Node-dependent Semantic Search over Heterogeneous Graph Neural Networks

Zhenyi Wang*
Beijing University of Posts and Telecommunications
China
zy_wang@bupt.edu.cn

Huan Zhao*
4Paradigm Inc.
China
zhaohuan@4paradigm.com

Fengqi Liang
Chuan Shi†
Beijing University of Posts and Telecommunications
China
lfq@bupt.edu.cn
shichuan@bupt.edu.cn

ABSTRACT

In recent years, Heterogeneous Graph Neural Networks (HGNNs) have been the state-of-the-art approaches for various tasks on Heterogeneous Graphs (HGs), e.g., recommendation and social network analysis. Despite the success of existing HGNNs, the utilization of the intricate semantic information in HGs is still insufficient. In this work, we study the problem of how to design powerful HGNNs under the guidance of node-dependent semantics. Specifically, to perform semantic search over HGNNs, we propose to develop semantic structures in terms of relation selection and connection selection, which could guide a task-relevant message flow. Furthermore, to better capture the diversified property of different node samples in HGs, we design predictors to adaptively decide the semantic structures per node. Extensive experiments on seven benchmarking datasets across different downstream tasks, i.e., node classification and recommendation, show that our method can consistently outperform various state-of-the-art baselines with shorter inference latency, which justifies its effectiveness and efficiency. The code and data are available at https://github.com/BUPT-GAMMA/NDS.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**.

KEYWORDS

Heterogeneous Graph Neural Networks, Node-dependent Semantic Search, Neural Architecture Search

ACM Reference Format:
Zhenyi Wang, Huan Zhao, Fengqi Liang, and Chuan Shi. 2023. Node-dependent Semantic Search over Heterogeneous Graph Neural Networks. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3614989

*Equal contribution.
†Corresponding author.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3614989

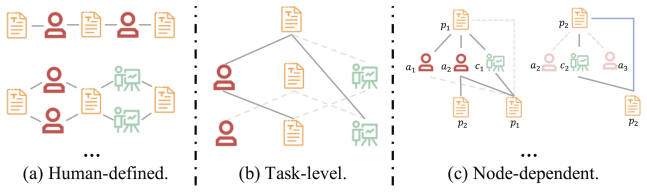


Figure 1: An overview of different genres of semantic structures for the given HG in Figure 2(a), which are, (a) the human-defined meta-paths and meta-graphs; (b) the task-level semantic structure searched by existing methods, e.g., GTN [53] and DiffMG [6]; and (c) the node-dependent semantic structures searched by our proposed NDS (more details can be checked in Section 4.4), respectively.

1 INTRODUCTION

Heterogeneous Graphs (HGs) [35, 42, 50] which involve various node types and relations, can model the complex relationships among entities across a variety of real-world scenarios, e.g., recommendation [1, 30], social network analysis [36, 40, 41, 43] (see Figure 2(a) for an example). Compared with homogeneous networks, HGs can encode rich semantic information by different combinations of relations. For example, early works [36] manually define relation sequences, i.e., *meta-path*, to compute the similarities under specific semantics between nodes. Some works [21, 57] further design more complex semantic structures, e.g., *meta-graph*.

Recently, Graph Neural Networks (GNNs) [24, 39, 46, 48, 59] have become the de facto technique in many graph-based learning tasks. GNNs typically adopt the message passing manner [11, 12], which updates the representations of the nodes by iterative feature aggregation from the topological neighbors. Despite the success of GNNs on homogeneous networks, it has been observed that the naive message passing on HGs which neglects the relations, i.e., semantic information, often leads to sub-optimal performance [6, 53]. Thus, many works make efforts in extending GNNs to utilize the task-relevant semantics to guide the message passing, i.e., designing different Heterogeneous GNNs (HGNNs) [6, 8, 17, 43, 53]. Due to the fact that HGNNs have been applied to various tasks [1, 2, 6, 13, 22, 30, 52, 53, 58], thus, it becomes an important problem to utilize task-relevant semantics in designing HGNNs.

Challenges. Despite the rapid development of HGNNs, the exploitation of the intricate semantic information in HGs is still inadequate. In this work, we study the problem of how to design

powerful HGNNs under the guidance of node-dependent semantics, which is non-trivial with the following two challenges:

(1) *How to design HGNNs with effective semantic search?* Early works conduct semantic search in HGs by manually defining semantic structures, i.e., meta-paths [36] or meta-graphs [7, 21], which are used for proximity computation. But in the context of HGNNs, effective semantic structures should (a) guide a task-relevant message flow; (b) be automatically derived to save human labor. Early HGNNs [8, 43] rely on human-defined meta-paths, which require extensive human endeavors and expert knowledge. The attention-based approaches [17, 32, 53], e.g., GTN [53], fuse relations with attention mechanisms, which still inevitably involves task-irrelevant noises. The recently proposed Neural Architecture Search (NAS) based approaches, e.g., DiffMG [6], tend to preserve only one relation in each HGNN layer, which sets an inflexible constraint to the searched semantic structures and thus hurts the expressiveness of HGNNs. The aforementioned methods are problematic from different perspectives; therefore, it remains to be investigated to design HGNNs with more effective task-relevant semantic search.

(2) *How to perform semantic search over HGNNs in a node-dependent manner*? Existing methods [6, 8, 10, 16, 17, 32, 43, 53] adopt the task-level semantic structure which is globally shared in an HG (see Figure 1(a)&(b)), neglecting the diversity of node instances. Recent studies have shown that the awareness of the disparity among data samples is useful in increasing model capacity and then boosting the performance of downstream tasks [5, 15, 19, 37, 38]. The similar intuition also exists in the context of HGNNs, where nodes with heterogeneous types and relations usually exhibit strong diversity, leading the task-relevant semantics to be node-dependent. For example, in a citation network shown in Figure 2(a), the research subjects of some papers could be highly relevant to the attributes of their authors, while those of others could be more related to their published conferences. Hence, searching for the same semantics for all nodes is not sufficient enough and uncovering the node-dependent semantics is in great demand. However, how to perform node-dependent semantic search over HGNNs (see Figure 1(c)) is still unexplored.

Present work. The above challenges motivate us to develop a novel **N**ode-**D**ependent **S**emantic search framework (dubbed **NDS**) over HGNNs. To address the first challenge, we propose to search for semantic structures in terms of two components: *relation selection* and *connection selection*, which contribute to the *local* and *high-order* development of the semantic structures, respectively. Then the HGNN architecture could be developed under the guidance of the searched semantic structures. To address the second challenge, we design predictors for predicting node-dependent selection strategies (see Figure 2(c)&(d)) inside the two components based on the node representations in the previous layer. The entire framework is trained with the alternate optimization of the backbone GNN and the predictor module to facilitate their collaboration. Furthermore, we conduct extensive experiments on two popular downstream tasks, i.e., node classification and recommendation, to verify the effectiveness of NDS.

To summarize, this work makes the following contributions.

- To the best of our knowledge, NDS is the first attempt to perform node-dependent semantic search over HGNNs, which not only is labor-saving but also better captures the diversified property of different node samples in HGs.
- We propose to develop semantic structures with relation selection and connection selection, and transform the semantic search over HGNNs to search the selection strategies inside these two key components. To promote the adaptiveness of NDS, we further design predictors to achieve node-dependent semantic search.
- Extensive experimental results demonstrate that NDS consistently outperforms the state-of-the-art methods on node classification and recommendation tasks with higher inference efficiency and could effectively discover the node-dependent semantics.

2 RELATED WORK

2.1 Heterogeneous Graph Neural Networks

GNNs have received significant research interest due to the prevalence of graph-structure data [46]. They typically adopt the message passing manner, which updates the representations of the nodes by iterative feature aggregation from the topological neighbors [24, 39, 48]. Considering the heterogeneity of real-world networks, many works have attempted to design different HGNNs to capture the intricate semantics in HGs for a better application in pratical scenarios, e.g., recomendation [1, 22, 30].

Early proposed HGNNs basically require human-defined meta-paths [8, 14, 22, 43]. HAN [13, 43] extracts homogeneous neighbors with meta-paths. MAGNN [8] additionally encodes the intermediate node information along each meta-path instance. However, designing useful meta-paths for different tasks requires extensive human endeavors and expert knowledge. Therefore, recently proposed HGNNs design different mechanisms to fuse the heterogeneous information [17, 32, 34, 53]. RGCN [34] differentiates relations with distinct weight matrices during message passing. GTN [53], HGT [17] and Simple-HGN [32] implicitly learn useful meta-paths with different attention mechanisms. Without explicit selection, the manner of fusing information from all types would inevitably incorporate task-irrelevant noise and is computationally inefficient [6]. In contrast, our work performs semantic search over HGNNs with explicit relation selection and connection selection, which effectively removes the task-irrelevant noises as well as lightens the computational load.

2.2 Graph Neural Architecture Search

Recently, researchers have attempted to leverage NAS to search for the optimal GNN architectures [3, 4, 9, 18, 26, 45, 51]. GraphNAS [9] and SANE [18] adopt Reinforcement Learning (RL)-based and differentiable search algorithm, respectively. F2GNN [45] searches the optimal stacking structure of GNNs for graph classification. In the context of HGNNs, GEMS [16] utilizes the evolutionary algorithm to search for meta-graphs in recommendation. HGNAS [10] uses RL-based search algorithm to search for the optimal HGNN architectures. DiffMG [6] searches for task-specific meta-graphs based on differentiable search. However, these works do not consider the dynamic nature of node samples in HGs. Besides, the NAS-based methods require a two-phase training procedure which involves

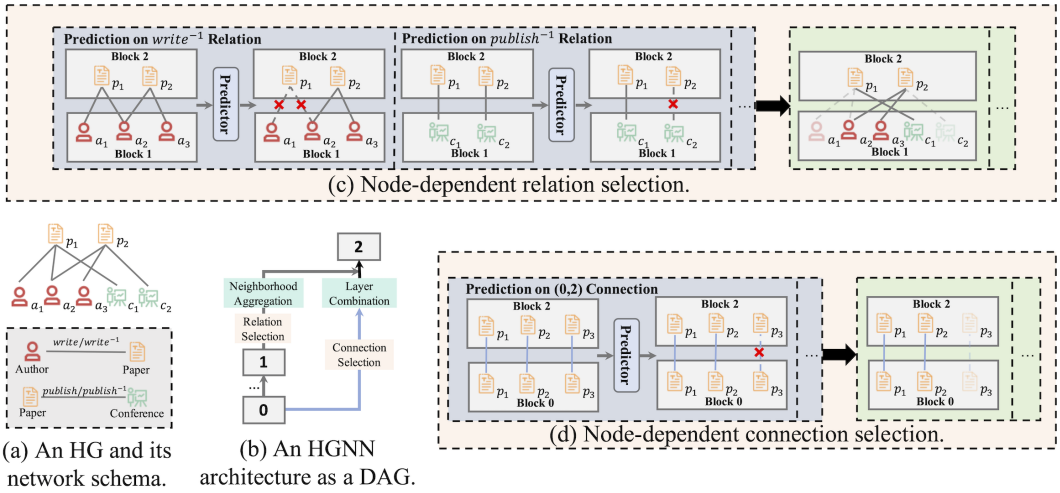


Figure 2: The overall framework of the proposed NDS. Best viewed in color. (a) An example HG, i.e., a citation network, and its network schema [50]; (b) A two-layer HGNN architecture example as a DAG with colored boxes on edges representing operations; (c)/(d) The pipeline of relation/connection selection (of paper nodes), where the block numbers could be corresponding to those in (b). Blue box: node-dependent relation/connection prediction. Green box: the predicted results of node-dependent relation/connection selection.

searching and retraining, while our proposed method can be trained in a single phase and thus enjoys ease of use.

2.3 Data-dependent Neural Networks

Data-dependent neural networks, which can adapt their structures or parameters to the input data during inference [15], have been extensively developed in the research field of computer vision [5, 15, 19, 37, 38]. The representative power of data-dependent neural networks can get increased due to the input-conditioned computation adaptation. Some researchers develop node-wise GNN models [25, 27, 31, 44, 56] for homogeneous graphs. Different from previous works, NDS is the first attempt to design more powerful HGNNs under the guidance of node-dependent semantic structures.

Compared with existing works, the delicately designed semantic structures and the node-dependent predictors lead to the superiority of NDS in expressiveness, adaptiveness and efficiency.

3 THE PROPOSED METHOD

3.1 Preliminary: Heterogeneous Message Passing

HGNNs [6, 8, 17, 43, 53] generally adopt the heterogeneous message passing manner [11, 12], which updates the representations of the nodes by iterative feature aggregation from the topological neighbors under different relations. Given an HG $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where $\mathcal{V}$ and $\mathcal{E}$ are the sets of nodes and edges, respectively, the heterogeneous message passing at the l^{th} HGNN layer can be formulated as follows:

$$\mathbf{H}^l = F_{\mathbf{\Theta}^l}\left(\mathcal{A}, \mathbf{H}^{l-1}\right), \tag{1}$$

where $\mathbf{H}^l \in \mathbb{R}^{N\times d}$ denotes the output node representations of the l^{th} layer ($l \in \{1, \cdots, L\}$), N denotes the number of nodes and d denotes the hidden dimension. $F_{\mathbf{\Theta}^l}$ is the relation-aware message passing function parameterized by $\mathbf{\Theta}^l$. $\mathcal{A} = \{\mathbf{A}_r\}_{r\in\mathcal{R}}$ is the collection of all the relation-specific adjacency matrices $\mathbf{A}_r \in \mathbb{R}^{N\times N}$ and $\mathcal{R}$ denotes the relation set of the HG. Specifically, $\mathbf{H}^0$ denotes the projected features output by type-specific transformation [8, 43] which projects the features of different node types into a common latent space. In this work, we focus on the message passing rule of GCN [24], while it is straightforward to extend the proposed framework to other GNNs [39, 48]. Despite various HGNNs proposed, the relationship between semantic search and message passing remains to be explored.

3.2 Semantic Structure over HGNNs

3.2.1 Overview. Conventional human-defined semantic structures, i.e., meta-path [36] or meta-graph [7, 21], mainly aim to characterize the proximity between nodes in an HG. As a comparison, **semantic structures** in the context of HGNNs tend to guide a task-relevant flow for message passing by effective selection and fusion over relations. The different combinations of such selection and fusion can be interpreted as different semantic information captured by the HGNNs, which is crucial to the network performance.

To facilitate our discussion, we represent an HGNN architecture as a directed acyclic graph (DAG) [18, 29, 47] as illustrated in Figure 2(b). The DAG is an ordered sequence of $(L+1)$ blocks ($L = 2$ in the illustrative example). The l^{th} block represents the output node representations $\mathbf{H}^l$ of the l^{th} HGNN layer and the directed edge (k, l) is associated with operations on $\mathbf{H}^k$ for deriving $\mathbf{H}^l$ (see the colored boxes in Figure 2(b)). Without loss of generality, the neighborhood aggregation and layer combination operations are fixed since they are not our focus in this work. **Semantic search** aims to discover appropriate relation selection and connection selection operations (see Figure 2(c)&(d) for the graphical pipeline)

in designing HGNN architectures. The two operations contribute to the local and high-order development of the semantic structures, respectively, arming the HGNNs with semantic expressiveness. We will introduce the technical details of these two operations in the following sections.

3.2.2 Relation Selection. The diverse relations in HGs provide information abundance, which helps to boost the downstream task performance [35, 42, 50]. Nevertheless, noises are also incorporated by the task-irrelevant type information. For example, in a citation network, the institution that an author belongs to can be deemed as noise when predicting his/her research field. Thus, when HGNNs perform neighborhood aggregation, the heterogeneous type information should be selectively utilized to attenuate the noises and generate task-relevant message flows.

Motivated by the above discussion, on the edges between two consecutive blocks in the DAG (edge $(0, 1)$ and $(1, 2)$ in Figure 2(b)), we assign the relation selection operation to prepare the local pattern of semantic structures for the subsequent neighborhood aggregation.

To be specific, at the l^{th} layer, the node representations get updated after the relation-aware neighborhood aggregation:

$$\mathbf{H}_R^l = \sigma\left(\tilde{\mathbf{A}}^l \mathbf{H}^{l-1} \mathbf{\Theta}^l\right), \tag{2}$$

where $\mathbf{H}_R^l$ denotes the representations derived from the relation selection component, $\tilde{\mathbf{A}}^l$ is the normalized adjacency matrix and σ is the activation function.

The relation selection is achieved by searching to filer out the task-irrelevant relations while preserving those task-relevant ones. $\mathbf{A}^l$ is generated as follows:

$$\mathbf{A}^l = \sum_{r \in \mathcal{R}} \mathbf{A}_r \odot \mathbf{M}_r^l, \tag{3}$$

where $\odot$ denotes the element-wise matrix multiplication, and $\mathbf{M}_r^l \in \mathbb{R}^{N \times N}$ denotes the mask matrix for relation r at the l^{th} layer and will be automatically learned. The detailed calculation procedure of $\mathbf{M}_r^l$ will be introduced in Section 3.3.

The information propagation paths of HGNNs established under the guidance of effective relation selection would convey task-relevant semantics.

3.2.3 Connection Selection. After exploring the local development of semantic structures guided by relation selection, we next introduce to develop their high-order wiring with connection selection, which further increases the capacity of HGNNs. The information propagation paths with different lengths could convey semantic information from different aspects. Since longer propagation paths may bring unwanted noises while shorter ones are not capable of carrying adequate semantic information, we enable HGNNs to flexibly fuse the semantics captured by information propagation paths with different lengths. Motivated by the effectiveness of residual connection [20, 28, 49] which fuses the features obtained by different neural network layers, we adopt the connection selection as the operation on the edges between two non-consecutive blocks of the DAG (edge $(0, 2)$ in Figure 2(b)).

Specifically, at the l^{th} layer, the connection selection procedure is performed between itself and all its non-consecutive predecessors. Then representations from the selected layers are combined with summation:

$$\mathbf{H}_C^l = \sum_{k=0}^{l-2} \mathbf{H}^k \odot \mathbf{O}_k^l, \tag{4}$$

where $\mathbf{H}_C^l$ denotes the representation derived from the connection selection component at the l^{th} layer, $\mathbf{O}_k^l \in \mathbb{R}^{N \times d}$ denotes the connection mask matrix, which will be automatically learned. The calculation procedure of $\mathbf{O}_k^l$ will also be introduced in Section 3.3.

The final output of the l^{th} HGNN layer is the combination of $\mathbf{H}_R^l$ and $\mathbf{H}_C^l$. Here, we choose summation as the combination function:

$$\mathbf{H}^l = \mathbf{H}_R^l + \mathbf{H}_C^l. \tag{5}$$

The collection of the relation and connection mask matrices, i.e., $\left\{\mathbf{M}_r^l | r \in \mathcal{R}, l \in \{1, \cdots, L\}\right\} \cup \left\{\mathbf{O}_k^l | l \in \{1, \cdots, L\}, k \in \{0, \cdots, l-2\}\right\}$, can be interpreted as the searched semantic structure. With the construction of both local patterns and high-order wiring, such an expressive semantic structure is capable of capturing complicated semantic information in HGs, which can guide a task-relevant message passing flow of HGNNs. To promote the adaptiveness of our proposed framework, we further refine the semantic search to be *node-dependent*.

3.3 Node-dependent Semantic Search

Recent studies have shown that the awareness of the disparity of data instances is useful in increasing the capacity of the neural networks and then boosting the downstream task performance [19, 25, 33]. The similar intuition also exists in the context of HGNNs. A static semantic structure shared by the whole HG is insufficient to adaptively capture the intricate semantic information for different nodes. To overcome the limitation, we propose to search for the node-dependent semantic structures. Specifically, the predictor modules are designed for predicting the node-dependent strategies of the relation selection and connection selection.

We first elaborate on the predictor for node-dependent relation selection. Concretely, as illustrated in Figure 2(c), given a target node i, our goal is to predict its selection probability for each of its associated relation, i.e., $\left\{p(\delta_r^l(i) = u) | u \in \{0, 1\}\right\}$, where $\delta_r^l(i)$ denotes the selection strategy of relation r for node i at the l^{th} layer. $\delta_r^l(i) = 1$ represents the strategy of preserving, and vice versa. To achieve node-dependent prediction, a predictor Φ_r^l takes $\mathbf{h}_i^{l-1}$ as input, and apply temperature-aware softmax to derive the relation selection strategy, as below:

$$\boldsymbol{\phi}_{r,i}^l = \Phi_r^l\left(\mathbf{h}_i^{l-1}\right), \tag{6}$$

$$p(\delta_r^l(i) = u) = \frac{\exp(\boldsymbol{\phi}_{r,i}^l[u]/\tau)}{\sum_{u' \in \{0,1\}} \exp(\boldsymbol{\phi}_{r,i}^l[u']/\tau)}, \tag{7}$$

where $\boldsymbol{\phi}_{r,i}^l \in \mathbb{R}^{1 \times 2}$ denotes the output vector of predictor Φ_r^l. In this work, we use a two-layer Multi-Layer Perceptron (MLP) as the predictor Φ_r^l. τ is the temperature parameter to control the "sharpness" of the output distribution: as $\tau \to 0$, the output distribution of softmax becomes one-hot. In this work, τ is steadily

annealed from an initial value τ_0 to a small value τ_{min}, which are both hyper-parameters.

After obtaining the approximation of the hard selection probabilities, the relation mask matrix $\mathbf{M}_r^l$ in Eq. (3) can be developed as, $\mathbf{M}_r^l[:,i] = p(\delta_r^l(i)=1)\cdot\mathbf{1}$, where $\mathbf{M}_r^l[:,i]\in\mathbb{R}^{N\times 1}$ denotes the i^{th} column vector of $\mathbf{M}_r^l$, and $\mathbf{1}\in\mathbb{R}^{N\times 1}$ denotes a N-dimensional column vector filled with value 1.

The connection mask matrix $\mathbf{O}_k^l$ in Eq. (4) can be obtained in a similar way. As illustrated in Figure 2(d), a two-layer MLP is adopted as the predictor Ψ_k^l to derive the connection selection strategy, as below:

$$\boldsymbol{\psi}_{k,i}^l = \Psi_k^l\left(\mathbf{h}_i^{l-1}\right), \tag{8}$$

$$p(\eta_k^l(i)=u) = \frac{\exp(\boldsymbol{\psi}_{k,i}^l[u]/\tau)}{\sum_{u'\in\{0,1\}}\exp(\boldsymbol{\psi}_{k,i}^l[u']/\tau)}, \tag{9}$$

where $\boldsymbol{\psi}_{k,i}^l\in\mathbb{R}^{1\times 2}$ denotes the output vector after dimension projection, $\eta_k^l(i)$ denotes the connection selection strategy between the k^{th} and the l^{th} layer for node i, and $\left\{p(\eta_k^l(i)=u)|u\in\{0,1\}\right\}$ denotes the probability of connection. For simplicity, the value of the temperature τ keeps consistent with that in Eq. (7).

After obtaining the approximation of hard selection probabilities, the connection mask matrices can be obtained as, $\mathbf{O}_k^l[i,:] = \left[p(\eta_k^l(i)=1)\cdot\mathbf{1}'\right]^{\mathrm{T}}$, where $\mathbf{O}_k^l[i,:]\in\mathbb{R}^{1\times d}$ denotes the i^{th} row vector of $\mathbf{O}_k^l$, and $\mathbf{1}'\in\mathbb{R}^{d\times 1}$ indicates the d-dimensional column vector filled with value 1.

With the help of the predictors, the semantic search can be performed in a node-dependent manner to enable HGNNs to adaptively capture the diverse semantic information for different nodes.

3.4 Optimization

Guided by the searched semantic structures, the HGNNs perform L steps of message passing as shown from Eq. (2) to Eq. (5). Then the final representation matrix $\mathbf{H}$ could be obtained and used for downstream tasks.

Given that the backbone GNN and the predictor modules are mutually dependent since the former relies on the latter to guide its message passing with the predicted semantic structures, while the latter makes predictions based on the information captured by the former, jointly training them from scratch could lead to inferior task performance. Let $\boldsymbol{\omega}$ and $\boldsymbol{\lambda}$ be the set of parameters of GNN and predictors, respectively; we use an alternating optimization schema to iteratively update them to promote their cooperation.

Update $\boldsymbol{\omega}$. When updating $\boldsymbol{\omega}$, we fix $\boldsymbol{\lambda}$ and arrive at: $\min_{\boldsymbol{\omega}}\mathcal{L}_{task}$, where $\mathcal{L}_{task}$ denotes the loss function determined by the corresponding task.

Update $\boldsymbol{\lambda}$. The $\boldsymbol{\omega}$ is fixed when updating $\boldsymbol{\lambda}$. It's worth noting that apart from $\mathcal{L}_{task}$, we further incorporate regularization terms to encourage the sparsity of the searched semantic structures by the predictors, which could increase the computational efficiency during the inference phase. Take relation selection as an example, we sum up the elements of the mask matrix $\mathbf{M}_r^l$ and penalize the deviation from a target ratio α_1. The relation sparsity regularization term is defined as: $\mathcal{L}_{reg}^R = \left(\frac{1}{\gamma_1}\sum_{r\in\mathcal{R}}\sum_{l=1}^{L}\|\mathbf{M}_r^l\|_1-\alpha_1\right)^2$, where $\|\cdot\|_1$ denotes the L_1 norm, γ_1 denotes the factor to normalize the first term to be in the range $[0,1]$, and α_1 is a hyper-parameter.

Similarly, the connection sparsity regularization term can be defined as: $\mathcal{L}_{reg}^C = \left(\frac{1}{\gamma_2}\sum_{l=1}^{L}\sum_{k=0}^{l-2}\|\mathbf{O}_k^l\|_1-\alpha_2\right)^2$, where γ_2 and α_2 denote the normalization factor and the target connection sparsity ratio, respectively. Then, the overall objective function when updating $\boldsymbol{\lambda}$ can be formulated as:

$$\min_{\boldsymbol{\lambda}}\mathcal{L} = \mathcal{L}_{task} + \beta_1\mathcal{L}_{reg}^R + \beta_2\mathcal{L}_{reg}^C, \tag{10}$$

where β_1 and β_2 are the regularization hyper-parameters.

NDS is optimized by stochastic gradient descent in the above alternating manner. Its full algorithm is summarized in Algorithm 1.

Algorithm 1: The framework of NDS.

Input: An HG $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ with relation set $\mathcal{R}$ and the collection of all the relation-specific adjacency matrices $\mathcal{A}=\{\mathbf{A}_r\}$, input node features $\{\mathbf{x}_i|i\in\mathcal{V}\}$, number of layers L ;

Output: The learned model parameters ω^* and λ^*;

```
Initialization with type-specific transformation ;
while not converged do
    for l ∈ {1, ··· , L} do
        /* node-dependent relation selection */
        for r ∈ R do
            for i ∈ { i | Σ_j A_r[j,i] ≠ 0 } do
                Derive { p(δ_r^l(i) = u) | u ∈ {0,1} } based on Eq. (6)-(7) ;
                M_r^l[:,i] = p(δ_r^l(i) = 1) · 1 ;
            end
        end
        Calculate H_R^l base on Eq. (2)-(3) ;
        /* node-dependent connection selection */
        for i ∈ V do
            for k ∈ {0, ··· , l − 2} do
                Derive { p(η_k^l(i) = u) | u ∈ {0,1} } based on Eq. (8)-(9) ;
                O_k^l[i,:] = [ p(η_k^l(i) = 1) · 1' ]^T ;
            end
        end
        Calculate H_C^l based on Eq. (4) ;
        H^l = H_R^l + H_C^l ;
    end
    Calculate L_task and back-propagate it to update ω ;
    Calculate L based on Eq. (10) and back-propagate it to update λ ;
end
```

Table 1: Statistics of the datasets for node classification task.

Dataset	# of Nodes	# of Node Types	# of Edges	# of Relations	Target	# of Classes
IMDB	12,624	3	37,288	4	movie	3
ACM	8,994	3	25,922	4	paper	3
DBLP	18,405	3	67,946	4	author	4
HGBn-DBLP	26,128	4	239,566	6	author	4

Table 2: Statistics of the datasets for recommendation task.

Dataset	# of Nodes	# of Node Types	# of Edges	# of Relations	Target
Amazon	9,279	5	419,492	8	user-item
Yelp	31,092	5	904,276	10	user-business
Douban	37,595	6	3,429,882	12	user-movie

In the inference phase, it directly applies the learned relation selection and connection selection strategy, alleviating the overhead of message passing.

In this work, two popular downstream tasks over HGs, i.e., node classification and recommendation, will be examined in the experiments.

Node classification aims to predict the node labels based on the out node representations. For the node classification task, a linear layer is appended after the last layer of NDS, which projects the hidden dimension to the number of classes: $\hat{\mathbf{Y}} = \text{softmax}\left(\mathbf{H}\mathbf{\Theta}_o\right)$, where $\mathbf{\Theta}_o \in \mathbb{R}^{d\times C}$ denotes the output weight matrix and C denotes the number of classes. Then, we use the widely-adopted cross-entropy loss over all labeled nodes as the task loss:

$$\mathcal{L}_{task} = -\sum_{v\in\mathcal{V}_L}\sum_{c=1}^{C}\boldsymbol{y}_v[c]\log\hat{\boldsymbol{y}}_v[c],$$

where $\mathcal{V}_L$ denotes the set of labeled nodes, $\boldsymbol{y}_v \in \mathbb{R}^{C\times 1}$ is a one-hot vector indicating the ground-truth label of node v, and $\hat{\boldsymbol{y}}_v$ denotes the predicted label for the corresponding node.

Recommendation aims to predict the existence of the link between source nodes (e.g., users) and target nodes (e.g., items) based on their output representations. Following previous works [6], for the recommendation task, we adopt the following function to calculate the task loss:

$$\mathcal{L}_{task} = -\sum_{(u,v)\in\Omega^+}\log\sigma'\left(\mathbf{h}_u^\top\mathbf{h}_v\right) - \sum_{(u',v')\in\Omega^-}\log\sigma'\left(-\mathbf{h}_{u'}^\top\mathbf{h}_{v'}\right),$$

where Ω^+ and Ω^- denote the set of observed positive pairs and the set of negative pairs respectively, and σ' denotes the sigmoid function. $\mathbf{h}_u$, $\mathbf{h}_v$, $\mathbf{h}_{u'}$ and $\mathbf{h}_{v'}$ are output node representations.

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Datasets. We evaluate the effectiveness of NDS on seven datasets across two benchmarking tasks: node classification and recommendation, for HGNNs. For the node classification task, we use four real-world datasets: ACM, DBLP, IMDB and HGBn-DBLP. For the former three datasets, we follow the data split provided by previous works [6, 43, 53] and for the last one, we use the preprocessed data provided by the authors of [32].

The statistics of the datasets are summarized in Table 1 and their descriptions are as follows.

- **ACM** [43] is a citation network which consists of three types of node: papers, authors and subjects. The papers are labeled by the conference they are published in.
- **DBLP** [43] is also a citation network which consists of three types of node: papers, authors and conferences. The authors are labeled by their research areas.
- **IMDB** [43] is a movie rating website and we use the extracted data subset which consists of three types of nodes: movies, actors and directors. Movies are labeled by their genres.
- **HGBn-DBLP** [32] is another preprocessed version of DBLP citation network with a larger scale and more node and edge types. Specifically, it contains four types of nodes: papers, authors, venues and terms.

For the recommendation task, we use three commonly used heterogeneous recommendation datasets [1]. Edges of a target relation are to be predicted. The statistics of the datasets are summarized in Table 2 and their descriptions are as follows.

- **Amazon** is a large e-commerce platform where users give ratings to different items. It consists of five types of nodes: users, items, reviews, categories and brands.
- **Yelp** is a platform where users give rating reviews to business. It is composed of five types of nodes: users, business, compliments, categories and cities.
- **Douban** is a social media community where users share rating reviews about movies. There are six types of nodes: users, movies, groups, actors, directors and types.

[1] https://github.com/librahu/HIN-Datasets-for-Recommendation-and-Network-Embedding

Table 3: The Macro-F1 results of node classification and AUC results of recommendation ($\% \pm \sigma$). Note that we directly use the reported results of common baselines on the common datasets in [6] and [32]. We further list the average rank of each method on all datasets, denoted by Avg. Rank. The best results are bold and the second-best are underlined. Vacant positions ("-") are due to the fact that GEMS is designed for recommendation.

Category	Method	Node Classification				Recommendation			Avg. Rank↓
		ACM	IMDB	DBLP	HGBn-DBLP	Amazon	Yelp	Douban	
Homogeneous GNNs	GCN	92.56±0.20	55.19±0.73	90.46±0.41	90.84±0.32	66.64±1.00	58.98±0.52	77.95±0.05	9.43
	GAT	92.50±0.23	53.37±1.27	93.92±0.28	93.83±0.27	55.70±1.13	56.55±0.05	77.58±0.33	8.57
HGNNs w/ meta-paths	HAN	91.20±0.25	55.09±0.67	92.13±0.26	91.67±0.49	67.35±0.11	64.28±0.20	82.65±0.08	8.57
	MAGNN	91.15±0.19	56.44±0.63	92.81±0.30	93.28±0.51	68.26±0.09	64.73±0.24	82.44±0.17	7.86
HGNNs w/o meta-paths	RGCN	92.71±0.20	58.16±0.80	91.68±0.47	91.52±0.50	71.27±0.15	67.10±0.15	82.76±0.03	6.00
	GTN	92.62±0.17	59.68±0.72	93.98±0.32	93.52±0.55	71.82±0.18	66.27±0.31	83.26±0.10	4.43
	HGT	91.83±0.23	59.35±0.79	93.67±0.22	93.01±0.23	74.75±0.08	68.07±0.35	83.38±0.06	5.00
	Simple-HGN	92.71±0.22	60.49±0.70	93.81±0.11	94.01±0.24	71.75±0.29	66.54±0.21	78.45±0.31	4.57
Semantic search	GEMS	-	-	-	-	70.66±0.14	65.12±0.27	83.00±0.05	6.67
	DiffMG	92.65±0.15	61.04±0.56	94.45±0.15	94.14±0.21	75.28±0.08	68.77±0.13	83.78±0.09	2.29
	Random	85.77±4.13	50.48±5.00	80.43±1.24	70.02±10.24	70.90±0.97	61.59±3.77	76.52±1.66	10.43
	NDS	**93.11±0.04**	**61.23±0.41**	**94.65±0.22**	**94.62±0.12**	**75.54±0.06**	**69.20±0.04**	**83.90±0.11**	**1.00**

We adopt the same data split and negative sampling strategies as [6]. Specifically, 50% of the ratings are randomly chosen to provide supervision signals and be removed from the message passing network to avoid test label leakage [54, 55]. Ratings higher than 3 are labeled as positive links and those lower than 4 are negative ones. Random negative sampling is performed among unconnected pairs to make the number of positive and negative links match. The links of the two labels are then randomly split into a training set, a validation set and a test set according to the ratio of 3:1:1, respectively.

4.1.2 Baselines. We use four categories of methods as baselines: (1) *homogeneous GNNs*: GCN [24] and GAT [39]; (2) *HGNNs requiring human-defined meta-paths* : HAN [43] and MAGNN [8], whose meta-paths are set as suggested by the original papers; (3) *HGNNs with no need for customized meta-paths*: RGCN [34], GTN [53], HGT [17] and Simple-HGN [32] and (4) *semantic search methods*: DiffMG [6], which is the state-of-the-art semantic search baseline based on NAS. For the recommendation methods, we also use GEMS [16] which utilizes the evolutionary algorithm to search for meta-graphs between the source and the target node type, and we additionally incorporate a baseline that randomly adopts the relation selection and connection selection strategies (denoted as "Random" in Table 3). We adopt the results of common baselines on the common datasets reported in previous works [6, 32]. The code and data are provided at https://github.com/BUPT-GAMMA/NDS.

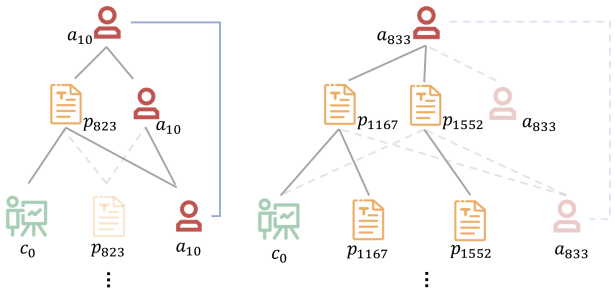
4.1.3 Implementation Details. We implement the proposed method NDS using the machine learning framework PyTorch[2] and the deep graph library DGL[3]. We use Intel(R) Xeon(R) Gold 6230R CPU @ 2.10GHz and GeForce RTX 3090 as the experimental environment. For each method, we report the average score and standard deviation of 10 runs, except for HGBn-DBLP, on which we reported the aggregated results of 5 runs returned by the official evaluation platform[4]. The max number of epochs is set as 250, and we use early-stopping strategy with patience 40, to alleviate over-fitting. Testing performance in the best epoch of validation set is reported. The layer number of all the methods is set as 4, and the hidden dimension is 64. All the methods are optimized with Adam [23] optimizer. For our method, the learning rate of the GNN module is searched in $\{1, 3\} \times \{10^{-3}, 10^{-2}\}$, the learning rate of the predictor is searched in $\{1, 5\} \times \{10^{-4}, 10^{-3}, 10^{-2}\}$, and the weight decay rate is searched in $\{0, 1, 2, 5\} \times \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$. Since we keep our experimental setting and evaluation metric consistent with that in [6] and [32], we use the experimental results reported in their papers of the common baselines on the common dataset in Table 3. And for the rest of results, the baselines are re-implemented based on DGL and are initialized with same hyper-parameters suggested by the original papers with further tuning for optimal performance.

4.2 Node Classification

The Macro-F1 scores of different methods on the node classification task are reported in Table 3. We have the following observations:

- The HGNNs, such as GTN and Simple-HGN, achieve competitive experimental performance, which reveals the necessity of considering the semantics in HGs when designing HGNNs.
- DiffMG is the most competitive baseline, which demonstrates the importance of searching for task-relevant semantics in designing HGNNs.
- The Random baseline consistently achieves poor performance, which demonstrates the effectiveness of the predictor modules in searching for node-dependent semantics.
- NDS achieves the best performance across all four datasets, which verifies its effectiveness. Specifically, comparing NDS with HGNNs like GTN, the consistent performance gain indicates the effectiveness of explicit relation selection to filter out the task-irrelevant noises. Meanwhile, NDS consistently achieves better performance

[2]https://pytorch.org/
[3]https://www.dgl.ai/
[4]https://www.biendata.xyz/hgb/



(a) a_{10}. (b) a_{833}.

Figure 3: The visualization of the searched semantic structures of two randomly selected target nodes. Best viewed in color. The grey/blue lines represent relations/connections. Note that self-loop is also regarded as a particular relation. The solid/dotted lines represent the preserved/removed relations or connections.

over DiffMG, which could be attributed to the improved expressiveness of the node-dependent semantic structures.

4.3 Recommendation

The AUC results of different methods on the recommendation task are reported in Table 3. From the table, the following observations can be obtained:

- On the recommendation datasets with larger scales and more diverse type information than those of the node classification task (see Table 1 and Table 2 for dataset details), HGNNs significantly perform better than homogeneous GNNs. It confirms the usefulness of designing heterogeneous message passing, especially for more complex networks in real-world recommendation scenarios, which aligns with the findings in previous works [22, 30].
- The Random baseline generally performs better than homogeneous GNNs, which again indicates the importance of leveraging the heterogeneous information in recommendation.
- Similar to the results of node classification, NDS achieves the best performance across all recommendation datasets, which again validates the effectiveness of node-dependent semantic search.

4.4 Case Study

To further showcase the effectiveness of the node-dependent semantic search, in this section, we provide a visualization study on DBLP dataset with regard to the searched semantic structures for different nodes. Specifically, we randomly select two nodes of the target type author, i.e., a_{10} and a_{833}, and visualize their semantic structures in Figure 3. For the sake of simplicity, we only depict the semantic structures starting from the target nodes and expanding for two layers. The selected relations and connections are in bright color while those unselected ones are faded. The nodes without connecting links are also faded.

From Figure 3, we can observe that the preserved relations and residual connections of a_{10} and a_{833} apparently differ from each other, resulting in a clear disparity between the two searched semantic structures and the captured semantic information. For example,

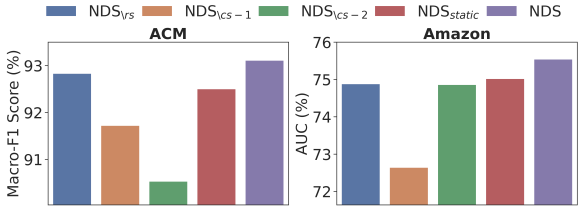


Figure 4: The comparison of NDS and its variants.

a_{10} is observed to relate more to the message transformed from its own features while neighbor messages are more favorable to a_{833}.

Taking into consideration all the above results, we can observe the effectiveness of the proposed NDS on capturing the node-dependent semantics according to the heterogeneous node property. Next, we further conduct experiments to analyze the designed components of NDS.

4.5 Ablation Study

In this section, we compare NDS with its four variants on dataset ACM and Amazon to validate the effectiveness of the proposed modules. The results are given in Figure 4 and the description of the used variants are given as below:

- NDS$_{\backslash rs}$: NDS without relation selection, i.e., all the available relations are remaining for message passing between consecutive HGNN layers.
- NDS$_{\backslash cs-1}$: NDS without connection selection where HGNN layers are *stacked* without residual connection and the output of the last layer is regarded as the final representation.
- NDS$_{\backslash cs-2}$: NDS without connection selection where each HGNN layer is *densely connected* with all its predecessors. Such connection selection manner has been proven to be effective in previous works [20, 28].
- NDS$_{static}$: NDS without node-dependent semantic search, i.e., the predicted semantic structure globally shared by all node samples.

From Figure 4, we can see that NDS consistently outperforms all its variants across the two datasets, demonstrating the effectiveness of the designed modules, i.e., the relation selection, connection selection, and node-dependent predictors. One more interesting observation is that the variants without connection selection obtain the worst two performances on both datasets, which implies the larger importance of the connection selection in designing HGNNs.

4.6 Inference Efficiency Analysis

In this part, we show the superiority of NDS in inference efficiency over existing methods. As shown in Figure 5, we compare the task performance and GPU/CPU inference latency of different methods on dataset ACM and Amazon. For better presentation, two baselines, i.e., MAGNN [8] and HGT [17], are excluded here due to their significantly longer inference latency than all other methods. It can be observed that NDS enjoys a clear advantage in inference efficiency over all the baselines while maintaining the best task performance across node classification and recommendation. It is mainly attributed to the fact that the node-dependent semantic search could derive sparsified semantic structures and remove

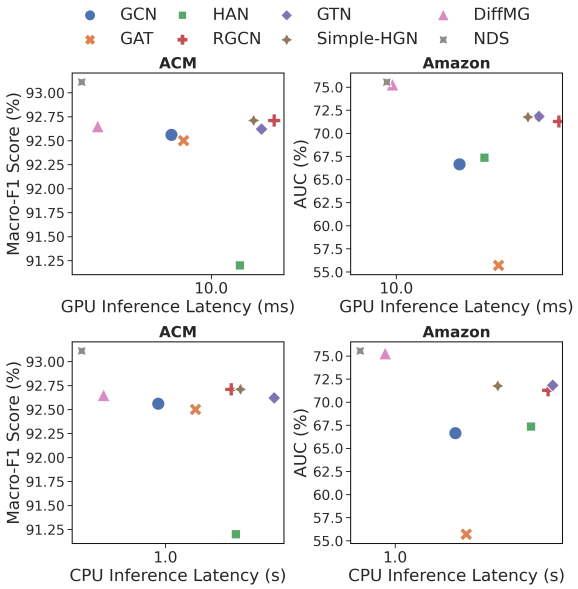


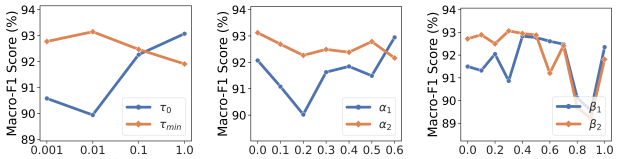
Figure 5: The comparison on task performance and inference efficiency of different methods.

unnecessary computations. Despite the competitive task performance of some attention-based methods, e.g., Simple-HGN [32] and GTN [53], they exhibit unsatisfactory inference efficiency due to the heavy attention computations. It is worth noting that DiffMG [6] obtains competitive inference efficiency since it only remains a single relation between HGNN layers for neighborhood aggregation, which is naturally more computationally efficient than adopting binary masks as in NDS. However, the consistent inference efficiency gain of NDS over DiffMG further demonstrates the effectiveness of the node-dependent semantic search in adaptively removing the noisy relations and connections for each node. In summary, the delicately designed semantic structure and node-dependent predictors lead to the improvement of NDS in both task performance and efficiency.

4.7 Impact of Hyper-parameters

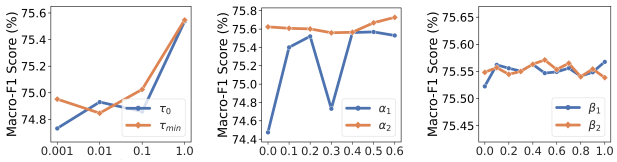
We investigate the impact of hyper-parameters and report the results for node classification task and recommendation task on dataset ACM and Amazon, respectively. In the experiments, we vary the studied hyper-parameter while fixing the others. Since only one hyper-parameter is varied at a time, the performance fluctuation would not be extremely sharp thanks to the existence of other working hyper-parameters.

4.7.1 Impact of the Temperature τ_0 and τ_{min}. The temperature defined in Eq. (7) and (9) controls the sharpness of the output distribution of softmax. To analyze the impact of temperature on NDS, we vary τ_0 and τ_{min} in the range of 0.001 to 1 with a log-scale step size of base 10, respectively, and show the corresponding results on datasets ACM and Amazon in Figure 6(a) and 7(a). We can observe that setting τ_0 as a large value 1 on both datasets can similarly achieve better performance than smaller values. It is probably because a small initial value of temperature may result in the gradient



(a) τ_0 and τ_{min}. (b) α_1 and α_2. (c) β_1 and β_2.

Figure 6: Impact of hyper-parameters on ACM.



(a) τ_0 and τ_{min}. (b) α_1 and α_2. (c) β_1 and β_2.

Figure 7: Impact of hyper-parameters on Amazon.

oscillation and training unstableness. As for the τ_{min}, NDS prefers to obtain a small temperature value by annealing on ACM, while it prefers to keep the temperature constant as 1 on Amazon.

4.7.2 Impact of the Target Sparsity Ratio α_1 and α_2. In the sparsity regularization terms of $\mathcal{L}$, i.e., $\mathcal{L}_{reg}^{R}$ and $\mathcal{L}_{reg}^{C}$ (see Section 3.4), α_1 and α_2 denote the target sparsity ratio of relations and connections, respectively. We vary each in the range from 0 to 0.6 with a step size of 0.1, and report the results on ACM and Amazon in Figure 6(b) and 7(b), respectively. We can observe that the optimal target sparsity ratio of relations α_1 on ACM is 0.6 while it is 0.4 on Amazon. A possible reason is that the relations of Amazon are more diverse than ACM, where a higher level of noises may be potentially incorporated, and a smaller α_1 can help to filter out more noises. As for the target sparsity ratio of connections α_2, the changing magnitude of the scores on both datasets is smaller than that of α_1, indicating lower sensitivity of NDS towards α_2.

4.7.3 Impact of the Coefficient β_1 and β_2. In the optimization loss $\mathcal{L}$ (see Section 3.4), the coefficient β_1 and β_2 control the influence of the relation and the connection sparsity regularization terms. To explore the impact of β_1 and β_2, we vary each in the range from 0 to 1 at the step size of 0.1, and summarize the results on ACM and Amazon in Figure 6(c) and 7(c), respectively. Similar to the former hyper-parameters, the performance on ACM is more sensitive to β_1 and β_2 than that of Amazon. Generally, NDS achieves the optimal performance at a moderate value of $\beta_1/\beta_2 = 0.5$.

5 CONCLUSION

In this work, we study the problem of how to design powerful HGNNs under the guidance of node-dependent semantics. To address the problem, we propose to search for the semantic structures in terms of relation selection and connection selection, which can guide the local and high-order development of HGNN architectures. Furthermore, we design the predictors to perform node-dependent semantic search. Experimental results verify that the proposed NDS consistently outperforms the state-of-the-art methods on node classification and recommendation tasks.

REFERENCES

[1] Ye Bi, Liqiang Song, Mengqiu Yao, Zhenyu Wu, Jianming Wang, and Jing Xiao. 2020. A heterogeneous information network based cross domain insurance recommendation system for cold start users. In *SIGIR*. 2211–2220.

[2] Yuwei Cao, Hao Peng, Jia Wu, Yingtong Dou, Jianxin Li, and Philip S Yu. 2021. Knowledge-preserving incremental social event detection via heterogeneous gnns. In *TheWebConf*. 3383–3395.

[3] Jiamin Chen, Jianliang Gao, Yibo Chen, Moctard Babatounde Oloulade, Tengfei Lyu, and Zhao Li. 2021. Graphpas: Parallel architecture search for graph neural networks. In *SIGIR*. 2182–2186.

[4] Jingfan Chen, Guanghui Zhu, Haojun Hou, Chunfeng Yuan, and Yihua Huang. 2022. AutoGSR: Neural architecture search for graph-based session recommendation. In *SIGIR*. 1694–1704.

[5] An-Chieh Cheng, Chieh Hubert Lin, Da-Cheng Juan, Wei Wei, and Min Sun. 2020. Instanas: Instance-aware neural architecture search. In *AAAI*. 3577–3584.

[6] Yuhui Ding, Quanming Yao, Huan Zhao, and Tong Zhang. 2021. DiffMG: Differentiable Meta Graph Search for Heterogeneous Graph Neural Networks. In *KDD*. 279–288.

[7] Yuan Fang, Wenqing Lin, Vincent W Zheng, Min Wu, Kevin Chen-Chuan Chang, and Xiao-Li Li. 2016. Semantic proximity search on graphs with metagraph-based learning. In *ICDE*. 277–288.

[8] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *TheWebConf*. 2331–2341.

[9] Yang Gao, Hong Yang, Peng Zhang, Chuan Zhou, and Yue Hu. 2020. Graph Neural Architecture Search.. In *IJCAI*. 1403–1409.

[10] Yang Gao, Peng Zhang, Zhao Li, Chuan Zhou, Yongchao Liu, and Yue Hu. 2021. Heterogeneous Graph Neural Architecture Search. In *ICDM*. 1066–1071.

[11] Floris Geerts, Filip Mazowiecki, and Guillermo Perez. 2021. Let's agree to degree: Comparing graph convolutional networks in the message-passing framework. In *ICML*. 3640–3649.

[12] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *ICML*. 1263–1272.

[13] Weili Guan, Fangkai Jiao, Xuemeng Song, Haokun Wen, Chung-Hsing Yeh, and Xiaojun Chang. 2022. Personalized Fashion Compatibility Modeling via Metapath-guided Heterogeneous Graph Learning. In *SIGIR*. 482–491.

[14] Xiaotian Han, Chuan Shi, Senzhang Wang, S Yu Philip, and Li Song. 2018. Aspect-Level Deep Collaborative Filtering via Heterogeneous Information Networks.. In *IJCAI*. 3393–3399.

[15] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. 2021. Dynamic neural networks: A survey. *arXiv preprint arXiv:2102.04906* (2021).

[16] Zhenyu Han, Fengli Xu, Jinghan Shi, Yu Shang, Haorui Ma, Pan Hui, and Yong Li. 2020. Genetic Meta-Structure Search for Recommendation on Heterogeneous Information Network. In *CIKM*. 455–464.

[17] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *TheWebConf*. 2704–2710.

[18] Zhao Huan, Yao Quanming, and Tu Weiwei. 2021. Search to aggregate neighborhood for graph neural network. In *ICDE*. 552–563.

[19] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q Weinberger. 2018. Multi-scale dense networks for resource efficient image classification. In *ICLR*.

[20] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *CVPR*. 4700–4708.

[21] Zhipeng Huang, Yudian Zheng, Reynold Cheng, Yizhou Sun, Nikos Mamoulis, and Xiang Li. 2016. Meta structure: Computing relevance in large heterogeneous information networks. In *KDD*. 1595–1604.

[22] Jiarui Jin, Jiarui Qin, Yuchen Fang, Kounianhua Du, Weinan Zhang, Yong Yu, Zheng Zhang, and Alexander J Smola. 2020. An efficient neighborhood-based interaction model for recommendation on heterogeneous graph. In *KDD*. 75–84.

[23] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

[24] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

[25] Kwei-Herng Lai, Daochen Zha, Kaixiong Zhou, and Xia Hu. 2020. Policy-gnn: Aggregation optimization for graph neural networks. In *KDD*. 461–471.

[26] Chao Li, Hao Xu, and Kun He. 2022. Differentiable Meta Multigraph Search with Partial Message Propagation on Heterogeneous Information Networks. *arXiv preprint arXiv:2211.14752* (2022).

[27] Dongyue Li, Tao Yang, Lun Du, Zhezhi He, and Li Jiang. 2021. AdaptiveGCN: Efficient GCN Through Adaptively Sparsifying Graphs. In *CIKM*. 3206–3210.

[28] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. 2019. Deepgcns: Can gcns go as deep as cnns?. In *ICCV*. 9267–9276.

[29] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2019. Darts: Differentiable architecture search. In *ICLR*.

[30] Siwei Liu, Iadh Ounis, Craig Macdonald, and Zaiqiao Meng. 2020. A heterogeneous graph neural model for cold-start recommendation. In *SIGIR*. 2029–2032.

[31] Zemin Liu, Yuan Fang, Chenghao Liu, and Steven CH Hoi. 2021. Node-wise localization of graph neural networks. In *IJCAI*.

[32] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks. In *KDD*. 1150–1160.

[33] Xiaojun Ma, Junshan Wang, Hanyue Chen, and Guojie Song. 2021. Improving Graph Neural Networks with Structural Adaptive Receptive Fields. In *TheWebConf*. 2438–2447.

[34] 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 *ESWC*. Springer, 593–607.

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

[36] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. In *VLDB*. 992–1003.

[37] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. 2016. Branchynet: Fast inference via early exiting from deep neural networks. In *ICPR*. 2464–2469.

[38] Andreas Veit and Serge Belongie. 2018. Convolutional networks with adaptive inference graphs. In *ECCV*. 3–18.

[39] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.

[40] Chi Wang, Rajat Raina, David Fong, Ding Zhou, Jiawei Han, and Greg Badros. 2011. Learning relevance from heterogeneous social network and its application in online targeting. In *SIGIR*. 655–664.

[41] Weiqing Wang, Hongzhi Yin, Xingzhong Du, Wen Hua, Yongjun Li, and Quoc Viet Hung Nguyen. 2019. Online user representation learning across heterogeneous social networks. In *SIGIR*. 545–554.

[42] Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and Philip S Yu. 2020. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. *arXiv preprint arXiv:2011.14867* (2020).

[43] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *TheWebconf*. 2022–2032.

[44] Zhen Wang, Zhewei Wei, Yaliang Li, Weirui Kuang, and Bolin Ding. 2022. Graph Neural Networks with Node-wise Architecture. In *KDD*. 1949–1958.

[45] Lanning Wei, Zhiqiang He, Huan Zhao, and Quanming Yao. 2023. Search to Capture Long-range Dependency with Stacking GNNs for Graph Classification. In *TheWebConf*. 588–598.

[46] 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.

[47] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. 2019. SNAS: stochastic neural architecture search. In *ICLR*.

[48] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks?. In *ICLR*.

[49] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *ICML*. 5453–5462.

[50] 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* (2020).

[51] Jiaxuan You, Zhitao Ying, and Jure Leskovec. 2020. Design space for graph neural networks. In *NeurIPS*. 17009–17021.

[52] Tan Yu, Yi Yang, Yi Li, Lin Liu, Hongliang Fei, and Ping Li. 2021. Heterogeneous attention network for effective and efficient cross-modal retrieval. In *SIGIR*. 1146–1156.

[53] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. In *NeurIPS*. 11983–11993.

[54] Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. 2019. Star-gcn: Stacked and reconstructed graph convolutional networks for recommender systems. In *IJCAI*.

[55] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *NeurIPS*.

[56] Wentao Zhang, Mingyu Yang, Zeang Sheng, Yang Li, Wen Ouyang, Yangyu Tao, Zhi Yang, and Bin Cui. 2021. Node Dependent Local Smoothing for Scalable Graph Learning. In *NeurIPS*. 20321–20332.

[57] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *KDD*. 635–644.

[58] Jun Zhao, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. 2019. Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In *KDD*. 2347–2357.

[59] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.