the target HG, we determine whether the source HG contains sufficient generalized knowledge for effective knowledge transfer. Specifically, we calculate the transferability of each source HG by comparing the meta-pattern distributions between this source HG and the target HG. Since the mean-view in the multi-view learning module captures the distributions of all meta-patterns, we utilize the vector \mathbf{h}_v^{mean} output by the mean-view (*cf.* Eq. 9) to compute the representation of the meta-pattern distribution for each HG:

$$\mathbf{h}_{G_i} = mean(\{\mathbf{h}_v^{mean} | v \in \tau, \tau \in T_i\}), \tag{13}$$

where τ is an *N*-way *K*-shot task, T_i is the set of tasks for *i*-th source HG G_i , and $mean(\cdot)$ denotes the averaging operation. For the target HG G_t , we randomly select $N \times K \times |T_i|$ nodes and input them into the current model to obtain h_v^{mean} and form h_{G_t} using the $mean(\cdot)$ operation. Note that this process is solely for obtaining representations of different HGs in the same vector space, and therefore the gradients during the vector acquisition process are not optimized. Next, we concatenate the representations of each source HG and the target HG and adopt softmax to compute the graph-level score for each source HG:

$$gs_{i} = \frac{\exp\left(\sigma\left(\mathbf{W}_{g}\left[\mathbf{h}_{G_{i}}||\mathbf{h}_{G_{t}}\right]\right)\right)}{\sum_{j=1}^{|\mathcal{G}_{sc}|}\exp\left(\sigma\left(\mathbf{W}_{g}\left[\mathbf{h}_{G_{j}}||\mathbf{h}_{G_{t}}\right]\right)\right)},$$
(14)

where \mathbf{W}_q is the trainable parameter.

Task-level Score Submodule. This submodule is proposed to evaluate the consistency of each task. In meta-learning, each task consists of a support set and a query set. The support set is utilized to adapt the model to a specific task, while the query set is employed to evaluate the model's performance on unseen examples from the same task. When it comes to few-shot learning on HGs, there may exist meta-pattern differences between nodes in the support and query sets, even if they are sampled from the same class. These discrepancies in meta-patterns can impact the feedback provided by the query set, potentially resulting in inaccurate evaluations of the model's ability to acquire knowledge from the support set. However, existing methods solely consider the graph structure to assess the task importance [18, 29], disregarding the influence of meta-pattern differences. Therefore, we propose to assess the importance of each task by examining the consistency between the meta-patterns surrounding nodes in the query set and support set. Similar to the mean-view vector \mathbf{h}_n^{mean} used for comparing meta-pattern distributions at the graph-level score, we utilize the sum-view vector \mathbf{h}_v^{sum} to represent all meta-patterns surrounding vand generate representations for the support set and query set:

$$\mathbf{h}_{\tau_{sot}^{i}} = mean(\{\mathbf{h}_{v}^{sum} | v \in \tau_{spt}^{i}\}), \tag{15}$$

$$\mathbf{h}_{\tau_{qrv}^{i}} = mean(\{\mathbf{h}_{v}^{sum} | v \in \tau_{qry}^{i}\}), \tag{16}$$

where $\tau^i = \{\tau^i_{spt}, \tau^i_{qry}\}$ is a single task. Then, we concatenate the representations of the support set and query set and apply the softmax function to compute the task-level score for each task:

$$ts_{i} = \frac{\exp\left(\sigma\left(\mathbf{W}_{t}\left[\mathbf{h}_{\tau_{spt}^{i}}||\mathbf{h}_{\tau_{qry}^{i}}\right]\right)\right)}{\sum_{j=1}^{|T|}\exp\left(\sigma\left(\mathbf{W}_{t}\left[\mathbf{h}_{\tau_{spt}^{j}}||\mathbf{h}_{\tau_{qry}^{j}}\right]\right)\right)},$$
(17)

where W_t is the trainable parameter. Since we only compare tasks within the same HG, *T* is the set of tasks of a single source HG. **Node-level Score Submodule**. This submodule is proposed to evaluate the informativeness of each labeled node. We consider

that the informativeness of the node is highly correlated to the importance of its meta-pattern instances. Consequently, we first calculate the importance of each instance as follows (here we take *i*-th SIP instance of node v, *i.e.*, I_{p-i}^{SIP} , as an example):

$$\epsilon_{v-i}^{SIP} = \frac{\exp\left(LeakyReLU\left(\mathbf{a}_{SIP}^{\mathsf{T}} \cdot \left[s_{v}\||s_{v-i}^{SIP}\right]\right)\right)}{\sum_{j=1}^{|I_{v}^{SIP}|}\exp\left(LeakyReLU\left(\mathbf{a}_{SIP}^{\mathsf{T}} \cdot \left[s_{v}\||s_{v-j}^{SIP}\right]\right)\right)},\tag{18}$$

where I_v^{SIP} is the set of instances that start from v and match patterns in \mathcal{P}^{SIP} , \mathbf{a}_{SIP} is the trainable attention parameter shared by SIP instances. s_v and s_{v-i}^{SIP} denote the initial scores of v and I_{v-i}^{SIP} , respectively, which are computed as follows:

$$s_v = \tanh\left(\mathbf{W}_s \cdot \mathbf{x}_v\right),\tag{19}$$

$$S_{v-i}^{SIP} = \tanh\left(\mathbf{W}_{SIP} \cdot f_{agg}\left(\left\{\mathbf{x}_{u} | u \in I_{v-i}^{SIP}\right\}\right)\right),\tag{20}$$

where \mathbf{W}_s and \mathbf{W}_{SIP} are trainable parameters. The function $f_{agg}(\cdot)$ is used to aggregate the information of nodes in each instance, such as operations like $mean(\cdot), max(\cdot), \text{ or } sum(\cdot)$. Then, we utilize $|\mathcal{I}_v^{SIP}|$ to quantify the popularity of node v, and then apply a sigmoid non-linearity to calculate the node-level score for the SIP category, which can be expressed as follows:

$$ns_{v}^{SIP} = sigmoid\left(\log\left(|\mathcal{I}_{v}^{SIP}| + \eta\right) \cdot \sum_{i=1}^{|\mathcal{I}_{v}^{SIP}|} \epsilon_{v-i}^{SIP} \cdot s_{v-i}^{SIP}\right), \quad (21)$$

where η is a small constant. Next, we calculate the average score for all meta-pattern categories to obtain the node-level score:

$$ns_v = (ns_v^{SAP} + ns_v^{WAP} + ns_v^{SIP} + ns_v^{WIP})/4.$$
(22)

Prototypical Network. After obtaining the node-level score for each node sample, CGFL follows the idea of prototypical networks [17] and calculates the weighted average of *K*-shot embedded nodes belonging to class *c* to obtain the prototype of that class:

$$\operatorname{proto}_{c} = \sum_{i=1}^{K} \operatorname{sc}_{i} \cdot \mathbf{h}_{i}, \qquad (23)$$

where $sc_i = ns_i / \sum_{i=1}^{K} ns_i$ is the normalized node-level score, \mathbf{h}_i is the embedding of node *i* that is output by multi-view learning. The node-level score submodule is trained through meta-training and used to compute the class prototype in meta-testing.

Loss Function. During meta-training, the prototype for each class is computed by utilizing the nodes in the support set τ^{spt} . To determine the class of a query node u in the query set τ^{qry} , we calculate the probability for each class based on the distance between the node embedding \mathbf{h}_u and each prototype:

$$prob(c|u) = \frac{\exp\left(-d(\mathbf{h}_u, \mathbf{proto}_c)\right)}{\sum_{c'} \exp\left(-d(\mathbf{h}_u, \mathbf{proto}_{c'})\right)},$$
(24)

where $d(\cdot)$ is a distance metric function and we adopt squared Euclidean distance [17]. Under the episodic training framework, the objective of each meta-training task is to minimize the classification loss between the predictions of the query set and the ground truth. The training loss of a single task τ can be defined as the average negative log-likelihood probability of assigning correct class labels:

$$\mathcal{L}_{\tau} = -\frac{1}{N \times K} \sum_{i=1}^{N \times K} \log(prob(y_i^*|v_i)), \tag{25}$$

CIKM '23, October 21-25, 2023, Birmingham, United Kingdom.

Table 2: Statistics of Datasets.

Datasets	Node Type #Nodes	Labeled Node Type #Classes		
DBLP	Paper/Author/Venue 4,057 / 14,328 / 20	Paper 4		
IMDB	Movie/Director/Actor 4,661 / 2,270 / 5,841	Movie 4		
YELP	Business/User/Service/Star/Reservation 2,614 / 1,286 / 9 / 2 / 2	Business 3		
PubMed	Disease/Gene/Chemical/Species 20,163 / 13,561 / 26,522 / 2,863	Disease 8		

where y_i^* is the ground truth label of v_i . Then, by incoporating the graph-level score and task-level score, the total meta-training loss can be defined as:

$$\mathcal{L}_{meta} = \sum_{i=1}^{|\mathcal{G}_{src}|} \sum_{j=1}^{|T_i|} gs_i \cdot ts_j \cdot \mathcal{L}^i_{\tau_j}, \tag{26}$$

where $\mathcal{L}_{\tau_i}^i$ denotes the training loss on *j*-th task of *i*-th source HG.

4.4 Efficiency Analysis

Targeting the novel and complex cross-heterogeneity scenario where there may exist multiple large-scale source HGs with various node types, CGFL is efficient due to the following properties:

Few-shot learning based on local subgraphs: CGFL extracts subgraphs around *N*-way *K*-shot labeled nodes for learning, rather than processing the entire graph structure. This approach ensures the input graph size for CGFL remains small and consistent during computation, resulting in stable memory consumption and improved computational efficiency.

Independent of node type for heterogeneous information extraction: CGFL does not rely on specific node types to capture heterogeneous information; instead, it focuses on capturing four widely existing meta-pattern categories across HGs. This ensures that the computational complexity of CGFL remains unaffected when the number of node types increases.

5 EXPERIMENTS AND ANALYSIS

We conduct extensive experiments on four real-world datasets to answer the following research questions: **RQ1**: How does CGFL perform compared with representative and state-of-the-art methods? **RQ2**: How do the three proposed modules (*i.e.*, meta-pattern extraction module, multi-view learning module, and meta-learning module) contribute to the overall performance? **RQ3**: How does the number of source HGs affect the performance of HGFL? **RQ4**: How does CGFL perform under different parameter settings?

5.1 Experimental Settings

Datasets. We adopt four real-world datasets: DBLP, IMDB, YELP, and PubMed. These datasets are publicly available and have been widely used in studies of node classification in HGs [16, 23]. Details of the datasets are provided in Table 2.

Meta-learning Settings. Since the node types in any two of the four datasets are completely different, we choose to use one dataset as the target HG for meta-testing, while the remaining three datasets are used as source HGs for meta-training. This setting yields the

CIKM '23, October 21-25, 2023, Birmingham, United Kingdom.

Table 3: Node classification accuracy on four datasets.

	DBLP		IMDB		YELP		PubMed		
	2-way 3-way		2-way	2-way 3-way		3-way	2-way	3-way	
1-shot									
GAT	0.7814	0.6443	0.5092	0.3421	0.5839	0.3938	0.5023	0.3318	
SGC	0.7745	0.6599	0.5142	0.3423	0.5964	0.4039	0.5072	0.3360	
GIN	0.7804	0.6503	0.5173	0.3499	0.5902	0.4172	0.4961	0.3393	
HAN	0.6828	0.5497	0.5153	0.3514	0.5681	0.3896	0.5080	0.3367	
MAGNN	0.7006	0.5552	0.5144	0.3505	0.5774	0.3935	0.5092	0.3375	
SGAT	0.7897	0.6631	0.5234	0.3584	0.5932	0.4132	0.5066	0.3419	
MAML	0.6263	0.4651	0.5111	0.3404	0.5303	0.3674	0.5071	0.3232	
ProtoNet	0.6404	0.4917	0.5133	0.3397	0.5436	0.3665	0.5037	0.3242	
Meta-GNN	0.6974	0.5423	0.5175	0.3495	0.5769	0.3717	0.5106	0.3449	
GPN	0.7611	0.6506	0.5197	0.3603*	0.6207	0.4266^{*}	0.5048	0.3303	
G-Meta	0.7827	0.6694	0.5241^{*}	0.3573	0.6274	0.4242	0.5155^{*}	0.3461^{*}	
TENT	0.8124^{*}	0.7035*	0.5174	0.3592	0.6440*	0.4157	0.5121	0.3452	
CGFL	0.8712^{1}	0.7849	0.5331	0.3846	0.7028	0.4377	0.5335	0.3676	
Improvement ²	6.75%	10.37%	1.69%	6.32%	8.37%	2.54%	3.37%	5.85%	
			3-8	shot					
GAT	0.8371	0.7284	0.5313	0.3669	0.6393	0.4373	0.4683	0.3040	
SGC	0.8150	0.7389	0.5377	0.3764	0.6580	0.4391	0.4840	0.3027	
GIN	0.8433	0.7411	0.5287	0.3769	0.6547	0.4082	0.4923	0.2978	
HAN	0.7274	0.6061	0.5344	0.3764	0.6796	0.4596	0.5116	0.3527	
MAGNN	0.7522	0.6456	0.5275	0.3905	0.6389	0.4359	0.5266	0.3789	
SGAT	0.8382	0.7426	0.5332	0.3824	0.6434	0.4521	0.5379	0.3824	
MAML	0.6685	0.5082	0.5253	0.3507	0.5621	0.3967	0.4997	0.3237	
ProtoNet	0.6953	0.5799	0.5194	0.3602	0.5785	0.4231	0.4927	0.3117	
Meta-GNN	0.7983	0.6953	0.5309	0.3982	0.6271	0.4326	0.5140	0.3890	
GPN	0.8450	0.7982	0.5467^{*}	0.4042^{*}	0.6766	0.4596	0.5213	0.3979*	
G-Meta	0.8680^{*}	0.8138^{*}	0.5350	0.4021	0.6836*	0.4577	0.5392^{*}	0.3744	
TENT	0.8576	0.7538	0.5212	0.3677	0.6744	0.4680*	0.5317	0.3968	
CGFL	0.9026	0.8572	0.5745	0.4181	0.7532	0.4904	0.5833	0.4204	
Improvement ²	3.83%	5.06%	4.84%	3.32%	9.24%	4.57%	7.56%	5.35%	

* Result of the best-performing baseline.

¹ Bold numbers are the results of the best-performing method.

² Improvement of our CGFL over the best-performing baseline.

most challenging scenario for heterogeneous graph few-shot learning, wherein the datasets for meta-training and meta-testing have entirely different heterogeneities. We repeat such challenging scenarios to conduct all four separate experiments, where each of the four datasets serves as the target HG in one experiment.

Baselines. Given the absence of dedicated solutions designed for the cross-heterogeneity graph few-shot learning problem, we select 12 representative and state-of-the-art methods as baselines. These baselines can be categorized into four groups:

- Homogeneous GNNs: GAT [20], SGC [24] and GIN [25].
- Heterogeneous GNNs: HAN [23], MAGNN [7] and SGAT [12].
- Few-shot learning methods: *MAML* [6] and *ProtoNet* [17].
- Graph few-shot learning methods: *Meta-GNN* [30], *GPN* [3], *G-Meta* [11] and *TENT* [21].

While most baselines can follow our meta-learning setting with minor modifications, it is worth mentioning that HAN and MAGNN employ specialized modules to capture information from specific node types and meta-paths. Consequently, these models can only deal with HGs with single-heterogeneity. Therefore, we only train HAN and MAGNN on the target HG to evaluate the effectiveness of these methods in leveraging heterogeneous information of the target HG to address the few-shot problem.

Parameter Settings. In the *N*-way *K*-shot setting, *N* is set to $\{2, 3\}$ and *K* is set to $\{1, 3, 5\}$. The number of task *m* is set to 100 for

Pengfei Ding, Yan Wang, and Guanfeng Liu

Table 4: Node classification F1-score on four datasets.

	DBLP		IMDB		YELP		PubMed		
	2-way 3-way		2-way 3-way		2-way	3-way	2-way	3-way	
1-shot									
GAT	0.7634	0.6238	0.4874	0.3230	0.5717	0.3656	0.4697	0.3058	
SGC	0.7516	0.6316	0.4609	0.3077	0.5496	0.3754	0.4451	0.2757	
GIN	0.7626	0.6213	0.4838	0.3218	0.5705	0.3775	0.4544	0.3014	
HAN	0.6569	0.5258	0.4994	0.3367	0.5086	0.3353	0.4745	0.2958	
MAGNN	0.6967	0.5492	0.4851	0.3259	0.5467	0.3805	0.4837	0.3280	
SGAT	0.7712	0.6516	0.4895	0.3365	0.5719	0.3745	0.4828	0.3372	
MAML	0.6233	0.4622	0.5088	0.3373	0.5281	0.3449	0.4698	0.3202	
ProtoNet	0.6455	0.5343	0.4831	0.3210	0.5491	0.3316	0.4409	0.2772	
Meta-GNN	0.6683	0.4851	0.5063	0.3397	0.5531	0.3603	0.5039	0.3222	
GPN	0.7492	0.6294	0.5147*	0.3376	0.6082^{*}	0.3941	0.5008	0.3447^{*}	
G-Meta	0.7649	0.6427	0.5084	0.3436^{*}	0.5439	0.3954^{*}	0.5075*	0.3372	
TENT	0.8081*	0.6767*	0.5029	0.3205	0.5567	0.3814	0.4932	0.3358	
CGFL	0.8456 ¹	0.7536	0.5244	0.3469	0.6876	0.4126	0.5155	0.3520	
Improvement ²	4.43%	10.20%	1.85%	0.95%	11.55%	4.17%	1.11%	2.07%	
			3-5	shot					
GAT	0.8042	0.7067	0.5133	0.3545	0.6059	0.4105	0.4493	0.2742	
SGC	0.8092	0.7168	0.5229	0.3649	0.6388	0.4208	0.4388	0.2616	
GIN	0.8001	0.7282	0.5180	0.3546	0.6470	0.4011	0.4562	0.2695	
HAN	0.6909	0.5839	0.5231	0.3659	0.6574	0.4396	0.4961	0.3439	
MAGNN	0.7489	0.6150	0.5022	0.3729	0.6300	0.4230	0.5093	0.3668	
SGAT	0.8123	0.7146	0.5153	0.3620	0.6355	0.4361	0.5118	0.3723	
MAML	0.6453	0.5052	0.5167	0.3466	0.5598	0.3805	0.4907	0.3055	
ProtoNet	0.6343	0.5136	0.4484	0.2717	0.5217	0.3530	0.4771	0.3038	
Meta-GNN	0.7723	0.6553	0.5248	0.3656	0.5902	0.4207	0.5076	0.3715	
GPN	0.8295	0.7825^{*}	0.5316*	0.3811^{*}	0.6451	0.4338	0.5185	0.3797*	
G-Meta	0.8461*	0.7769	0.5246	0.3559	0.6494	0.4266	0.5313*	0.3319	
TENT	0.8396	0.7423	0.5089	0.3399	0.6592*	0.4430^{*}	0.5249	0.3566	
CGFL	0.8974	0.8512	0.5539	0.3957	0.7007	0.4664	0.5615	0.4013	
Improvement ²	5.72%	8.07%	4.03%	3.69%	5.92%	5.02%	5.38%	5.38%	

* Result of the best-performing baseline.

¹ Bold numbers are the results of the best-performing method.

 2 Improvement of our CGFL over the best-performing baseline.

all datasets. To ensure fair comparisons, the embedding dimension is set to 64 for both the baselines and CGFL; the parameters for each baseline are initially set to the values reported in the original paper and then optimized through grid-searching to achieve the best performance. For CGFL, in the meta-pattern extraction module, N_{path} is set to 20, l is set to 40, K_{mp} is set to 10, θ_{lp} is set to 3. In the multi-view learning module, N_{mean} is set to 5, $f_{\theta}^{SAP}(\cdot)$, $f_{\theta}^{WAP}(\cdot)$, $f_{\theta}^{SIP}(\cdot)$ and $f_{\theta}^{WIP}(\cdot)$ are set as mean-pooling(\cdot), K_{att} is set to 4, the activation $\sigma(\cdot)$ is set to relu(\cdot). In the meta-learning module, $f_{agg}(\cdot)$ is set to sum(\cdot).

Evaluation Metrics. The performance of all methods is evaluated by two widely used node classification metrics: *Accuracy* and *Macro-F1 score* [20, 30]. To ensure a fair and accurate assessment of the performance of all methods, we perform 10 independent runs for each *N*-way *K*-shot setting and report the average results.

5.2 **Experimental Results**

Performance Comparison with Baselines (RQ1). Tables 3 and 4 present the performance comparison between CGFL and baselines. The "DBLP" column presents the results of the experiment where the DBLP dataset is set as the target HG and the rest three datasets are set as source HGs. The results demonstrate a substantial improvement of CGFL over the best-performing baselines. On average, CGFL achieves a 5.56% increase in accuracy (ranging from 1.69%



Figure 3: Node classification performance of CGFL variants.

Table 5: Node classification performance of CGFL with dif-ferent numbers of source HGs on four datasets.

#Source	DBLP		IMDB		YELP		PubMed				
HGs	2-way	3-way	2-way	3-way	2-way	3-way	2-way	3-way			
	Accuracy: 1-shot										
Rand-1	0.8382	0.7305	0.5084	0.3382	0.6512	0.4042	0.5013	0.3346			
Rand-2	0.8644	0.7542	0.5193	0.3705	0.6745	0.4201	0.5122	0.3575			
All-3	0.8712	0.7849	0.5331	0.3846	0.7028	0.4377	0.5335	0.3676			
	Accuracy: 3-shot										
Rand-1	0.8746	0.8226	0.5475	0.3977	0.7079	0.4657	0.5356	0.3839			
Rand-2	0.8917	0.8423	0.5646	0.4129	0.7375	0.4812	0.5677	0.4065			
All-3	0.9026	0.8572	0.5745	0.4181	0.7532	0.4904	0.5833	0.4204			
			F1-s	core: 1-s	hot						
Rand-1	0.8008	0.7148	0.5035	0.3103	0.6427	0.3951	0.4863	0.3161			
Rand-2	0.8268	0.7343	0.5133	0.3262	0.6651	0.4035	0.5028	0.3372			
All-3	0.8456	0.7536	0.5244	0.3469	0.6876	0.4126	0.5155	0.3520			
F1-score: 3-shot											
Rand-1	0.8622	0.8149	0.5253	0.3546	0.6754	0.4243	0.5223	0.3736			
Rand-2	0.8864	0.8394	0.5500	0.3794	0.6925	0.4441	0.5439	0.3943			
All-3	0.8974	0.8512	0.5539	0.3957	0.7007	0.4664	0.5615	0.4013			

to 10.37%) and a 4.97% increase in F1-score (ranging from 0.95% to 11.55%). This improvement can be attributed to the utilization of the proposed multi-view HGNN and score module in CGFL, which effectively learn generalized knowledge across HGs. In contrast, the baselines struggle to generalize information from source HGs with different heterogeneities. thereby resulting in inferior performance when applied to a new heterogeneity with few-labeled data.

In addition, it is worth noting that some baselines exhibit a decline in performance as the number of labeled samples (*i.e.*, *K*) increases. For instance, on the PubMed dataset, the performance of homogeneous graph methods (GAT, SGC, GIN) and few-shot learning methods (MAML, ProtoNet) deteriorates in the 3-shot scenario compared to the 1-shot scenario. This decline can be attributed to their limited capacity to effectively extract generalized knowledge from source HGs, leading to negative transfer. In contrast, CGFL adopts the proposed score module to evaluate the transferability of source HG data, thereby mitigating negative transfer.

Ablation Study (RQ2). We create ten variants of CGFL to investigate the impact of its three main modules: (1) Four variants are created to investigate the impact of the meta-pattern extraction module by removing meta-patterns of SAP, WAP, SIP, and WIP. These variants are denoted as M\SAP, M\WAP, M\SIP, and M\WIP, respectively. (2) Three variants are developed to explore the influence of the multi-view learning module by removing the sum-view, max-view, and mean-view. These variants are denoted as M\Sum, M\Max, and M\Mean, respectively. (3) Three variants are created to study the impact of the score module by removing submodules of the graph-level score, task-level score, and node-level score. These variants are named M\G-Score, M\T-Score, and M\N-Score, respectively. From Fig. 3, we have the following observations:

- M\SAP, M\WAP, M\SIP, and M\WIP all exhibit inferior performance compared to the original CGFL. This indicates that exploring each of the four meta-pattern categories is crucial for effective knowledge transfer across HGs. Notably, the removal of SAP and SIP results in a more significant decline in performance (approximate decreases of 7% and 8% respectively) compared to WAP and WIP (approximate decreases of 3% and 5% respectively). This highlights the importance of extracting SAP and SIP and emphasizes the necessity to distinguish between meta-patterns based on their relationship strength levels.
- M\Sum, M\Max, M\Mean are outperformed by CGFL. This suggests that learning generalized relations from all three views enhances the generalizability across HGs.
- M\G-Score, M\T-Score and M\N-Score yield inferior performance compared to CGFL. This indicates that the score module plays a crucial role in evaluating source HG data and facilitating the extraction of generalized knowledge across HGs. Notably, the node-level score submodule has the greatest influence on performance, because it not only measures the informativeness of nodes in the source HGs but also determines the importance of few-labeled nodes to derive robust prototypes in the target HG.

Impact of the Number of Source HGs (RQ3). We investigate whether CGFL can effectively extract generalized knowledge from different types of heterogeneities by varying the number of source HGs. The results presented in Table 5 illustrate that as the number of source HGs increases, CGFL consistently demonstrates improved performance. This is due to CGFL's capability to enrich generalized knowledge by learning from more source HGs with various heterogeneities and effectively transfer the enriched generalized knowledge to the target HG.

Parameter Study (RQ4). We evaluate the sensitivity of several important parameters in CGFL, and show their impacts in Fig. 4. For the meta-pattern threshold θ_{mp} and length threshold θ_{lp} , moderate values should be set around 10 and 3 respectively. This selection helps effectively categorize meta-patterns into general categories. For the number of meta-patterns K_{mp} , a larger value (greater than 10) generally yields better results. This is because a larger K_{mp} allows CGFL to capture a broader range of diverse meta-patterns. For the number of instances in the mean-view N_{mean} , performance stabilizes when N_{mean} exceeds 5. This is because subsequently extracted instances may not possess distinct characteristics that contribute to the learning of meta-pattern distributions.

6 CONCLUSION

In this paper, we propose a novel cross-heterogeneity graph fewshot learning problem, and provide a solution called CGFL. CGFL incorporates a multi-view learning module to efficiently extract generalized knowledge across source HGs, and adopts a scorebased meta-learning module to transfer the knowledge to the target HG with different heterogeneity for few-shot learning. Extensive CIKM '23, October 21-25, 2023, Birmingham, United Kingdom.



Figure 4: Node classification accuracy of CGFL with different parameter settings on DBLP dataset.

experiments demonstrate the superior performance of our CGFL. In the future work, we plan to further enhance CGFL by distinguishing between heterogeneity-specific knowledge and heterogeneity-cross knowledge. Additionally, we plan to improve the meta-learning module in CGFL by considering the mutual information between source HGs and tasks originating from different source HGs.

REFERENCES

- Mingyang Chen, Wen Zhang, Yushan Zhu, Hongting Zhou, Zonggang Yuan, Changliang Xu, and Huajun Chen. 2022. Meta-knowledge transfer for inductive knowledge graph embedding. In SIGIR. 927–937.
- [2] Ting Chen and Yizhou Sun. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In WSDM. 295–304.
- [3] Kaize Ding, Jianling Wang, Jundong Li, Kai Shu, Chenghao Liu, and Huan Liu. 2020. Graph prototypical networks for few-shot learning on attributed networks. In CIKM. 295–304.
- [4] Pengfei Ding, Yan Wang, Guanfeng Liu, and Xiaofang Zhou. 2023. Few-Shot Semantic Relation Prediction Across Heterogeneous Graphs. IEEE TKDE (2023).
- [5] Yuan Fang, Wenqing Lin, Vincent W Zheng, Min Wu, Jiaqi Shi, Kevin Chen-Chuan Chang, and Xiao-Li Li. 2019. Metagraph-based learning on heterogeneous graphs. IEEE TKDE 33, 1 (2019), 154–168.
- [6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic metalearning for fast adaptation of deep networks. In *ICML*. 1126–1135.
- [7] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In WWW. 2331–2341.
- [8] Zhichun Guo, Chuxu Zhang, Wenhao Yu, John Herr, Olaf Wiest, Meng Jiang, and Nitesh V Chawla. 2021. Few-shot graph learning for molecular property prediction. In WWW. 2559–2567.
- [9] Huiting Hong, Hantao Guo, Yucheng Lin, Xiaoqing Yang, Zang Li, and Jieping Ye. 2020. An attention-based graph neural network for heterogeneous structural learning. In AAAI. 4132–4139.

- [10] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In WWW. 2704–2710.
- [11] Kexin Huang and Marinka Zitnik. 2020. Graph meta learning via local subgraphs. In NIPS. 5862–5874.
- [12] See Hian Lee, Feng Ji, and Wee Peng Tay. 2022. SGAT: Simplicial Graph Attention Network. arXiv preprint arXiv:2207.11761 (2022).
- [13] Yuanfu Lu, Chuan Shi, Linmei Hu, and Zhiyuan Liu. 2019. Relation structureaware heterogeneous information network embedding. In AAAI. 4456–4463.
- [14] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In KDD. 1150–1160.
- [15] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE TKDE* 29, 1 (2016), 17–37.
- [16] Chuan Shi, Yuanfu Lu, Linmei Hu, Zhiyuan Liu, and Huadong Ma. 2020. Rhine: relation structure-aware heterogeneous information network embedding. *IEEE TKDE* 34, 1 (2020), 433–447.
- [17] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In NIPS. 4077–4087.
- [18] Xiangyu Song, Jianxin Li, Qi Lei, Wei Zhao, Yunliang Chen, and Ajmal Mian. 2022. Bi-CLKT: Bi-graph contrastive learning based knowledge tracing. *Knowledge-Based Systems* 241 (2022), 108274.
- [19] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *PVLDB* 4, 11 (2011), 992–1003.
- [20] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2017. Graph attention networks. *stat* 1050, 20 (2017), 10–48550.
- [21] Song Wang, Kaize Ding, Chuxu Zhang, Chen Chen, and Jundong Li. 2022. Taskadaptive few-shot node classification. In KDD. 1910–1919.
- [22] Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and S Yu Philip. 2022. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. *IEEE Trans. on Big Data* 9, 2 (2022), 415–436.
- [23] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In WWW. 2022–2032.
- [24] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *ICML*. 6861– 6871.
- [25] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018).
- [26] Minji Yoon, John Palowitch, Dustin Zelle, Ziniu Hu, Ruslan Salakhutdinov, and Bryan Perozzi. 2022. Zero-shot Transfer Learning within a Heterogeneous Graph via Knowledge Transfer Networks. In NIPS. 27347–27359.
- [27] Chuxu Zhang, Kaize Ding, Jundong Li, Xiangliang Zhang, Yanfang Ye, Nitesh V Chawla, and Huan Liu. 2022. Few-Shot Learning on Graphs: A Survey. arXiv preprint arXiv:2203.09308 (2022).
- [28] Qiannan Zhang, Xiaodong Wu, Qiang Yang, Chuxu Zhang, and Xiangliang Zhang. 2022. Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer. In KDD. 2450–2460.
- [29] Qiannan Zhang, Xiaodong Wu, Qiang Yang, Chuxu Zhang, and Xiangliang Zhang. 2022. HG-Meta: Graph Meta-learning over Heterogeneous Graphs. In SDM. 397– 405.
- [30] 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 CIKM. 2357–2360.
- [31] Zifeng Zhuang, Xintao Xiang, Siteng Huang, and Donglin Wang. 2021. HINF-Shot: A Challenge Dataset for Few-Shot Node Classification in Heterogeneous Information Network. In *ICMR*, 429–436.