Heterogeneous Graph Transformer with Poly-Tokenization

Zhiyuan Lu[1] , **Yuan Fang**[2] , **Cheng Yang**[1] and **Chuan Shi**[1*]
[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Singapore Management University, Singapore
{luzy, yangcheng, shichuan}@bupt.edu.cn, yfang@smu.edu.sg

Abstract

Graph neural networks have shown widespread success for learning on graphs, but they still face fundamental drawbacks, such as limited expressive power, over-smoothing, and over-squashing. Meanwhile, the transformer architecture offers a potential solution to these issues. However, existing graph transformers primarily cater to homogeneous graphs and are unable to model the intricate semantics of heterogeneous graphs. Moreover, unlike small molecular graphs where the entire graph can be considered as the receptive field in graph transformers, real-world heterogeneous graphs comprise a significantly larger number of nodes and cannot be entirely treated as such. Consequently, existing graph transformers struggle to capture the long-range dependencies in these complex heterogeneous graphs. To address these two limitations, we present Poly-tokenized Heterogeneous Graph Transformer (PHGT), a novel transformer-based heterogeneous graph model. In addition to traditional node tokens, PHGT introduces a novel poly-token design with two more token types: semantic tokens and global tokens. Semantic tokens encapsulate high-order heterogeneous semantic relationships, while global tokens capture semantic-aware long-range interactions. We validate the effectiveness of PHGT through extensive experiments on standardized heterogeneous graph benchmarks, demonstrating significant improvements over state-of-the-art heterogeneous graph representation learning models.

1 Introduction

In recent years, graph neural networks (GNNs) have emerged as a powerful tool for learning on graph-structured data [Scarselli *et al.*, 2008; Kipf and Welling, 2017; Hamilton *et al.*, 2017; Veličković *et al.*, 2018]. They typically adopt a message-passing architecture that learns node representations by aggregating messages (i.e., features) from neighboring nodes in a recursive manner. Despite their effectiveness, message-passing GNNs still face several challenging problems. One major limitation is their expressive power, which is bounded by the 1-Weisfeiler-Lehman test [Xu *et al.*, 2019]. Moreover, they often suffer from the so-called "over-smoothing" [Chen *et al.*, 2020] and "over-squashing" [Alon and Yahav, 2021] issues as more message-passing layers are stacked. Given more layers, the receptive field of a node grows exponentially. Thus, node representations tend to become similar or overly smooth across the graph; at the same time, messages from the entire receptive field get overly compressed or squashed into fixed-length node vectors, leading to a loss of important information across the graph.

Meanwhile, the transformer architecture [Vaswani *et al.*, 2017] has revolutionized natural language processing [Devlin *et al.*, 2019; Brown *et al.*, 2020] and computer vision [Dosovitskiy *et al.*, 2021; Liu *et al.*, 2021], achieving state-of-the-art performance. This success has inspired the design of transformer architectures for graphs [Ying *et al.*, 2021; Chen *et al.*, 2022; Kreuzer *et al.*, 2021] as an appealing alternative to conventional message-passing GNNs. Graph transformers have demonstrated promising results, particularly on molecular prediction tasks [Ying *et al.*, 2021], owning to their fully-connected self-attention mechanism which can overcome the limitations of message-passing GNNs such as over-smoothing and over-squashing [Müller *et al.*, 2023]. However, most existing graph transformers are designed for homogeneous graphs, which cannot preserve the rich and complex semantics in heterogeneous graphs. Furthermore, the quadratic complexity of self-attention hinders the ability to handle a broader receptive field beyond the very small molecular graphs.

In particular, heterogeneous graphs or heterogeneous information networks (HIN) [Shi *et al.*, 2016] have become a popular choice for modeling complex real-world systems such as bibliographic and e-commerce networks. For example, as shown in Figure 1(a), in a bibliographic network there exist many nodes of diverse types, e.g., papers (P), authors (A), conferences (C). These heterogeneous nodes often interact via various relations, e.g., writing (A–P) or collaborating (A–P–A). To effectively capture such rich semantic information, heterogeneous graph neural networks (HGNNs) [Wang *et al.*, 2019; Hu *et al.*, 2020; Zhang *et al.*, 2019] have been proposed, but they boil down to the message-passing architecture and thus suffer from the same problems including limited ex-

*Corresponding author.

pressive power, over-smoothing and over-squashing. Hence, it becomes natural to consider transformers for heterogeneous graphs. Specifically, we need to address the following two challenges on heterogeneous graphs.

First, *how do we integrate the complex semantics on a heterogeneous graph into transformers?* The fundamental input blocks of transformers are *tokens*, e.g., words in language tasks or nodes in graph tasks, where the attention mechanism operates on. It is still unclear how the rich node types and relations on a heterogeneous graph can seamlessly integrate with the token-based input. A recent study [Mao *et al.*, 2023] introduced a transformer-based heterogeneous graph model called HINormer, but it only adjusts the attention scores between node tokens based on their heterogeneous types, ignoring explicit interactions between them and high-order semantic relations. For example, in a bibliographic network, an author node not only attends to different paper or conference nodes based on topical interest, but also focuses on different "strategies" that can be abstracted using semantic relations, e.g., seeking collaboration (A–P–A) or benchmarking competitors (A–P–C–P–A). Hence, we need to learn fine-grained attention between nodes and semantic abstractions, in addition to the traditional attention between nodes only.

Second, *how do we expand the receptive fields for graph transformers to capture long-range interactions on a heterogeneous graph?* Unlike small molecular graphs where the entire graph can be the receptive field, real-world heterogeneous graphs extend beyond the capacity to be treated entirely within a single receptive field. For instance, HINormer [Mao *et al.*, 2023] uniformly samples the neighbors around each node up to a predefined maximum depth. The sampled subgraphs are fed into the transformer, effectively reducing the number of input tokens. However, this also limits the receptive field of a node to its sampled subgraph, which ignores long-range dependencies and complex patterns in heterogeneous graphs [Li *et al.*, 2022; Li *et al.*, 2021; Li *et al.*, 2023]. Alternatively, linear transformer blocks [Rampášek *et al.*, 2022; Wu *et al.*, 2022] apply a linear kernel approximation to calculate the otherwise quadratic pairwise attention. Nevertheless, modeling the interaction for every pair of nodes is susceptible to noises, as most nodes can be irrelevant or unimportant to the query node. On the other hand, graph coarsening-based transformer models [Zhang *et al.*, 2022] attempt to compress a graph using its spectral features, which can be a computationally expensive process. Furthermore, except for HINormer, existing approaches are designed for homogeneous graphs without accounting for the abundant semantics on heterogeneous graphs.

To address these two challenges, we propose Poly-tokenized Heterogeneous Graph Transformer (PHGT), a transformer-based approach designed specifically for heterogeneous graphs. It boils down to the notion of *poly-tokens*, where distinct types of tokens are integrated into the transformer input to address both the heterogeneity and receptive field challenges. For the first challenge, we introduce a new type of token called the *semantic token*, in addition to the commonly used *node token* in existing graph transformers. A semantic token encodes a specific semantic detail and enjoys the same status as the node token in the transformer input, allowing fine-grained attention between not only nodes, but also nodes and heterogeneous semantics. Concretely, we materialize a semantic token as a meta-path [Sun *et al.*, 2011] instance embedding, which can be regarded as a basic semantic unit in heterogeneous graphs. For instance, a meta-path like A–P–A can signify a collaborator relationship, while A–P–C–P–A represents a peer relationship in the same field, abstracting the heterogeneous semantics into well-defined semantic units. The semantic tokenization essentially elevates the treatment of rich semantics as first-class citizens in the transformer architecture, enabling us to better exploit the semantics on a heterogeneous graph.

To tackle the second challenge, we utilize another type of token called semantic-aware *global token*. Each global token attempts to encode a structure- and semantic-coherent part of the global information on a graph, to complement the traditional node tokens that only capture local information. Concretely, we employ a fast, heterogeneous graph clustering algorithm aimed at grouping structurally and semantically related nodes into a set of global tokens, where each global token represents a cluster on the graph. This formulation allows the transformer to also model the interactions between each node and the global tokens across the entire graph. This effectively expands the receptive field of each node to capture the semantic-aware long-range interactions, yet without suffering from quadratic complexity.

Together, our ploy-token design with three distinct types of tokens, namely, node tokens, semantic tokens and global tokens, is well-suited to realize the transformer architecture for node-level tasks in heterogeneous graphs. Our major contributions are summarized as follows. (1) We introduce poly-tokens for heterogeneous graph transformers: the *semantic token* captures semantic relationships and the *global token* incorporates semantic-aware long-range information, to complement the traditional node tokens. (2) We further present Poly-tokenized Heterogeneous Graph Transformer or PHGT, which captures both semantic and global information in a unified manner through the poly-token design. (3) We perform extensive experiments on four benchmark heterogeneous graph datasets, and demonstrate the advantages of PHGT over state-of-the-art heterogeneous graph models.

2 Related Works

Heterogeneous Graph Neural Networks. GNNs have emerged as a powerful framework for analyzing structured data represented as graphs, as demonstrated in seminal works such as GCN [Kipf and Welling, 2017], GAT [Veličković *et al.*, 2018] and GraphSAGE [Hamilton *et al.*, 2017]. HGNNs [Yang *et al.*, 2020] have gained prominence as an extension to traditional GNNs, specifically tailored to handle graphs with diverse node and edge types. HAN [Wang *et al.*, 2019] relies on manually designed meta-paths and hierarchical aggregation to capture rich semantics. MAGNN [Fu *et al.*, 2020] further leverages node content features by using a relational rotation encoder to aggregate meta-path instances. Most HGNNs follow the message-passing paradigm, which makes them susceptible to the ongoing challenges of the paradigm, including restricted expressive power, over-

smoothing, and over-squashing. It is worth noting that, although Heterogeneous Graph Transformer (HGT) [Hu *et al.*, 2020] appears to explore the transformer architecture for heterogeneous graphs, it essentially utilizes an attention mechanism solely for neighborhood information aggregation. Consequently, HGT fundamentally remains a message-passing framework, and thus retains the associated limitations.

Transformers for Graph. For graph-structured data, the transformer architecture emerges as a promising alternative to address the limitations of the message-passing mechanism [Liu *et al.*, 2023]. Given the transformer's inherent lack of awareness about the underlying graph structure, researchers have focused on methods to integrate graph topological information into the transformer architecture. For instance, Graphormer [Ying *et al.*, 2021] employs pairwise shortest path distances to define relative positional encodings of the nodes. SAT [Chen *et al.*, 2022] leverages GNNs to extract subgraph structures and utilizes structural similarity for calculating attention weights. Meanwhile, other studies have attempted to expand the application of transformer architectures beyond the scope of small molecular graphs. NodeFormer [Wu *et al.*, 2022] employs a kernelized Gumbel-Softmax operator to transform the self-attention operation with quadratic complexity into one with linear complexity. Meanwhile, ANS-GT [Zhang *et al.*, 2022] adaptively samples informative nodes and captures long-range dependencies through graph coarsening algorithms. In a recent study [Mao *et al.*, 2023], researchers introduced a transformer-based model for heterogeneous graph representation learning, named HINormer, which demonstrates state-of-the-art performance in node classification tasks. However, HINormer does exhibit two notable limitations. First, its context sampling strategy leads to a constrained receptive field, resulting in the loss of global information. Second, HINormer only leverages heterogeneous types to modulate attention scores among node tokens, thereby overlooking explicit interactions between node tokens and high-order semantic information.

3 Notations and Preliminaries

Heterogeneous Graph. A heterogeneous graph or heterogeneous information network [Shi *et al.*, 2016], denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R}, \phi, \psi)$, is a graph with a set of nodes $\mathcal{V}$, a set of edges $\mathcal{E}$, a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and an edge type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{R}$. Here $\mathcal{A}$ and $\mathcal{R}$ denote the sets of predefined node types and edge types, respectively, such that $|\mathcal{A}| + |\mathcal{R}| > 2$.

Meta-Path. A meta-path [Sun *et al.*, 2011] $\mathcal{P}$ is defined as a sequence in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$ (abbreviated as $A_1 A_2 \cdots A_{l+1}$ when there is only one unique relation type between A_i and A_{i+1}), for $A_i \in \mathcal{A}$ and $R_i \in \mathcal{R}$. It describes a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between A_1 and A_{l+1}.

Transformer. The original transformer architecture has been designed for sequence data, employing a series of transformer layers [Vaswani *et al.*, 2017]. The self-attention mechanism is at the core of the transformer architecture. Given a sequence of input embeddings $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_m]^\top \in \mathbb{R}^{m \times d}$, the self-attention mechanism computes the weighted sum of embeddings:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_K}}\right)\mathbf{V}, \tag{1}$$

where $\mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \mathbf{K} = \mathbf{H}\mathbf{W}_K, \mathbf{V} = \mathbf{H}\mathbf{W}_V$ are known as the queries, keys and values, respectively, $\mathbf{W}_Q$, $\mathbf{W}_K$ and $\mathbf{W}_V$ are projection matrices, and d_K is the dimension of the queries/keys.

4 Methodology

In this section, we present the Poly-tokenized Heterogeneous Graph Transformer (PHGT) model, capitalizing on three distinct types of tokens to capture different grains of information in a heterogeneous graph. We start with an overview of the PHGT framework, followed by a detailed exposition of the different types of tokens.

4.1 Overall Framework

We illustrate the overall framework of our proposed PHGT in Figure 1. Given an input heterogeneous graph $\mathcal{G}$, we first employ a structure encoder to capture the local structures of the graph. Subsequently, a poly-tokenization mechanism is employed to model information from different perspectives.

Structure Encoder. The self-attention mechanism of the transformer calculates all pairwise interactions in a receptive field, ignoring the local structures in a graph (i.e., explicit neighborhood connectivity). Hence, we insert several graph convolutional layers before the transformer layers, utilizing the message-passing mechanism to encode local structures. As abstracted by Eq. (2), a structure encoder is applied to a graph $\mathcal{G}$ with adjacency matrix $\mathbf{A}$ and node feature matrix $\mathbf{X}$, to obtain an embedding matrix $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times d}$ for all nodes in the graph:

$$\mathbf{Z} = \texttt{StructureEnc}(\mathbf{A}, \mathbf{X}). \tag{2}$$

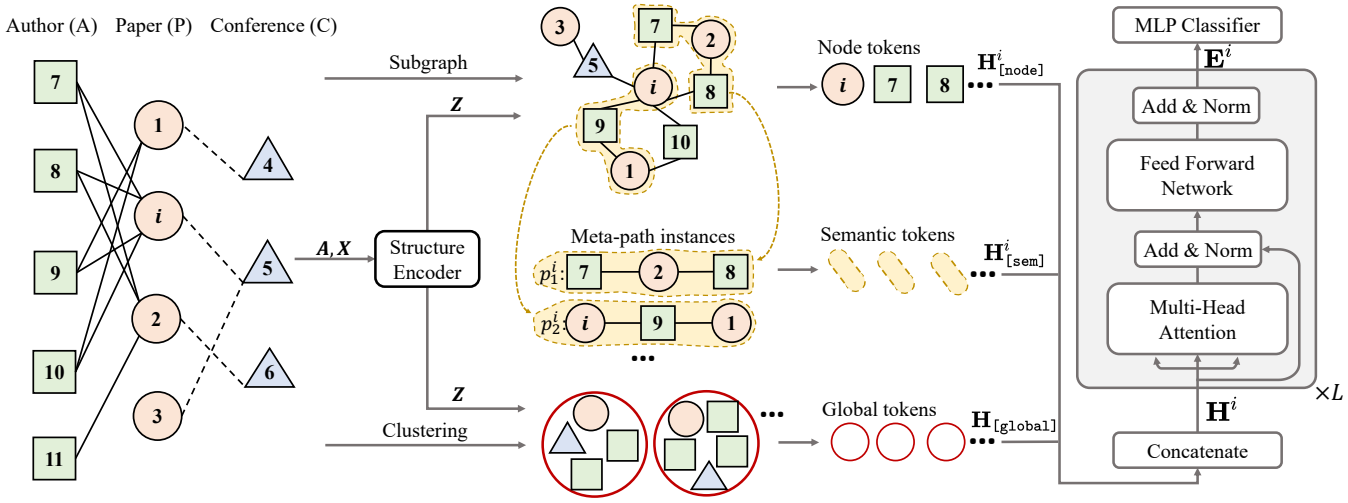
$\mathbf{Z}$ contains the preliminary node representations to construct the poly-token input for the transformer. Note that $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{|\mathcal{V}|}]^\top$ where each row corresponds to a d-dimensional embedding of a node in the graph.

Poly-Tokenization Mechanism. The key idea behind our approach lies in the utilization of distinct types of tokens to model different grains of information. Specifically, each node i is represented by a sequence of T token vectors $\mathbf{H}^i \in \mathbb{R}^{T \times d}$, which forms the input of the instance i to the transformer. In our poly-token design, $\mathbf{H}^i$ comprises three types of tokens:

$$\mathbf{H}^i = [\mathbf{H}^i_{[\texttt{node}]}, \mathbf{H}^i_{[\texttt{sem}]}, \mathbf{H}_{[\texttt{global}]}]^\top. \tag{3}$$

Here, $\mathbf{H}^i_{[\texttt{node}]}$ is a node token sequence to capture pairwise interactions in the local receptive field of node i. Next, $\mathbf{H}^i_{[\texttt{sem}]}$ is a semantic token sequence to preserve high-order semantic relations associated with node i on the heterogeneous graph. Moreover, $\mathbf{H}_{[\texttt{global}]}$ is a semantic-aware global token sequence that effectively expands the receptive field to model long-range interactions.

Among the poly-tokens, node token has been a traditional design in graph transformers. Due to the inherent quadratic



(a) A heterogeneous graph (b) Poly-tokenization: three types of token (c) A transformer block

Figure 1: The overall framework of PHGT.

complexity of the self-attention mechanism, it is infeasible to treat the whole graph as the receptive field except on very small graphs (e.g., molecular graphs). Following previous work [Mao *et al.*, 2023], for each target node i, we sample a subgraph $\mathcal{G}^i$ around i. Suppose $\mathcal{G}^i$ consists of a set of nodes $\mathcal{V}^i = \{v_1^i, v_2^i, \ldots, v_{|\mathcal{V}^i|}^i\}$. This subgraph serves as a local receptive field of v_i with $|\mathcal{V}^i|$ nodes only. Then, we can use these nodes to form a node token sequence, denoted by $\mathbf{H}_{[\texttt{node}]}^i \in \mathbb{R}^{|\mathcal{V}^i| \times d}$, that characterizes the target node i, as follows.

$$\mathbf{H}_{[\texttt{node}]}^i = [\mathbf{z}_1^i, \mathbf{z}_2^i, ..., \mathbf{z}_{|\mathcal{V}^i|}^i]^\top, \tag{4}$$

where $\mathbf{z}_k^i$ is the preliminary representation of node v_k^i from the structure encoder in Eq. (2). Note that although we call $\mathbf{H}_{[\texttt{node}]}^i$ a "sequence" following NLP convention, the order of the node tokens is irrelevant in graph domains[1].

Beyond the standard node tokens, we also propose the semantic and global tokens among the poly-tokens. As they form our key contributions, we defer their discussions to Sect. 4.2 and 4.3, respectively.

Transformer Block. The resulting poly-token sequence, $\mathbf{H}^i$ in Eq. (3), is used as the input of node i to the transformer layers, as shown in Figure 1(c). The computation formula of a transformer layer is provided in Appendix A. By stacking L transformer layers, PHGT can encode the node i and obtain its output embedding $\mathbf{E}^i$, followed by a task-specific layer such as a classifier for node classification.

4.2 Semantic Token

As part of the poly-token formulation, we introduce our semantic token to capture the rich semantic information on a heterogeneous graph. In this regard, previous work [Mao *et al.*, 2023] merely uses graph heterogeneity as side information to adjust self-attention. In contrast, our proposed semantic token stands as a first-class citizen in the transformer architecture, enabling the modeling of explicit interactions between nodes and high-order semantic relations. Specifically, we resort to the tool of *meta-path* [Sun *et al.*, 2011] to build our semantic tokens.

Meta-Path Instance as Semantic Token. Intuitively, a meta-path is a semantic pattern to describe a high-order relation in the heterogeneous graph (see Sect. 3). For example, in Figure 1(a), APA describes co-author relations and APCPA describes researchers in the same field. Each meta-path has a set of instances on the graph, e.g., 7–i–8 is an instance of APA which implies nodes 7 and 8 are co-authors. Naturally, a target node is associated with instantiations of various high-order semantic relations, which can be formulated into semantic tokens to characterize the semantic context of the target node.

Specifically, consider a pre-defined set of meta-paths $\mathcal{P}$. For each subgraph $\mathcal{G}^i$ surrounding a target node i, we extract a set of N meta-path instances $\mathcal{S}_{[\texttt{sem}]}^i = \{p_1^i, p_2^i, \ldots, p_N^i\}$, where each p_k^i is an instance of some meta-path in $\mathcal{P}$, and N is a hyperparameter. For example, Figure 1(b) illustrates an example of a subgraph centered on node i, where p_1^i : 7–2–8 is an instance of APA and p_2^i : i–9–1 is an instance of PAP. Each instance p_k^i is regarded as a semantic token, and we derive its embedding in the next step to form a semantic token sequence as part of the input to the transformer.

Semantic Token Embedding. For each semantic token $p_k^i \in \mathcal{S}_{[\texttt{sem}]}^i$, its embedding $\mathbf{z}_{p_k}^i$ is aggregated from that of every node in the corresponding meta-path instance using a readout function:

$$\mathbf{z}_{p_k}^i = \texttt{ReadOut}(\{\mathbf{z}_u \mid u \in p_k^i\}), \tag{5}$$

where u is a node in p_k^i, and $\mathbf{z}_u$ is the preliminary representation of node u obtained via the structure encoder in Eq. (2).

[1]To achieve permutation invariance, our implementation removes the positional encoding from the transformer architecture.

The choice of ReadOut is flexible, including sum or mean pooling and recurrent neural networks, among others. In our implementation, we opt for a simple mean pooling which shows strong empirical performance.

Finally, gathering all the semantic tokens for a target node i, we obtain a semantic token sequence for i:

$$\mathbf{H}^i_{[\texttt{sem}]} = [\mathbf{z}^i_{p_1}, \mathbf{z}^i_{p_2}, ..., \mathbf{z}^i_{p_N}]^\top, \tag{6}$$

which forms part of the input representing node i to the transformer, as shown in Eq. (3).

4.3 Global Token

To cope with the size of the receptive field, the traditional node tokens are taken from a small subgraph around each target node. Hence, the receptive field of the target node is limited, preventing it from receiving long-range interactions with nodes outside the subgraph.

To expand the receptive field, yet without incurring a quadratic overhead, we propose the concept of semantic-aware global token. In particular, the global token aims to extract global information on the graph in a semantic-aware manner, enabling a target node to attend to parts of the graph beyond its local receptive field. Thus, the inclusion of global tokens effectively expands the receptive field on a heterogeneous graph. More specifically, we resort to a heterogeneous graph clustering algorithm, where each cluster can be regarded as a global token.

Heterogeneous Cluster as Global Token. To extract global information from a graph, we perform clustering to group structurally and semantically related nodes. Each group or cluster of nodes can serve as a semantic-aware global token, and the set of global tokens provides a summary of the global information on the graph.

An efficient approach to cluster nodes in a graph is pseudo-label propagation [Gregory, 2010]. However, it is not designed for heterogeneous graphs, in which different node/edge types may entail significantly different roles in the graph. On the other hand, several clustering methods [Zhou and Liu, 2013; Zhang *et al.*, 2019] designed for heterogeneous graphs focus solely on clustering nodes of a specific type, which diverges from the goal of our global token to capture global information across the graph. Hence, we adapt the pseudo-label propagation algorithm to handle heterogeneous graphs. The high-level idea is to differentiate the edge weights based on node/edge types, and use these semantic-aware weights to guide the pseudo-label propagation process. Specifically, inspired by [Yang *et al.*, 2022], we introduce a heterogeneous variant of GAT [Veličković *et al.*, 2018], which learns the weight $\alpha_{i,j}$ of the edge (i, j) based on a type-wise projection of the node features. That is,

$$\alpha_{i,j} = \frac{\exp(\mathbf{W}_{\phi(i)}\mathbf{h}_i, \mathbf{W}_{\phi(j)}\mathbf{h}_j)}{\sum_{k\in\mathcal{N}_i} \exp(\mathbf{W}_{\phi(i)}\mathbf{h}_i, \mathbf{W}_{\phi(k)}\mathbf{h}_k)}, \tag{7}$$

where $\mathbf{W}_{\phi(i)}$ is a project matrix specific to the node type of i, and $\mathcal{N}_i$ is the set of neighbors of node i. After getting the edge weight $\alpha_{i,j}$, we use it to guide the label-propagation process.

As the heterogeneous graph clustering method is not the primary focus of our contribution, we provide its details in Appendix B.

Global Token Embedding. After clustering the graph $\mathcal{G}$, we obtain a set of M clusters $\{c_1, c_2, \ldots, c_M\}$, where M is a hyperparameter. Each cluster is regarded as a global token, and we aggregate the embeddings of nodes in each cluster as the global token embedding. Specifically, the embedding of the global token corresponding to the cluster c_k can be calculated as

$$\mathbf{z}_{c_k} = \texttt{ReadOut}(\{\mathbf{z}_u \mid u \in c_k\}), \tag{8}$$

where u is a node in c_k, and $\mathbf{z}_u$ is the preliminary representation of node u obtained via the structure encoder in Eq. (2). Again, we utilize mean pooling as the ReadOut function.

Finally, all the global tokens form a sequence:

$$\mathbf{H}_{[\texttt{global}]} = [\mathbf{z}_{c_1}, \mathbf{z}_{c_2}, ..., \mathbf{z}_{c_M}]^\top, \tag{9}$$

which is concatenated with the node and semantic token sequences of each node as the input to the transformer, as shown in Eq. (3).

4.4 Training Objective

Lastly, we demonstrate the training of PHGT for the classic node classification task on graphs. Given the embedding $\mathbf{E}^L$ obtained by the model, we adopt a multi-layer perceptron (MLP) with parameter θ to predict the label distribution of node i:

$$\hat{\mathbf{y}}^i = \text{MLP}(\mathbf{E}^L; \theta). \tag{10}$$

where $\hat{\mathbf{y}}^i \in \mathbb{R}^C$ is the predicted label distribution and C is the number of classes. In the training process, we optimize the cross entropy loss of the node i:

$$\mathcal{L} = \textstyle\sum_{i\in\mathcal{V}_{\texttt{train}}} \texttt{CrossEntropy}(\hat{\mathbf{y}}^i, \mathbf{y}^i) \tag{11}$$

where $\mathcal{V}_{\texttt{train}}$ is the training node set, and $\mathbf{y}^i$ is the ground-truth one-hot label vector of node i.

5 Experiments

In this section, we first evaluate PHGT on the node classification task, and then perform ablation studies, parameter analysis, and efficiency studies.

5.1 Experimental Setup

Datasets. Our experiments encompass four widely-used heterogeneous graph datasets: DBLP and ACM (academic networks), IMDB (movie network), and Freebase (knowledge graph). Among these datasets, DBLP, ACM, and IMDB are sourced from a standardized benchmark called the Heterogeneous Graph Benchmark (HGB) [Lv *et al.*, 2021]. The preprocessing and data splitting procedures follow the guidelines established by HGB. For Freebase, we have employed the version provided by a recent state-of-the-art work [Mao *et al.*, 2023]. A summary of the statistics of each dataset is provided in Table 1.

Our proposed semantic tokenization requires a pre-defined meta-path set. For the DBLP dataset, we extract instances of APA, APT and APC (A: Author, P: Paper, T: Term, C: Conference). For the ACM dataset, we extract instances of PAP and PSP (P: Paper, A: Author, S: Subject). For the IMDB dataset, we extract instances of MDM, MAM and MKM (M: Movie, D: Director, A: Actor, K: Keyword). For the Freebase dataset, we extract the instances of MAM, MDM and MWM (M: Movie, A: Actor, D: Director, W: Writer).

Table 1: Statistics of the datasets

Dataset	Nodes	# Node types	#Edges	# Edge types	Target	#Classes
DBLP	26,128	4	239,566	6	author	4
IMDB	21,420	4	86,642	6	movie	5
ACM	10,942	4	547,872	8	paper	3
Freebase	43,854	4	151,034	6	movie	3

Baselines. To comprehensively evaluate the proposed PHGT against state-of-the-art approaches, we consider a variety of baseline models that fall into two major categories. In the first category, we focus on message-passing-based HGNNs. This category encompasses the following models: RGCN [Schlichtkrull *et al.*, 2018], HAN [Wang *et al.*, 2019], GTN [Yun *et al.*, 2019], HetGNN [Zhang *et al.*, 2019], HGT [Hu *et al.*, 2020], MAGNN [Fu *et al.*, 2020], and Simple-HGN [Lv *et al.*, 2021]. The second category involves transformer-based models. Within this category, we consider ANS-GT [Zhang *et al.*, 2022], a coarsening-based model; NodeFormer [Wu *et al.*, 2022], a linear transformer-based model; HINormer [Mao *et al.*, 2023], a model specifically tailored for heterogeneous graphs. We have excluded certain other graph transformer models [Ying *et al.*, 2021; Chen *et al.*, 2022; Hussain *et al.*, 2022] from our experimental results. The reason is that these models are designed for small molecular graphs, and they run out of memory when applied to our datasets.

For the baseline results that were already reported in the original HGB paper [Lv *et al.*, 2021], we directly quote the reported results. For results that were not included in the HGB paper, we conducted our experiments using authors' code and the OpenHGNN library [Han *et al.*, 2022].

Experimental Settings. We performed multi-class node classification experiments on three datasets, namely, DBLP, Freebase and ACM, and multi-label node classification on IMDB. For DBLP, IMDB and ACM, we used the same data split from HGB for training, validation, and testing on each dataset. For the Freebase dataset, we use the split provided by HINormer [Mao *et al.*, 2023]. Classification performance was evaluated using micro-F1 and macro-F1 metrics. For robustness, all experiments were repeated three times, and we report the average results along with their corresponding standard deviations. For detailed hyperparameter settings, please refer to Appendix C.3.

5.2 Performance Evaluation

We present the node classification results in Table 2. Based on the results reported, we make the following observations.

First, our proposed PHGT model demonstrates superior performance in most scenarios, outperforming other baseline models. The only exception is the macro-F1 metric on the Freebase dataset, where HINormer performs the best but the performance of PHGT remains competitive, and PHGT still achieves the best micro-F1 score. This observation suggests that local structures may be relatively more important on the Freebase dataset, which is a knowledge graph constructed from a collection of individual facts or triples. As HINormer limits its receptive field to a local subgraph, it may have an advantage on this dataset.

Second, PHGT consistently outperforms ANS-GT and NodeFormer. This is noteworthy because both ANS-GT and NodeFormer possess a global receptive field, which emphasizes the effectiveness of our proposed semantic token and global token.

Third, PHGT surpasses all the message passing-based HGNN baselines in almost all cases. This finding suggests that the incorporation of a pairwise self-attention mechanism beyond a local receptive field is beneficial in enhancing the performance on heterogeneous graphs.

5.3 Ablation Studies

To assess the effectiveness of each component in our PHGT model, we conduct an ablation study, examining three ablated variants: (1) w/o semantic token: In this variant, the semantic token is removed to gauge its impact on performance; (2) w/o global token: In this variant, the global token is removed to assess its contribution; (3) w/o both: In this variant, both the semantic tokens and the global tokens are removed, retaining only the node tokens.

The micro-F1 results are reported in Table 3. From the results, we observe that removing the semantic tokens causes the performance to drop across all datasets, underscoring the significance of the semantic tokens in capturing heterogeneous high-order relations. A similar trend is observed when global tokens are removed, indicating their effectiveness in capturing long-range dependencies. Finally, when both semantic tokens and global tokens are removed, the performance becomes the lowest, which implies that the two types of tokens complement each other and play their own distinct role in the full PHGT model.

5.4 Parameter Analysis

We assess the impact of three key hyperparameters in PHGT: (1) the number of node tokens, (2) the number of semantic tokens N, and (3) the number of global tokens M (which is equivalent to the number of heterogeneous clusters).

Our experiments are conducted on the DBLP and ACM datasets, and the effect of these hyperparameters are depicted in Figure 2. We observe that the performance trends for the number of node tokens and the number of semantic tokens display a similar pattern. Initially, the performance experiences a notable improvement, which stabilizes afterward. This pattern suggests that with too few tokens, their contribution to performance is limited, while beyond a certain threshold, their impact plateaus. In terms of the number of global tokens, the performance initially improves before declining. This trend might be attributed to the possibility that excessively fine-grained long-range information could introduce additional noise, adversely affecting the overall performance. Overall, opting for 64–128 node tokens and semantic tokens as well as 64 global tokens appears to be an optimal and robust setting.

5.5 Efficiency Studies

First, an idea parallel to our global tokenization method is to use a graph coarsening algorithm to aggregate nodes

Table 2: Performance evaluation (%). Vacant positions ("-") mean that the model runs out of memory on the corresponding dataset.

Methods	DBLP		IMDB		ACM		Freebase	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
RGCN	92.07±0.50	91.52±0.50	62.05±0.15	58.85±0.26	91.41±0.75	91.55±0.74	60.82±1.23	59.08±1.44
HAN	92.05±0.62	91.67±0.49	64.63±0.58	57.74±0.96	90.79±0.43	90.89±0.43	61.42±3.56	57.05±2.06
GTN	93.97±0.54	93.52±0.55	65.14±0.45	60.47±0.98	91.20±0.71	91.31±0.70	-	-
HetGNN	92.33±0.41	91.76±0.43	51.16±0.65	48.25±0.67	86.05±0.25	85.91±0.25	62.99±2.31	58.44±1.99
MAGNN	93.76±0.45	93.28±0.51	64.67±1.67	56.49±3.20	90.77±0.65	90.88±0.64	64.43±0.73	58.18±3.87
HGT	93.49±0.25	93.01±0.23	67.20±0.57	63.00±1.19	91.00±0.76	91.12±0.76	66.43±1.88	60.03±2.21
Simple-HGN	94.46±0.22	94.01±0.24	67.36±0.57	63.53±1.36	93.35±0.45	93.42±0.44	67.49±0.97	62.49±1.69
ANS-GT	93.15±0.51	92.75±0.43	66.65±0.35	62.52±0.61	92.55±0.54	93.67±0.62	67.33±0.61	61.24±0.57
NodeFormer	93.68±0.42	93.05±0.38	65.86±0.42	62.15±0.77	91.89±0.31	92.72±0.84	67.01±0.52	60.83±1.41
HINormer	94.94±0.21	94.57±0.23	67.83±0.34	64.65±0.53	93.15±0.36	93.28±0.43	67.78±0.39	**62.76±1.10**
PHGT	**95.33±0.18**	**94.96±0.17**	**68.81±0.08**	**65.91±0.30**	**93.72±0.40**	**93.79±0.39**	**68.74±1.42**	61.73±1.86

Table 3: Micro-F1 (%) for ablation studies.

	DBLP	IMDB	ACM	Freebase
w/o both	94.80	68.35	93.34	67.58
w/o semantic token	94.94	68.58	93.41	67.73
w/o global token	94.91	68.54	93.55	68.06
PHGT	**95.33**	**68.81**	**93.72**	**68.89**

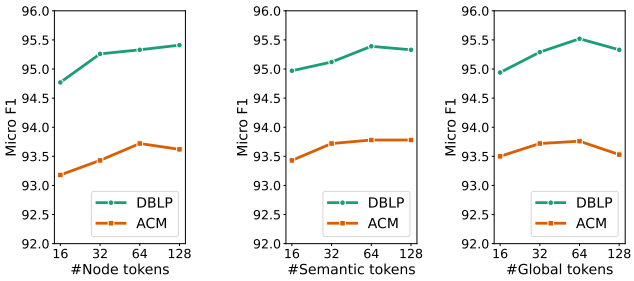
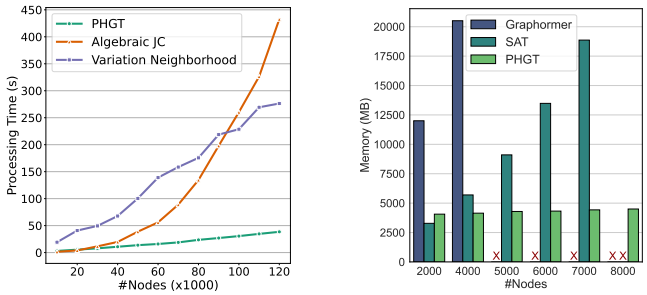


Figure 2: Impact of key hyperparameters.



(a) Time comparison (b) Memory comparison

Figure 3: Efficiency evaluation. × denotes that the model runs out of memory at the corresponding graph size.

into super nodes, and convert these super nodes to transformer tokens. However, graph coarsening methods often involve the computation of spectral features, rendering them less efficient in comparison (besides, existing graph coarsening-based transformers are only designed for homogeneous graphs). To evaluate the efficiency, we conducted a comparison of time costs between our method and two graph coarsening algorithms, namely, Variation Neighborhood [Loukas, 2019] and Algebraic JC [Ron *et al.*, 2011]. As illustrated in Figure 3(a), the time consumption of our method exhibits a drastically slower growth rate as the graph size expands. Notably, when processing a graph of 120,000 nodes, our approach achieves a speedup of about 7–11 times compared to the two graph coarsening methods. This highlights the superior computational efficiency of our method.

Second, we evaluate our advantage in memory efficiency over conventional graph transformer models in which the entire graph is treated as the receptive field. We present a visualization of the memory consumption for PHGT, Graphormer [Ying *et al.*, 2021], and SAT [Chen *et al.*, 2022] in Figure 3(b). Evidently, both conventional graph transformer models run out of memory when the graph size reaches 8,000 nodes, whereas our method continues to exhibit relatively low memory consumption. This demonstrates that our approach is practical beyond small molecular graphs.

6 Conclusion

In this paper, we proposed a novel Poly-tokenized Heterogeneous Graph Transformer (PHGT) to address the two limitations of existing graph transformer models in handling complex heterogeneous graphs: (1) the inability to capture heterogeneous semantics; (2) the incapacity to model intricate long-range dependencies. In PHGT, we introduced two novel token types, namely, semantic tokens and global tokens, in addition to the traditional node tokens. Semantic tokens are designed to encapsulate high-order semantic relations, while global tokens facilitate the incorporation of semantic-aware long-range dependencies. Through comprehensive experiments on four benchmark datasets, we demonstrate the efficacy of our PHGT approach.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62192784, U22B2038, 62002029, 62322203, 62172052), Young Elite Scientists Sponsorship Program (No. 2023QNRC001) by CAST.

References

[Alon and Yahav, 2021] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[Chen *et al.*, 2020] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3438–3445, 2020.

[Chen *et al.*, 2022] Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *International Conference on Machine Learning*, pages 3469–3489, 2022.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[Dosovitskiy *et al.*, 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[Fu *et al.*, 2020] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of The Web Conference 2020*, pages 2331–2341, 2020.

[Gregory, 2010] Steve Gregory. Finding overlapping communities in networks by label propagation. *New journal of Physics*, 12(10):103018, 2010.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

[Han *et al.*, 2022] Hui Han, Tianyu Zhao, Cheng Yang, Hongyi Zhang, Yaoqi Liu, Xiao Wang, and Chuan Shi. Openhgnn: an open source toolkit for heterogeneous graph neural network. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 3993–3997, 2022.

[Hu *et al.*, 2020] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710, 2020.

[Hussain *et al.*, 2022] Md Shamim Hussain, Mohammed J Zaki, and Dharmashankar Subramanian. Global self-attention as a replacement for graph convolution. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 655–665, 2022.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[Kreuzer *et al.*, 2021] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

[Li *et al.*, 2021] Jianxin Li, Hao Peng, Yuwei Cao, Yingtong Dou, Hekai Zhang, S Yu Philip, and Lifang He. Higher-order attribute-enhancing heterogeneous graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):560–574, 2021.

[Li *et al.*, 2022] Mei Li, Xiangrui Cai, Linyu Li, Sihan Xu, and Hua Ji. Heterogeneous graph attention network for drug-target interaction prediction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1166–1176, 2022.

[Li *et al.*, 2023] Chao Li, Zijie Guo, Qiuting He, Hao Xu, and Kun He. Long-range dependency based multi-layer perceptron for heterogeneous information networks. *arXiv preprint arXiv:2307.08430*, 2023.

[Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[Liu *et al.*, 2023] Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S Yu, et al. Towards graph foundation models: A survey and beyond. *arXiv preprint arXiv:2310.11829*, 2023.

[Loukas, 2019] Andreas Loukas. Graph reduction with spectral and cut guarantees. *J. Mach. Learn. Res.*, 20(116):1–42, 2019.

[Lv *et al.*, 2021] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1150–1160, 2021.

[Mao *et al.*, 2023] Qiheng Mao, Zemin Liu, Chenghao Liu, and Jianling Sun. Hinormer: Representation learning on heterogeneous information networks with graph transformer. In *Proceedings of the ACM Web Conference 2023*, pages 599–610, 2023.

[Müller *et al.*, 2023] Luis Müller, Mikhail Galkin, Christopher Morris, and Ladislav Rampášek. Attending to graph transformers. *arXiv preprint arXiv:2302.04181*, 2023.

[Rampášek *et al.*, 2022] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

[Ron *et al.*, 2011] Dorit Ron, Ilya Safro, and Achi Brandt. Relaxation-based coarsening and multiscale graph organization. *Multiscale Modeling & Simulation*, 9(1):407–423, 2011.

[Scarselli *et al.*, 2008] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[Schlichtkrull *et al.*, 2018] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *International Conference on The Semantic Web*, pages 593–607, 2018.

[Shi *et al.*, 2016] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37, 2016.

[Sun *et al.*, 2011] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, 4(11):992–1003, 2011.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018.

[Wang *et al.*, 2019] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

[Wu *et al.*, 2022] Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401, 2022.

[Xu *et al.*, 2019] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[Yang *et al.*, 2020] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. Heterogeneous network representation learning: A unified framework with survey and benchmark. *IEEE Transactions on Knowledge and Data Engineering*, 34(10):4854–4873, 2020.

[Yang *et al.*, 2022] Yaming Yang, Ziyu Guan, Zhe Wang, Wei Zhao, Cai Xu, Weigang Lu, and Jianbin Huang. Self-supervised heterogeneous graph pre-training based on structural clustering. *Advances in Neural Information Processing Systems*, 35:16962–16974, 2022.

[Ying *et al.*, 2021] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021.

[Yun *et al.*, 2019] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[Zhang *et al.*, 2019] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 793–803, 2019.

[Zhang *et al.*, 2022] Zaixi Zhang, Qi Liu, Qingyong Hu, and Chee-Kong Lee. Hierarchical graph transformer with adaptive node sampling. *Advances in Neural Information Processing Systems*, 35:21171–21183, 2022.

[Zhou and Liu, 2013] Yang Zhou and Ling Liu. Social influence based clustering of heterogeneous information networks. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 338–346, 2013.

A The Transformer Layer

Formally, the computation of a transformer layer follows the equations:

$$\begin{aligned} \hat{\mathbf{X}}^l &= \text{Norm}(\text{MHA}(\mathbf{X}^{l-1}) + \mathbf{X}^{l-1}), \\ \mathbf{X}^l &= \text{Norm}(\text{FNN}(\hat{\mathbf{X}}^l) + \hat{\mathbf{X}}^l), \end{aligned} \tag{12}$$

where $\text{Norm}(\cdot)$ denotes the layer-norm function. And $\mathbf{X}^0 = \mathbf{H}^i$ in PHGT.

B Details of the Heterogeneous Graph Clustering Technique

We make a simple modification to the pseudo-label propagation algorithm. We use an attention-based graph neural network to learn the edge weights and use these weights to guide the pseudo-label propagation process. For this purpose, we introduce a simple heterogeneous variant of GAT called GAT-Hete. It calculates the weight $\alpha(i,j)$ for the edge (i,j) by employing a type-wise projection of the node features, as demonstrated in Eq. (7). Then, the embedding of node i can be aggregated by

$$\mathbf{h}_i^{'} = \sigma(\textstyle\sum_{j \in \mathcal{N}_i} \alpha_{i,j} \mathbf{W} \mathbf{h}_j), \tag{13}$$

where σ is a nonlinear function.

By Eq. (7), we can obtain the edge weights $\alpha_{i,j}$ for every edge (i,j) in $\mathcal{G}$, reflecting the importance of different edges in the graph. We use the obtained edge weight information to guide the label propagation process. Since label propagation can be seen as a type of message passing, we write the label propagation process in a message passing manner:

$$l_i = \sigma(\textstyle\sum_{j \in \mathcal{N}_i} \alpha_{i,j} l_j), \tag{14}$$

where $\alpha_{i,j}$ is directly acquired from the GAT-Hete model and l_i is the pseudo-label of node i. To train the GAT-Hete model, we follow the approach outlined by [Yang *et al.*, 2022]. Initially, we perform random initialization of the pseudo-labels as $\mathbf{L} = [l_0, l_1, ..., l_{|\mathcal{V}|}]^\top$. Next, we utilize the GAT-Hete model to predict these pseudo-labels $\mathbf{L}$ and calculate edge weights for each edge in the set $\mathcal{E}$. Subsequently, we execute label propagation based on Eq. (14). After updating the values in $\mathbf{L}$, we retrain the GAT-Hete model to predict the updated pseudo-labels $\mathbf{L}$. This iterative process continues until the pseudo-labels $\mathbf{L}$ converge. The final $\mathbf{L}$ indicates the cluster of every node in $\mathcal{G}$.

The pseudo-code for our proposed heterogeneous graph clustering technique is provided in Algorithm 1.

Algorithm 1 Heterogeneous Graph Clustering

Input: A heterogeneous graph $\mathcal{G}$.
Output: The pseudo-label $\mathbf{L}$.

```
Randomly initialize L.
while L not converge do
    Train GAT-Hete to fit L
    Update the edge weight according to Eq. (7)
    Label propagation according to Eq. (14)
end while
```

C Experiment Details

C.1 Details of Datasets

We employ a comprehensive collection of four real-world heterogeneous graph datasets, encompassing both academic and movie-related networks. Specifically, the academic datasets comprise DBLP and ACM, while the movie-related datasets encompass IMDB and Freebase.

- **DBLP** constitutes a bibliographic dataset in the realm of computer science, encompassing papers disseminated from 1994 to 2014 across 20 conferences within four distinct research domains. The dataset encompasses four primary node types: authors (A), papers (P), terms (T), and venues (V). We have directly employed the partitioned dataset as provided by Heterogeneous Graph Benchmark (HGB) [Lv *et al.*, 2021].
- **IMDB** is an online platform dedicated to movies and associated information. The sole multi-label dataset consists of movies categorized into Action, Comedy, Drama, Romance, and Thriller genres. This dataset encompasses four primary node types: movies (M), directors (D), actors (A), and keywords (K). We use the split provided by HGB.
- **ACM** is a bibliographic dataset containing papers published in renowned conferences such as KDD, SIGMOD, SIGCOMM, MobiCOMM, and VLDB. This heterogeneous graph encompasses 3025 papers (P), 5835 authors (A), and 56 subjects (S). For our analysis, we rely on the partitioning provided by HGB.
- **Freebase** constitutes a subset of the broader Freebase knowledge graph, encompassing entities categorized into four distinct types: movies (M), actors (A), directors (D), and writers (W). In our study, we adopt the partitioning scheme provided by [Mao *et al.*, 2023].

C.2 Details of Baselines

In order to conduct a comprehensive comparison between our PHGT model and state-of-the-art counterparts, we incorporate baseline models from two primary categories: (1): message passing-based HGNNs; (2): transformer-based graph models.

(1) Message passing-based HGNNs

- **RGCN** [Schlichtkrull *et al.*, 2018] employs a series of distinct weight matrices to project node embeddings into various relation spaces, capturing the inherent heterogeneity within the graph.
- **HAN** [Wang *et al.*, 2019] represents a pioneering effort in extending GNNs to accommodate heterogeneous graphs. It introduces a novel hierarchical attention mechanism that adeptly captures both the structural and semantic aspects of the graph.
- **GTN** [Yun *et al.*, 2019] devises an aggregation function capable of autonomously identifying appropriate meta-paths throughout the message passing procedure.

- **HetGNN** [Zhang *et al.*, 2019] initially employs random walks with restart to generate neighboring nodes, followed by the utilization of a Bi-LSTM network to aggregate node features within and across different types.
- **MAGNN** [Fu *et al.*, 2020] builds upon the foundation of HAN and extends its methodology by incorporating all nodes within a meta-path instance, as opposed to solely considering nodes at the two endpoints.
- **HGT** [Hu *et al.*, 2020] use a transformer-like heterogeneous attention mechanism to aggregate neighbor information, which still falls into the message passing framework. The approach employs type-specific parameters to define heterogeneous attention patterns along each edge.
- **Simple-HGN** [Lv *et al.*, 2021] introduces a straightforward yet powerful baseline model based on GAT. This model incorporates both the edge type embedding and node embeddings to compute the attention scores.

(2) Transformer-based graph models

- **ANS-GT** [Zhang *et al.*, 2022] innovates by adapting a multi-armed bandit algorithm to enable adaptive node sampling. Furthermore, it introduces a hierarchical graph attention scheme, wherein coarse-grained attention is accomplished through graph coarsening.
- **NodeFormer** [Wu *et al.*, 2022] employs a kernelized Gumbel-Softmax operator to approximate the otherwise quadratic complexity inherent in the standard self-attention mechanism. Additionally, it incorporates a relation bias to integrate graph structure into the model.
- **HINormer** [Mao *et al.*, 2023] is a transformer-based model designed for heterogeneous graphs. It employs a local structure encoder alongside a heterogeneous relation encoder to capture both the structural and heterogeneous information.

C.3 Hyperparameter Settings

For the baseline models, in the cases of the DBLP, IMDB, and ACM datasets, we have maintained an identical experimental settings as presented in HGB [Lv *et al.*, 2021]. Consequently, we have directly incorporated the baseline results published in HGB. For the Freebase dataset, we have adopted the default hyperparameter settings as presented in the original paper for our experiment. For PHGT, we use Simple-HGN as the structure encoder.

We report the hyperparameter adopted by PHGT in Table 4.

Experiments are conducted on a server with:

- GPU: GeForce RTX 3090;
- CPU: Intel(R) Xeon(R) Gold 6348 CPU @ 2.60GHz;
- Operating system: Ubuntu 18.04.5 LTS.

Table 4: Hyperparameter Settings for PHGT. NL denotes the number of node tokens, HD represents the hidden dimension, and LR signifies the learning rate.

Dataset	NL	N	M	L	HD	dropout	LR
DBLP	64	256	256	2	128	0.5	1e-4
IMDB	64	32	64	3	256	0.4	1e-4
ACM	256	64	64	2	64	0.5	1e-4
Freebase	128	128	320	2	64	0.5	1e-4