HHGT: Hierarchical Heterogeneous Graph Transformer for Heterogeneous Graph Representation Learning

Qiuyu Zhu
Nanyang Technological University
Singapore
qiuyu002@e.ntu.edu.sg

Liang Zhang*†
HEC Paris
France
zhangl@hec.fr

Qianxiong Xu
Nanyang Technological University
Singapore
qianxion001@e.ntu.edu.sg

Kaijun Liu
Nanyang Technological University
Singapore
kaijun001@e.ntu.edu.sg

Cheng Long*
Nanyang Technological University
Singapore
c.long@ntu.edu.sg

Xiaoyang Wang
University of New South Wales
Australia
xiaoyang.wang1@unsw.edu.au

Abstract

Despite the success of Heterogeneous Graph Neural Networks (HGNNs) in modeling real-world Heterogeneous Information Networks (HINs), challenges such as expressiveness limitations and over-smoothing have prompted researchers to explore Graph Transformers (GTs) for enhanced HIN representation learning. However, research on GT in HINs remains limited, with two key shortcomings in existing work: (1) A node's neighbors at different distances in HINs convey diverse semantics; for instance, a paper's direct neighbor (a paper) in an academic graph signifies a citation relation, whereas the indirect neighbor (another paper) implies a thematic association, reflecting distinct meanings. Unfortunately, existing methods ignore such differences and uniformly treat neighbors within a given distance in a coarse manner, which results in semantic confusion. (2) Nodes in HINs have various types, each with unique semantics, e.g., papers and authors in an academic graph carry distinct meanings. Nevertheless, existing methods mix nodes of different types during neighbor aggregation, hindering the capture of proper correlations between nodes of diverse types. To bridge these gaps, we design an innovative structure named (k, t)-ring neighborhood, where nodes are initially organized by their distance, forming different non-overlapping k-ring neighborhoods for each distance. Within each k-ring structure, nodes are further categorized into different groups according to their types, thus emphasizing the heterogeneity of both distances and types in HINs naturally. Based on this structure, we propose a novel **H**ierarchical **H**eterogeneous **G**raph **T**ransformer (HHGT) model, which seamlessly integrates a Type-level Transformer for aggregating nodes of different types within each k-ring neighborhood, followed by a Ring-level Transformer for aggregating different k-ring neighborhoods in a hierarchical manner. Extensive experiments are conducted on downstream tasks to verify HHGT's superiority over **14** baselines, with a notable improvement of up to 24.75% in NMI and 29.25% in ARI for node clustering task on the ACM dataset compared to the best baseline.

CCS Concepts

• **Computing methodologies** → **Learning latent representations**; • **Information systems** → **Data mining**.

Keywords

heterogeneous information network, node representation learning, hierarchical heterogeneous graph Transformer

ACM Reference Format:
Qiuyu Zhu, Liang Zhang, Qianxiong Xu, Kaijun Liu, Cheng Long, and Xiaoyang Wang. 2025. HHGT: Hierarchical Heterogeneous Graph Transformer for Heterogeneous Graph Representation Learning. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining (WSDM '25), March 10–14, 2025, Hannover, Germany*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3701551.3703511

1 Introduction

Heterogeneous Information Networks (HINs) [26], also well-known as Heterogeneous Graphs (HGs), consist of multiple types of objects (i.e., nodes) and relations (i.e., edges). They are prevalent in real-world scenarios, ranging from citation networks [13, 30], social networks [2, 17] to recommendation systems [3, 44]. For example, the academic data shown in Figure 1(a) can be represented as an HIN, which contains three types of nodes (i.e., paper, author, subject) and three types of relations (i.e., author-write-paper, paper-belong-subject, paper-cite-paper). Recently, there has been a notable surge in research focusing on representation learning for HINs [8, 10, 24, 25], which emerges as a powerful technique for embedding nodes into low-dimensional representations while retaining both graph structures and heterogeneity.

Given the success of traditional Graph Neural Networks (GNNs) [7, 13, 17] in handling homogeneous graphs (containing only one type of nodes and relations), researchers are increasingly turning their attention to HIN representation learning using GNNs, known as Heterogeneous Graph Neural Networks (HGNNs). HGNN-based approaches [11, 19] often leverage neighbor aggregation strategies to effectively capture and propagate information across diverse types of nodes in HINs. For example, R-GCN [23] extends the traditional Graph Convolutional Networks (GCNs) [17] by incorporating relation-specific weight matrices, capturing diverse relations within

*Co-corresponding authors.
†The work was done while Liang Zhang was a PhD student at NTU.

WSDM '25, March 10–14, 2025, Hannover, Germany

ACM ISBN 979-8-4007-1329-3/25/03
https://doi.org/10.1145/3701551.3703511

an HIN. Fu et al. [11] propose to incorporate intermediate nodes along meta-paths, using both intra-meta-path and inter-meta-path information for higher-order semantic information aggregation.

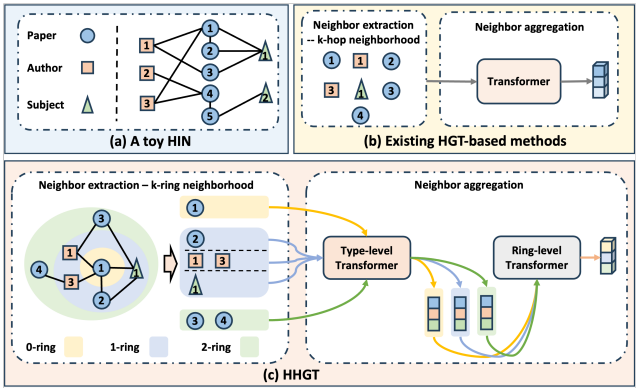


Figure 1: The difference between existing HGT-based methods and our HHGT model: (a) A toy HIN; (b) Learning node representations through existing HGT-based methods; (c) Learning node representations through our HHGT model. Here, node P_1's 0-ring neighborhood, 1-ring neighborhood, 2-ring neighborhood are visually highlighted with yellow, blue, green colors, respectively. Within the 1-ring neighborhood, nodes of different types are separated by dashed lines.

Despite HGNNs have achieved success in modeling real-world HINs, the presence of challenges such as limitations in expressiveness [35], over-smoothing [5] and over-squashing [1] has driven researchers to investigate Graph Transformers (GTs) [40] for enhanced HIN representation learning. For instance, Hu et al. [14] propose a heterogeneous Transformer-like attention architecture for neighbor aggregation. Mao et al. [21] leverage a local structure encoder and a heterogeneous relation encoder to capture structure and heterogeneity information in HINs. In general, existing GT-based methodologies for HIN representation learning, i.e., HGT-based methods, depicted in Figure 1(b), adhere to a typical principle: Given a target node, its k-hop neighborhood (i.e., those nodes within a reachable distance of $\leq k$ from the target node) is first extracted. Then, Transformer [29] would be utilized to propagate information from these nodes to the target node.

Nevertheless, existing HGT-based approaches tend to mix nodes of different types and uniformly treat all nodes within k-hop neighborhood during neighbor aggregation, leading to potential semantic confusion. In particular, (1) **Limitation 1:** Neighbors of a target node at different distances in HINs carry varied semantics. Using Figure 1(a) as an illustration, paper P_1's direct neighbor, paper P_2, indicates a citation relation. Conversely, the indirect neighbor, paper P_3, implies a thematic connection without a direct citation relation, showcasing different connotations. Regrettably, existing strategies overlook such distinctions by uniformly addressing each neighbor within distance k, i.e., packing P_1, P_2, P_3 together into a single sequence and aggregating them uniformly. This is not desirable since these nodes serve different functions. (2) **Limitation 2:** Neighbors of a target node with different types also carry distinct semantics. Taking Figure 1(a) as an instance, paper P_1's direct neighbors include paper P_2, authors A_1, A_3 and subject S_1. Here, P_2 represents a citation relation, A_1, A_3 reflect authorship relations, while S_1 signifies a topic alignment relation. While existing HGT-based methods consider node types, they typically pack P_2, A_1, A_3, S_1 together as a unified sequence. This approach is not desirable because it mixes nodes of different types during neighbor aggregation, blurring the distinct functions of papers, authors, and subjects.

To overcome these challenges, we propose the following two main designs: (1) **Design 1:** To distinguish a node's neighbors at varying distances, we introduce an innovative structure called the k-ring neighborhood. This structure specifically refers to nodes whose distance from the target node is exactly k, differentiating it from the commonly known k-hop neighborhood. In essence, we split the k-hop neighborhood into $k+1$ non-overlapping k-ring neighborhoods, where the nodes in each k-ring neighborhood share the same distance to the target node. As illustrated in Figure 1(c), considering $k = 2$, for paper P_1, its neighbors within a distance of 2 can be decomposed into three distinct k-ring neighborhoods: the 0-ring neighborhood $\{P_1\}$, the 1-ring neighborhood $\{P_2, A_1, A_3, S_1\}$, and the 2-ring neighborhood $\{P_3, P_4\}$. Building upon this new structure, we extract diverse k-ring neighborhoods for each node, which can naturally discern different functions and thus preventing semantic confusion. Then, a Ring-level Transformer is designed to aggregate distinct k-ring neighborhoods separately, with aggregation based on the relevance and significance of each k-ring neighborhood to the target node. (2) **Design 2:** To avoid mixing nodes of different types within each k-ring structure, we further propose a novel (k, t)-ring structure by arranging nodes into different groups based on their types within each k-ring structure. Based on such neighborhood partition, a Type-level Transformer is proposed to separately aggregate neighbors of distinct types for a target node within each k-ring structure, considering the importance of each type to the target node. In Figure 1(a), consider the 1-ring neighborhood of node P_1 (i.e., P_2, A_1, A_3, S_1), where nodes of diverse types coexist. We partition this 1-ring neighborhood into three groups based on node types, namely, paper P_2, authors A_1, A_3, and subject S_1, with each group carrying unique functions. Then, we apply a Type-level Transformer to aggregate each group separately, rather than treating them as a unified sequence as done by existing HGT-based methods. This approach enables us to mimic the diverse roles of nodes with various types.

In summary, for each target node, we extract its neighbors from diverse k-ring neighborhoods, where the nodes within each ring are further grouped according to their types, forming an innovative (k, t)-ring neighborhood structure. Building upon this structure, we introduce a novel **H**ierarchical **H**eterogeneous **G**raph **T**ransformer (HHGT) model. This model seamlessly integrates a Type-level Transformer for aggregating nodes of different types within each k-ring neighborhood separately, followed by a Ring-level Transformer for aggregating different k-ring neighborhoods in a hierarchical manner. The main contributions of our paper are summarized as follows:

- For the first time, we design an innovative (k, t)-ring neighborhood structure for HIN representation learning, emphasizing the heterogeneity of both distances and types in HINs naturally.
- To the best of our knowledge, we are the first to propose a hierarchical graph transformer model for node representation learning in HINs, which seamlessly integrates a Type-level Transformer

for aggregating nodes of distinct types within each k-ring structure separately, followed by hierarchical aggregation utilizing a Ring-level Transformer for different k-ring neighborhoods.
- Experimental results on two real-world HIN benchmark datasets demonstrate that our HHGT significantly outperforms **14** baseline methods on two typical downstream tasks. Additionally, the ablation study validates the advantages and significance of considering the heterogeneity of both distances and types in HINs.

2 Related Work

2.1 Shallow Models for HIN Embedding

In recent years, a plethora of graph embedding techniques [12, 13] have emerged with the goal of mapping nodes or substructures into a low-dimensional space, preserving the connecting structures within the graph. As real-world networks typically consist of various types of nodes and relations [26], research on shallow models for HIN embedding [31, 36] has garnered significant attention. Shallow models for HIN embedding can be broadly classified into random walk-based methods [8, 24] and first/second-order proximity-based methods [10, 25, 27, 43]. For instance, Metapath2vec [8] adopts meta-path guided random walk to acquire the semantic information between pairs of nodes. These methods leverage meta-paths or type-aware network closeness constraints to exploit network heterogeneity for HIN embedding. Despite their contributions, these shallow models lack the ability to effectively capture intricate relations and semantics within HINs, resulting in suboptimal representation learning.

2.2 Deep Models for HIN Embedding

As deep learning models have shown remarkable success in capturing both structural and content information within homogeneous graphs [4, 16, 18, 29, 34], research has shifted focus to HINs, giving rise to deep models for HINs [20, 36]. Deep models for HIN embedding are broadly categorized into two types: meta-path-based deep models [11, 15, 32, 37, 41, 47] and meta-path-free deep models [14, 19, 21, 23, 46, 48]. Meta-path-based deep models employ meta-paths to aggregate information from type-specific neighborhoods, offering the advantage of capturing higher-order semantic information dictated by selected meta-paths. For example, HAN [32] leverages a hierarchical attention mechanism, which considers both node-level attention and semantic-level attention to learn the importance of nodes and meta-paths, respectively. However, these approaches require expert knowledge for meta-path selection, posing a significant impact on model performance. For meta-path-free strategies, Schlichtkrull et al. [23] propose to model relational data through relation-aware graph convolutional layers, enabling robust representation learning in HINs without meta-paths. Hu et al. [14] introduce an attention mechanism inspired by Transformers, specifically designed for neighbor aggregation. Despite eliminating handcrafted meta-paths, meta-path-free deep models exhibit two key shortcomings: (1) They mix nodes of different types during neighbor aggregation, resulting in a failure to adequately capture the correlations between nodes of different types. (2) They ignore the fact that a node's neighbors at different distances in HINs carry distinct semantics and thus treating them uniformly during neighbor aggregation, which may lead to semantic confusion and suboptimal performance on downstream tasks.

3 Preliminaries

3.1 Problem Definition

Definition 3.1. Heterogeneous Information Network [26]. A heterogeneous information network (HIN) is formally defined as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, C, \mathcal{R}\}$, where $\mathcal{V}$, $\mathcal{E}$, C, $\mathcal{R}$ represent the set of nodes, edges, node types, and relation types, respectively. In an HIN, each node is associated with a node type in C and each edge has its corresponding relation type in $\mathcal{R}$. Each node $v \in \mathcal{V}$ is associated with a feature vector $f \in \mathbb{R}^d$, where d is the feature dimension. An HIN is characterized by the condition $|C| + |\mathcal{R}| > 2$.

Definition 3.2. HIN Representation Learning Problem [31]. Given an HIN, we aim to learn the node embedding $z_v \in \mathbb{R}^d$ for each node v, where $d \ll |\mathcal{V}|$ denotes the embedding dimension.

After learning the node representation of HINs, we can use the embeddings obtained for many downstream tasks, including semi-supervised node classification, unsupervised node clustering, etc.

3.2 Transformer Encoder

Transformer encoder, a core component of Transformer [29], consists of multiple identical layers, each containing two main submodules: the Multi-Head Self-Attention (MSA) module and the Feed-Forward Network (FFN) module. Both components incorporate residual connections and Layer Normalization (LN). To simplify the explanation, we just focus on the single-head self-attention module. Given an input sequence $\mathcal{H} \in \mathbb{R}^{n \times d}$ where n denotes the token number and d denotes the hidden dimension, MSA firstly projects it to query, key and value spaces (namely Q, K, V, respectively), which are written as:

$$Q = \mathcal{H}W_q, K = \mathcal{H}W_k, V = \mathcal{H}W_v, \tag{1}$$

where $W_q \in \mathbb{R}^{d \times d_k}$, $W_k \in \mathbb{R}^{d \times d_k}$, $W_v \in \mathbb{R}^{d \times d_v}$ are learnable matrices. Then, it calculates attention scores by taking the dot product of Q and the transpose of K, normalized by the scaling factor $\sqrt{d_k}$:

$$MSA(\mathcal{H}) = Softmax(\frac{QK^T}{\sqrt{d_k}})V. \tag{2}$$

The MSA output is passed through FFN with an LN and a residual connection to generate the output of the l-th Transformer layer as:

$$\begin{aligned} \tilde{\mathcal{H}}^{(l)} &= MSA(LN(\mathcal{H}^{(l-1)})) + \mathcal{H}^{(l-1)}, \\ \mathcal{H}^{(l)} &= FFN(LN(\tilde{\mathcal{H}}^{(l)})) + \tilde{\mathcal{H}}^{(l)}. \end{aligned} \tag{3}$$

Here, $l = 1, \ldots, L$ represents the l-th layer of the Transformer.

4 Methodology

In this section, we present the details of HHGT model, consisting of two important modules: Ring2Token and TRGT. The overall framework is depicted in Figure 2(a). Given an HIN and an integer K, for each target node, we initially utilize Ring2Token to extract multiple k-ring neighborhoods ($k \in [0, K]$), spanning from 0-ring neighborhood to K-ring neighborhood, with well-organized nodes partitioned by their types within each k-ring structure, forming the

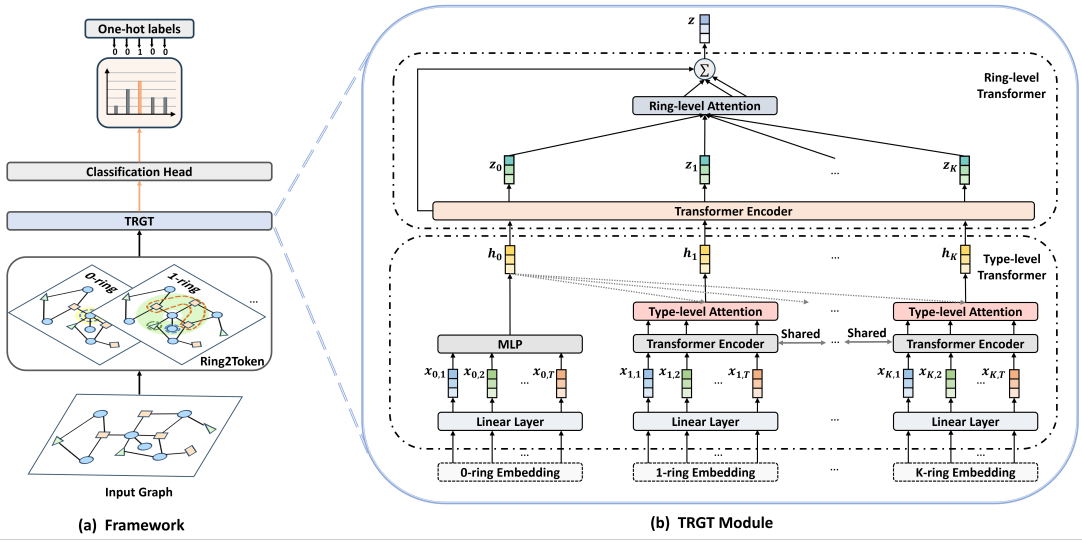


Figure 2: (a) Illustration of the framework for node classification task. (b) Diagram of the TRGT module incorporating both Ring-level Transformer and Type-level Transformer.

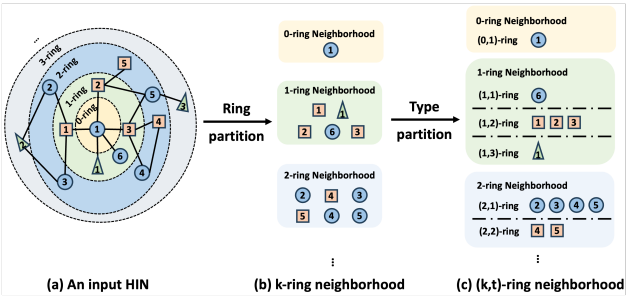


Figure 3: Neighborhood partitioning by Ring2Token.

(k,t)-ring neighborhood structure. After the neighborhood partition, we use the TRGT module to learn node representations via the GT layer based on these extracted (k,t)-ring neighborhoods. This involves a Type-level Transformer to aggregate nodes of different types within each k-ring neighborhood, followed by a Ring-level Transformer to aggregate different k-ring neighborhoods hierarchically. After obtaining representations for all nodes via our HHGT model, following previous work [19], we apply the classification head to transform these node representations into the classification results and train HHGT using the cross-entropy loss function. The details of Ring2Token and TRGT modules are discussed as follows.

4.1 Ring2Token

How to effectively aggregate information from neighbors into a node is critical for designing a powerful HIN representation learning model [11, 14, 32]. However, existing methods overlook the distinctions between neighbors at different distances and mix nodes of distinct types during neighbor aggregation.To address this limitation, we introduce Ring2Token, which considers neighbor information involving different node types at distinct distances. To grasp the distinctions between neighbors at different distances, we first design the novel k-ring neighborhood structure as follows.

Definition 4.1. k-ring Neighborhood. Given a node u, suppose $\Gamma_k(u) = \{v \in \mathcal{V} | d(u,v) = k\}$ denote the k-ring neighborhood of u, where $d(u,v)$ refers to the shortest path distance between nodes u and v. The 0-ring neighborhood is the target node, i.e., $\Gamma_0(u) = \{u\}$. Taking Figure 3(a) as an example where paper P_1 is the target node, and paper P_6 is its 1-ring neighbor, while paper P_2, P_3, P_4, P_5 are its 2-ring neighbors. Here, paper P_6 signifies a citation relation and paper P_2, P_3, P_4, P_5 imply co-authorship associations. Using the 2-hop (including P_2, P_3, P_4, P_5, P_6) mix nodes at different distances, failing to distinguish different functions associated with paper P_6 and P_2, P_3, P_4, P_5. In contrast, the k-ring separates the 2-hop neighbors into different subsets as P_6 and P_2, P_3, P_4, P_5, which naturally discerns different functions and thus prevents semantic confusion.

Meanwhile, each node type carries specific information, embodying distinct concepts. For instance, in Figure 3(b), the 1-ring neighborhood of paper P_1 involves nodes of three different types: paper P_6, authors A_1, A_2, A_3, and subject S_1, where papers offer citation relations, authors contribute to the creation of the paper while subjects provide thematic information. Therefore, it is not suitable to mix nodes of all types together during neighbor aggregation. Motivated by this, we introduce the concept of type-aware k-ring neighborhood (named (k,t)-ring neighborhood) to categorize nodes within each k-ring structure by their types.

Definition 4.2. (k,t)-ring Neighborhood. Given a node u, let $\Gamma_{k,t}(u) = \{v \in \mathcal{V} | v \in \Gamma_k(u) \wedge C_v = t\}$ denote the (k,t)-ring neighborhood of u, where $\Gamma_k(u)$ refers to u's k-ring neighborhood and C_v denotes v's node type.

Based on the concept of (k,t)-ring, we can further partition the 1-ring neighborhood of P_1 into three different subsets as $\Gamma_{1,1}(P_1) = \{P_6\}, \Gamma_{1,2}(P_1) = \{A_1, A_2, A_3\}, \Gamma_{1,3}(P_1) = \{S_1\}$. Then, we can aggregate them separately, enabling us to mimic their distinct roles and avoid mixing different types.

To sum up, given an HIN, for each node, Ring2Token extracts and partitions all its neighborhoods with a (k,t)-ring structure. Specifically, given an integer K and node type number T, for a node

u, it possesses a sequence of k-ring neighborhoods with length $K+1$. Within each k-ring neighborhood, it can be further divided into a series of (k,t)-ring neighborhoods with a length of T. These (k,t)-ring sets will be fed into the TRGT module for model training.

4.2 TRGT Module

Built upon this innovative (k,t)-ring structure, TRGT module seamlessly integrates a Type-level Transformer for aggregating nodes of different types within each k-ring neighborhood, followed by a Ring-level Transformer for aggregating different k-ring neighborhoods in a hierarchical manner. The diagram of TRGT is shown in Figure 2(b) and the details are elaborated below.

4.2.1 Type-level Transformer. Recall that nodes in HINs come in various types, each representing distinct concept. However, existing HGT-based methods tend to mix nodes of different types by packing all node types into a single sequence and uniformly employing attention over them during neighbor aggregation, failing to model distinct roles of nodes with various types as discussed before.

To overcome this limitation, we design a Type-level Transformer to aggregate neighbors by explicitly considering node type differences. Particularly, given a node u and its neighbors, we first adopt the Ring2Token module to divide these neighbors into several subsets, i.e., (k,t)-ring sets. Within each k-ring structure, a series of (k,t)-ring neighborhoods with a total length of T are extracted. For instance, in Figure 3(c), three (k,t)-ring sets are formed within P_1's 1-ring neighborhood. Here, nodes in the same (k,t)-ring are associated with the same node type and semantic function. Therefore, for each (k,t)-ring set, the features of all nodes within it are firstly aggregated using an average pooling function to create an embedding token with a specific size d, i.e., $x_{k,t} \in \mathbb{R}^d$, which explicitly summarizes the information of all nodes with type t within the k-ring. In case of an empty set, the embedding token is filled with zeros, ensuring a consistent size of d. Thus, each k-ring neighborhood can be represented as a sequence of tokens denoted as $x_k = \{x_{k,1}, \ldots, x_{k,T}\} \in \mathbb{R}^{T\times d}$. Then, we aggregate representations of different subsets by adopting a Type-level Transformer encoder over the sequence x_k, as discussed in Section 3.2. Through this type-aware aggregation, our model can explicitly distinguish neighbors with different types and avoid potential semantic confusion. Note that for the 0-ring feature x_0, we employ a Multi-Layer Perceptron (MLP) to convert the embedding dimension from $\mathbb{R}^{T\times d}$ to $\mathbb{R}^{1\times d}$.

By stacking L Transformer blocks, we derive the final representation for each k-ring neighborhood using a read-out function, denoted as $\{h_0, h_1, \ldots, h_K\}$, where $h_0 \in \mathbb{R}^{1\times d}$ and $h_k = \{h_{k,1}, \ldots, h_{k,T}\} \in \mathbb{R}^{T\times d}$.

Type-level Attention Mechanism. We further introduce a type-level attention mechanism as the read-out function within each k-ring structure. The attention function maps a query to key-value pairs, yielding an output. Specifically, the type-level attention function within each k-ring can be defined as follows:

$$\alpha_{k,t} = \frac{exp\left(h_0 \cdot h_{k,t}\right)}{\sum_{i=1}^{T} exp\left(h_0 \cdot h_{k,i}\right)}. \tag{4}$$

Here, $h_0 \in \mathbb{R}^{1\times d}$ denotes the 0-ring representation and $h_{k,t} \in \mathbb{R}^{1\times d}$ denotes the (k,t)-ring representation after Transformer encoder. $\alpha_{k,t} \in \mathbb{R}^1$ is an attention score, T denotes the number of node types and $\cdot$ denotes the dot product. The final representation of each k-ring neighborhood is calculated as:

$$h_k = \sum_{t=1}^{T} \alpha_{k,t} \cdot h_{k,t}. \tag{5}$$

Finally, Type-level Transformer outputs a sequence of k-ring representations for each node, which later are forwarded into the Ring-level Transformer for representation learning.

4.2.2 Ring-level Transformer. Each k-ring neighborhood contributes a new layer of information, and their combination provides diverse perspectives, essential for a comprehensive understanding of the HIN. Thus, the effective collection of information from these k-ring neighborhoods is crucial. To tackle this challenge, we develop the Ring-level Transformer for global aggregation across different k-ring neighborhoods. For each node, given a sequence of k-ring tokens $\{h_0, h_1, \ldots, h_K\}$ obtained from Type-level Transformer, with each summarizing the unique information of neighbors belonging to a specific k-ring structure, Ring-level Transformer first leverages Transformer encoder to learn the node representations. After stacking L Transformer layers, we obtain the representation of each node, i.e., a sequence of representations $\{z_0, z_1, \ldots, z_K\}$ where $z_k \in \mathbb{R}^d$.

Ring-level Attention Mechanism. Considering the potential diverse and unique impacts of different k-ring neighborhoods, we design the ring-level attention mechanism for information aggregation. Specifically, it calculates attention coefficients by assessing the relations between the 0-ring neighborhood (the node itself) and all other k-ring neighborhoods, which is formulated as:

$$\alpha_k = \frac{exp\left((z_0||z_k)W^T\right)}{\sum_{i=1}^{K} exp\left((z_0||z_i)W^T\right)}, \tag{6}$$

where $W \in \mathbb{R}^{1\times 2d}$ denotes the learnable projection, and $||$ indicates the concatenation operator. Once the attention scores are obtained, they are employed to calculate a linear combination of the corresponding representations, which is written as:

$$z = z_0 + \sum_{k=1}^{K} \alpha_k \cdot z_k, \tag{7}$$

where K is the number of rings, z_0 and z_k denote the representations of 0-ring and k-ring neighborhood, respectively. Then, we can derive the final representation of each node as $z \in \mathbb{R}^d$.

4.3 Objective Function

After obtaining the final representations of all nodes through our HHGT model, we employ the cross-entropy loss to optimize node embeddings in HINs following existing work [32]. Specifically, we utilize a classification head to predict the labels of nodes. This prediction results in a predicted label matrix for nodes, denoted by $\hat{Y} \in \mathbb{R}^{n\times|\mathcal{L}|}$, where $|\mathcal{L}|$ denotes the number of classes. The cross-entropy loss employed is then described as follows:

$$\mathcal{L} = -\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{L}} Y_{i,j} ln\hat{Y}_{i,j}. \tag{8}$$

Here, $\mathcal{I}$ and $Y_{i,j}$ denote the labeled node set and the true label, respectively. The model is refined through back-propagation to

learn the node representations. The time complexity of HHGT is $O(n(K+1)^2d + n(K+1)T^2d)$, where n, K, d, T denote the node number, ring number, feature dimension, type number, respectively. Details are in Appendix A of the supplementary material.

5 Experiments

We conduct experiments to answer following research questions:

- **RQ1:** Can HHGT outperform baselines across downstream tasks?
- **RQ2:** What do the learned node embeddings represent? Can they capture the intricate structures and heterogeneity within HINs?
- **RQ3:** How do different modules of HHGT contribute to enhancing the model performance?
- **RQ4:** How do hyper-parameters affect HHGT's performance?

5.1 Experimental Settings

5.1.1 Datasets. We conduct experiments on two publicly real-world HIN benchmark datasets (i.e. ACM[1] and MAG[2]), which are widely employed in related works [22, 32, 33, 38]. The statistics of datasets are summarized in Table 1, with details in Appendix B.

5.1.2 Baselines. To verify the effectiveness of our model, we compare our HHGT with two groups of baselines: shallow model-based methods and deep model-based methods. The former group includes PTE [27], ComplEx [28], HIN2Vec [10], M2V [8], AspEm [25]. The latter group can be further divided into two categories: (1) HGNN-based models, including R-GCN [23], HAN [32], AGAT [19], SHGP [39], GTN [41], FastGTN [42]; (2) GT-based models, comprising one homogeneous GT-based method NAGphormer [6], and two heterogeneous GT-based methods, HGT [14] and HINormer [21]. Here, SHGP, PTE, ComplEx, HIN2Vec, M2V and AspEm are unsupervised methods, while R-GCN, HAN, AGAT, GTN, FastGTN, HGT, HINormer and NAGphormer are semi-supervised methods, which are the same as our model. To ensure reproducibility, we include our source code, datasets, as well as the instructions to the selected baselines, in a repository[3]. More details are in Appendix C.

5.2 Node Classification (RQ1)

Settings. Node classification task aims to assign categories to nodes within a network. Following [14], we train a separate linear Support Vector Machine (LinearSVC) [9] with 80% of the labeled nodes and predict on the remaining 20% data. We repeat the process 10 times and report the average results. Micro-F1 and Macro-F1 scores are employed to evaluate the effectiveness following [21, 32, 36].

Results. The overall experimental results are shown in Table 2. As observed, HHGT achieves the best overall performance[§], which indicates its superior effectiveness. The main reasons for the observed improvement may include: (1) The Type-level Transformer optimally utilizes node type information during neighbor aggregation, enabling the effective capture of proper correlations between nodes of different types; (2) The Ring-level Transformer considers the distinctions between neighbors at different distances, facilitating powerful neighbor aggregation across various hierarchical levels. Additionally, we observe that the second best is relatively unstable, demonstrating that our model is very robust against various settings as well as metrics. Among the baselines, we can observe deep model-based approaches generally outperform their shallow model-based counterparts, highlighting the benefits of multi-layered feature extraction in capturing complex information within HINs.

5.3 Node Clustering (RQ1)

Settings. Node clustering task seeks to group nodes in a network, according to common structural and attribute features. Following [22, 32, 39, 45], we adopt an unsupervised learning set where the input is solely node embeddings, and we leverage K-Means to cluster nodes based on their representations generated by the model. The number of clusters for K-Means is set as the category number, and we utilize the same ground truth as employed in node classification task. NMI and ARI are utilized as evaluation metrics here, following [32, 39]. Due to the sensitivity of K-Means to initial centroids, we conduct the process 10 times and report the average results.

Results. Table 3 demonstrates the overall results of all methods for node clustering task. As we can see, HHGT consistently surpasses all baseline methods in node clustering across various HINs, demonstrating substantial improvements in both NMI and ARI metrics. Specifically, HHGT exhibits an improvement of up to 24.75% in NMI and 29.25% in ARI over the top-performing baselines on the ACM dataset, respectively. This is because our model simultaneously incorporates Ring-level Transformer and Type-level Transformer, where the former aids in capturing the differences between neighbors at different distances while the latter emphasizes the importance of node types during neighbor aggregation. The hierarchical integration of these two Transformers enables the model to synthesize information at multiple levels, enhancing the diversity and richness of node representations. Additionally, deep learning-based methods significantly outperform shallow models, as discussed in Section 5.2. Furthermore, homogeneous GT-based methods perform worse than their heterogeneous GT-based counterparts in many cases, verifying the importance of considering relation heterogeneity within HINs. Our model also benefits from this aspect by designing the (k, t)-ring neighborhood structure to emphasize the inherent heterogeneity of both distance and types within HINs.

5.4 Embedding Visualization (RQ2).

Settings. For a more intuitive comparison, we conduct embedding visualization to represent an HIN in a low-dimensional space. The goal is to learn node embeddings using the HIN representation learning model and project them into a 2-dimensional space. We employ the t-SNE [9] technique for visualization, with a specific focus on paper representations on both datasets. Following [39], nodes here are color-coded based on fields and published venues for ACM and MAG, respectively. The visualization results for ACM dataset can be found in Appendix D.1.

Results. The results are shown in Figure 4, from which we can find the following phenomenons: (1) The shallow model-based methods always show mixed patterns among papers from various venues, lacking clear clustering boundaries. For example, HIN2Vec mixes all papers together, limiting its ability to capture complex structural and semantic relationships in HINs. (2) HGNN-based models, such

[1] https://dl.acm.org
[2] https://www.microsoft.com/en-us/research/project/microsoft-academic-graph
[3] https://github.com/qiuyu111/HHGT
[§] The subgraph extracted from the ACM dataset in our paper differs from those used by certain baselines, e.g., FastGTN [42].

Table 1: Dataset Statistics

Dataset	Objects (#)	#Object	Relations	#Relation	#Label Type	#Labeled Object
ACM	P (4025), A (7167), S (60)	11,252	$P \rightleftharpoons A, P \rightleftharpoons S$	17,432	3	4,025
MAG	P (4017), A (15383), I (1480), F (5454)	26,334	$P \rightleftharpoons P, P \rightleftharpoons F, P \rightleftharpoons A, A \rightleftharpoons I$	86,230	4	4,017

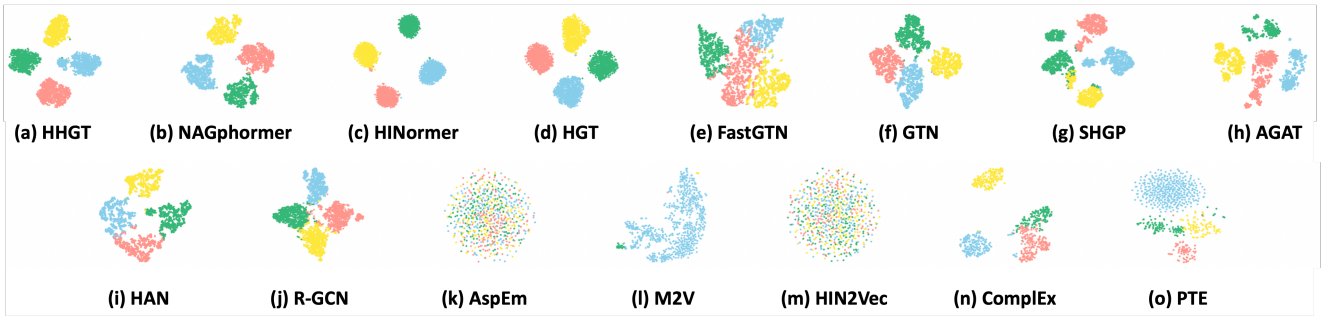
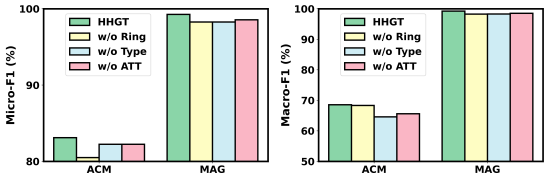


Figure 4: Embedding visualization on MAG dataset, where our HHGT model clearly separates papers from different published venues with well-defined boundaries.

Table 2: Overall evaluation on node classification. Tabular results are in percent; the best results are highlighted in bold; the underlined results indicate the second-best performance.

Methods	ACM		MAG	
	Macro-F1	Micro-F1	Macro-F1	Micro-F1
PTE	25.40	61.57	59.75	60.31
ComplEx	56.67	78.93	91.65	91.69
HIN2Vec	25.40	61.57	19.21	28.08
M2V	60.31	74.24	91.16	91.22
AspEm	58.72	80.70	16.85	26.11
R-GCN	51.34	72.80	96.40	96.39
HAN	60.34	81.42	97.39	97.36
AGAT	58.14	80.75	97.50	97.51
SHGP	61.91	81.69	97.19	97.04
GTN	64.49	78.26	96.63	96.64
FastGTN	64.70	76.77	93.08	93.03
HGT	65.48	74.91	98.80	98.76
HINormer	60.31	82.98	98.85	98.88
NAGphormer	58.94	81.49	97.76	97.76
HHGT	**68.56**	**83.11**	**99.25**	**99.25**

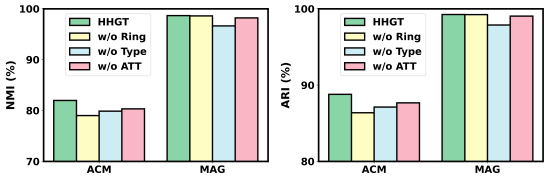


(a) Micro-F1 **(b) Macro-F1**

Figure 5: Ablation study on node classification.

Table 3: Overall evaluation on node clustering. Tabular results are in percent; the best results are highlighted in bold; the underlined results indicate the second-best performance.

Methods	ACM		MAG	
	NMI	ARI	NMI	ARI
PTE	0.26	0.03	20.44	10.71
ComplEx	29.83	25.25	73.45	71.71
HIN2Vec	0.40	0.15	0.40	0.01
M2V	39.54	32.29	3.68	2.12
AspEm	39.54	32.29	0.46	0.01
R-GCN	24.74	18.27	84.82	87.34
HAN	50.12	50.37	84.90	88.08
AGAT	40.33	39.42	86.73	81.40
SHGP	39.08	32.30	86.71	88.77
GTN	65.71	68.69	91.24	93.99
FastGTN	65.68	68.65	75.62	77.31
HGT	37.96	32.76	96.84	98.07
HINormer	41.66	35.06	96.35	97.76
NAGphormer	47.30	40.05	96.22	97.61
HHGT	**81.97**	**88.78**	**98.65**	**99.25**



(a) NMI **(b) ARI**

Figure 6: Ablation study on node clustering.

as GTN and R-GCN, provide more reasonable visualization results, but their clusters are more spread out and less compact. (3) In contrast, visualizations of heterogeous GT-based models, including HINormer, HGT and our HHGT, consistently exhibit high intra-class similarity. These models effectively distinguish papers from different published venues with well-defined boundaries, which effectively demonstrates the power of graph transformers in HINs.

5.5 Ablation Study (RQ3)

To understand the impact of different components within the proposed framework on the overall performance across both tasks,

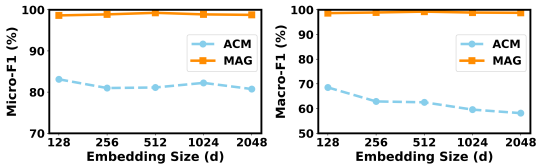
we conduct ablation studies by removing or replacing key modeling modules of HHGT on two datasets. Specifically, we focus on three key modules: (1) the k-ring neighborhood structure and its corresponding Ring-level Transformer for distance heterogeneity modeling, (2) the (k,t)-ring neighborhood structure and its corresponding Type-level Transformer for further type heterogeneity modeling, (3) and the attention-based readout function upon two Transformer encoders. By removing or replacing these modules, we can obtain different variants of HHGT as follows:

- **w/o Ring:** In this variant, we replace our k-ring structures with traditional k-hop patterns for neighbor extraction without partitioning them into different distance-based non-overlapping subsets, and the Ring-level Transformer is then utilized upon the extracted hop-based neighborhood structure.
- **w/o Type:** Within each k-ring structure, this variant mixes all neighbors of different node types without further partitioning them into different type-based subsets, and removes the Type-level Transformer module.
- **w/o ATT:** In this variant, we replace the attention-based readout functions defined in Equation (5) and Equation (7) with average pooling functions.

The results for node classification and node clustering on both datasets are illustrated in Figure 5 and Figure 6, respectively. Based on the results, we have the following observations: (1) The model performance significantly decreases across both datasets when the k-ring structure is replaced with the traditional k-hop pattern (i.e., HHGT vs. w/o Ring), which emphasizes the importance of the proposed k-ring structure. The reason is that the k-ring structure, integral to HHGT, excels in capturing distance heterogeneity within HINs by effectively differentiating between neighbors at varying distances. (2) HHGT consistently outperforms w/o Type in all metrics on both datasets, which demonstrates the effectiveness of our Type-level Transformer module. The results also highlight the importance of explicitly considering type heterogeneity in HIN representation learning by further partitioning k-ring to (k,t)-ring structure. (3) w/o ATT shows inferior performance compared to HHGT across two downstream tasks and two datasets, indicating that the proposed attention-based readout function is beneficial for learning more general and expressive node representations.

5.6 Parameter Study (RQ4)

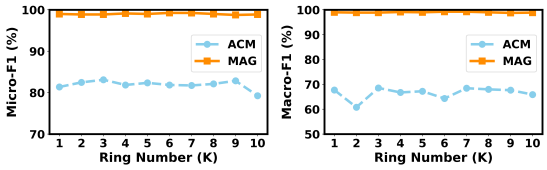
We investigate the sensitivity of HHGT with respect to four key hyper-parameters, i.e., the embedding size d, the number of rings K, the Ring-level Transformer layer number Lr and the Type-level Transformer layer number Lt. The results of node classification with varying d and K on both datasets are depicted in Figures 7-8. Further information with varying Lr and Lt is provided in Appendix D.2.



(a) Micro-F1 (b) Macro-F1

Figure 7: Embedding size study on node classification.

Effect of Embedding Size d. We vary d in $\{128, 256, 512, 1024, 2048\}$ to validate the impact of embedding size. Figure 7 reports the node classification results over both datasets. As observed, in most cases, model performance improves with increasing hidden dimension size, as a larger embedding size generally provides stronger representational power. However, it is interesting to discover that employing high-dimensional representations does not consistently yield optimal results. For instance, the model achieves the optimal Micro-F1 and Macro-F1 when $d = 128$ on ACM dataset and $d = 512$ on MAG dataset, respectively. This indicates adopting a higher-dimensional representation does not guarantee the best performance across all scenarios.



(a) Micro-F1 (b) Macro-F1

Figure 8: Ring number study on node classification.

Effect of Ring Number K. We range K from 1 to 10 to analyze the effect of the number of rings, and the node classification results are illustrated in Figure 8. As observed, the model achieves the best with different K on different datasets, since various HINs display distinct neighborhood configurations. Besides, as K increases, performance gradually improves across all datasets, followed by a slight decline observed with further increments. Though greater K implies nodes consider a broader neighborhood, too large K may cover the entire network, causing the node's neighborhood to include a significant amount of irrelevant information and even over-fitting.

6 Conclusion

In this paper, we study the HIN representation learning problem. To deal with it, we introduce an innovative (k,t)-ring neighborhood structure to extract neighbors for each node, aiming to capture the differences between neighbors at distinct distances and with different types. Based on this novel structure, we propose an effective HHGT model, seamlessly integrating a Type-level Transformer for aggregating nodes of different types within each k-ring neighborhood, and a Ring-level Transformer for hierarchical aggregation across multiple k-ring neighborhoods. Experiments on both datasets demonstrate the advantages of our HHGT model across various downstream tasks. In the future work, we plan to expand our evaluation to include additional downstream tasks such as link prediction, and further incorporate the direction of relations to improve representation learning for directed HINs.

Acknowledgments

This research is supported by the Ministry of Education, Singapore, under its Academic Research Fund (Tier 2 Award MOE-T2EP20221-0013, Tier 2 Award MOE-T2EP20220-0011, and Tier 1 Award (RG20/24)). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

References

[1] Uri Alon and Eran Yahav. 2020. On the bottleneck of graph neural networks and its practical implications, arXiv. *arXiv preprint arXiv:2006.05205* (2020).

[2] James Atwood and Don Towsley. 2016. Diffusion-convolutional neural networks. *Advances in neural information processing systems* 29 (2016).

[3] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).

[4] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE transactions on knowledge and data engineering* 30, 9 (2018), 1616–1637.

[5] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 3438–3445.

[6] Jinsong Chen, Kaiyuan Gao, Gaichao Li, and Kun He. 2022. NAGphormer: A tokenized graph transformer for node classification in large graphs. In *The Eleventh International Conference on Learning Representations*.

[7] Enyan Dai, Wei Jin, Hui Liu, and Suhang Wang. 2022. Towards robust graph neural networks for noisy graphs with sparse labels. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 181–191.

[8] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 135–144.

[9] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *the Journal of machine Learning research* 9 (2008), 1871–1874.

[10] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1797–1806.

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

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[13] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

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

[15] Zhipeng Huang, Yudian Zheng, Reynold Cheng, Yizhou Sun, Nikos Mamoulis, and Xiang Li. 2016. Meta structure: Computing relevance in large heterogeneous information networks. In *Proceedings of the 22nd ACM SIGKDD International conference on knowledge discovery and data mining*. 1595–1604.

[16] Wei Ju, Zheng Fang, Yiyang Gu, Zequn Liu, Qingqing Long, Ziyue Qiao, Yifang Qin, Jianhao Shen, Fang Sun, Zhiping Xiao, et al. 2024. A comprehensive survey on deep graph representation learning. *Neural Networks* (2024), 106207.

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

[18] Chuang Liu, Yibing Zhan, Xueqi Ma, Liang Ding, Dapeng Tao, Jia Wu, and Wenbin Hu. 2023. Gapformer: graph transformer with graph pooling for node classification. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI*. 2196–2205.

[19] Qidong Liu, Cheng Long, Jie Zhang, Mingliang Xu, and Dacheng Tao. 2022. Aspect-Aware Graph Attention Network for Heterogeneous Information Networks. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[20] 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 *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 1150–1160.

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

[22] Yuxiang Ren, Bo Liu, Chao Huang, Peng Dai, Liefeng Bo, and Jiawei Zhang. 2019. Heterogeneous deep graph infomax. *arXiv preprint arXiv:1911.08538* (2019).

[23] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*. Springer, 593–607.

[24] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S Yu Philip. 2018. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 31, 2 (2018), 357–370.

[25] Yu Shi, Huan Gui, Qi Zhu, Lance Kaplan, and Jiawei Han. 2018. Aspem: Embedding learning by aspects in heterogeneous information networks. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 144–152.

[26] Yizhou Sun and Jiawei Han. 2013. Mining heterogeneous information networks: a structural analysis approach. *Acm Sigkdd Explorations Newsletter* 14, 2 (2013), 20–28.

[27] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1165–1174.

[28] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*. PMLR, 2071–2080.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[30] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 1225–1234.

[31] 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 Transactions on Big Data* 9, 2 (2022), 415–436.

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

[33] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 1726–1736.

[34] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

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

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

[37] Xiaocheng Yang, Mingyu Yan, Shirui Pan, Xiaochun Ye, and Dongrui Fan. 2023. Simple and efficient heterogeneous graph neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 10816–10824.

[38] Yaming Yang, Ziyu Guan, Jianxin Li, Wei Zhao, Jiangtao Cui, and Quan Wang. 2021. Interpretable and efficient heterogeneous graph convolutional network. *IEEE Transactions on Knowledge and Data Engineering* 35, 2 (2021), 1637–1650.

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

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

[41] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *Advances in neural information processing systems* 32 (2019).

[42] Seongjun Yun, Minbyul Jeong, Sungdong Yoo, Seunghun Lee, S Yi Sean, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2022. Graph Transformer Networks: Learning meta-path graphs to improve GNNs. *Neural Networks* 153 (2022), 104–119.

[43] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2018. Metagraph2vec: Complex semantic path augmented heterogeneous network embedding. In *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part II 22*. Springer, 196–208.

[44] Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. 2019. Star-gcn: Stacked and reconstructed graph convolutional networks for recommender systems. *arXiv preprint arXiv:1905.13129* (2019).

[45] Jianan Zhao, Xiao Wang, Chuan Shi, Zekuan Liu, and Yanfang Ye. 2020. Network schema preserving heterogeneous information network embedding. In *International joint conference on artificial intelligence (IJCAI)*.

[46] Zeyuan Zhao, Qingqing Ge, Anfeng Cheng, Yiding Liu, Xiang Li, and Shuaiqiang Wang. 2023. Exploiting Latent Attribute Interaction with Transformer on Heterogeneous Information Networks. *arXiv preprint arXiv:2311.03275* (2023).

[47] Yizhen Zheng, Vincent CS Lee, Zonghan Wu, and Shirui Pan. 2021. Heterogeneous graph attention network for small and medium-sized enterprises bankruptcy prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 140–151.

[48] Shichao Zhu, Chuan Zhou, Shirui Pan, Xingquan Zhu, and Bin Wang. 2019. Relation structure-aware heterogeneous graph neural network. In *2019 IEEE international conference on data mining (ICDM)*. IEEE, 1534–1539.