Defining and Discovering Hyper-meta-paths for Heterogeneous Hypergraphs

Yaming Yang[1], **Ziyu Zheng**[1], **Weigang Lu**[2], **Zhe Wang**[1], **Xinyan Huang**[3], **Wei Zhao**[1], **Ziyu Guan**[1*]
[1]School of Computer Science and Technology, Xidian University, China
[2]Department of Civil and Environmental Engineering,
The Hong Kong University of Science and Technology, Hong Kong, China
[3]Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education,
School of Artificial Intelligence, Xidian University, China
{yym@, zhengziyu@stu., wglu@stu., zwang@stu., ywzhao@mail., zyguan@}xidian.edu.cn

Abstract

Heterogeneous hypergraph is a kind of structural data that contains multiple types of nodes and multiple types of hyperedges. Each hyperedge type corresponds to a specific multi-ary relation (called hyper-relation) among subsets of nodes, which goes beyond traditional pair-wise relations in simple graphs. Existing representation learning methods for heterogeneous hypergraphs typically learn embeddings for nodes and hyperedges based on graph neural networks. Although achieving promising performance, they are still limited in capturing more complex structural features and richer semantics conveyed by the composition of various hyper-relations. To fill this research gap, in this work, we propose the concept of hyper-meta-path for heterogeneous hypergraphs, which is defined as the composition of a sequence of hyper-relations. Besides, we design an attention-based heterogeneous hypergraph neural network (HHNN) to automatically learn the importance of hyper-meta-paths. By exploiting useful ones, HHNN is able to capture more complex structural features to boost the model's performance, as well as leverage their conveyed semantics to improve the model's interpretability. Extensive experiments show that HHNN can achieve significantly better performance than state-of-the-art baselines, and the discovered hyper-meta-paths bring good interpretability for the model predictions. To facilitate the reproducibility of this work, we provide our dataset as well as source code at: `https://github.com/zhengziyu77/HHNN`.

1 Introduction

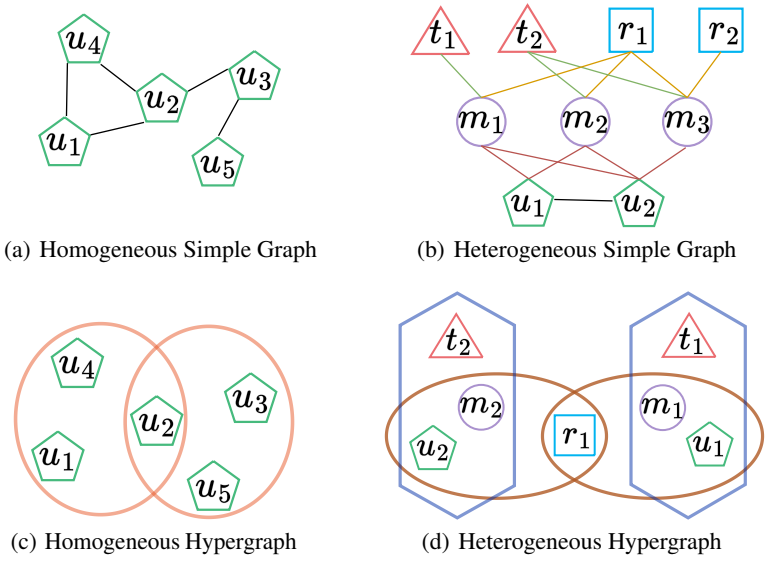
Graph data is a common type of non-independent identically (non-i.i.d.) distributed data in our real-world life, which can be used to describe various complex relations between objects. Representation learning and knowledge mining on graph data are beneficial for many real-world scenarios, such as social networks [29], bioinformatics [42], e-commerce platform [43], and relational databases [13], etc. In this paper, we categorize the existing graph data into four categories based on their complexity, as shown in Figure 1.

❶ **Homogeneous Simple Graphs** consist of one type of nodes and one type of edges, and each edge represents a binary (pair-wise) relation between two nodes. Figure 1(a) shows a toy homogeneous simple graph, which describes five users and their pairwise friendship relations.

❷ **Heterogeneous Simple Graphs** consist of multiple types of nodes and multiple types of edges, with edges representing various types of binary relations between node pairs. Figure 1(b) shows a toy

*Corresponding author

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) Homogeneous Simple Graph (b) Heterogeneous Simple Graph

(c) Homogeneous Hypergraph (d) Heterogeneous Hypergraph

Figure 1: The toy examples of four categories of graph data.

heterogeneous simple graph, which includes four types of nodes: users, movies, tags, and ratings, denoted by different shapes, and four types of binary relations, denoted by different colors of lines.

❸ **Homogeneous Hypergraphs** generalize the binary relations between node pairs to multi-ary (tuple-wise) relations among a set of nodes. The multi-ary relation among a set of nodes is called a hyperedge. In this work, we also refer to the hyperedge type (i.e., the multi-ary relation) as the *hyper-relation*. Similar to homogeneous simple graphs, homogeneous hypergraphs consist of one type of nodes and one type of hyperedges. Figure 1(c) shows a toy homogeneous hypergraph, which describes a social group (i.e., a hyperedge) among users u_1, u_2, and u_4, and another social group among users u_2, u_3, and u_4.

❹ **Heterogeneous Hypergraphs** consist of multiple types of nodes and multiple types of hyperedges (hyper-relations), which can describe multiple types of hyper-relations among multiple types of nodes. Figure 1(d) shows a toy heterogeneous hypergraph, which contains four types of nodes (indicated by different shapes) and two types of hyperedges (indicated by different shapes with different colors). The two hyperedges $\{u_2, m_2, t_2\}$ and $\{u_1, m_1, t_1\}$ belong to the first type of hyper-relation, indicating the semantics that "a user associates a movie with a tag". The two hyperedges $\{u_2, m_2, r_1\}$ and $\{u_1, m_1, r_1\}$ belong to the second type of hyper-relation, indicating the semantics that "a user associates a movie with a rating".

We can observe that for heterogeneous hypergraphs, when the cardinality of their hyperedges is two, or the types of their nodes and hyperedges are one, they can be reduced to the other three categories of graphs, i.e., the latter are three special cases of the former. *In this work, we focus on the representation learning of heterogeneous hypergraphs (more than two types of nodes and more than two types of hyperedges) as they are not only the most general but also the most challenging types of graph data.*

At present, several heterogeneous hypergraph representation learning methods [4, 22, 26, 30, 10, 19, 25, 14] have been proposed. They are all graph neural network (GNN)-based methods, and follow the typical framework of GNNs by first aggregating information from nodes to hyperedges and then aggregating information from hyperedges back to nodes. They have achieved promising performance and have been successfully applied to practical scenarios such as social recommendation [22, 26], spatiotemporal activity prediction [25], and complex system analysis [11]. However, these methods fail to capture more complex structural features and richer semantics conveyed by various types of hyperedges. For the example shown in Figure 1(d), the two hyperedge types convey two different semantics as we have explained in Paragraph ❹. In addition, the two hyperedge types interact with each other through the shared node types (users and movies, i.e., the overlapping area of blue circles and brown circles), resulting in more complex semantics.

In this work, we aim to capture the rich semantics contained in heterogeneous hypergraphs. To this end, let us first review an important concept in heterogeneous simple graphs, i.e., meta-path, which is defined as a sequence of binary relations between nodes [31]. Referring to the example shown in Figure 1(b), let us use "R_1" to denote the binary relation of "users watch movies", and use "R_2" to denote the binary relation of "movies have tags". Then, we can compose the metapath "$R_1 \diamond R_2$", where "$\diamond$" denotes the composition operator. This meta-path describes the new composite binary relation between users and tags: "users watch movies that have specific tags". Meta-paths are often represented by node types as well. Here, the meta-path can also be denoted as $U \xrightarrow{R_1} M \xrightarrow{R_2} T$, where U, M, and T denote the node types of users, movies, and tags, respectively. It is noteworthy that this meta-path is very different from the hyper-relation $\{U, M, T\}$, since from the meta-path, we cannot know who assigned these tags to the movie, while the hyper-relation conveys a more precise semantics of "users associate movies with specific tags". Meta-path has been proven to be very useful in capturing heterogeneous structural features from heterogeneous simple graphs [31, 8, 35, 40]. Unfortunately, it can only handle heterogeneous *simple graphs* and cannot be directly applied to heterogeneous *hypergraphs* due to the challenge of the multi-arity of hyper-relations.

To fill this research gap, we innovatively propose a novel concept called *hyper-meta-path*, which is formally defined as a sequence of hyper-relations, in which each pair of adjacent hyper-relations shares some common node types. It describes the composite relation of these hyper-relations. In this view, hyper-meta-path is defined in the spirit of the meta-path, with the former being a generalization of the latter. Hence, hyper-meta-path can capture more complex structural features and richer semantic information in heterogeneous hypergraphs.

Further, we develop a GNN-based representation learning model called Heterogeneous Hypergraph Neural Network (**HHNN**), which is able to automatically learn the importance of hyper-meta-paths. Each model layer goes through three levels of attention aggregation blocks to refine the representations of nodes and hyperedges. Finally, useful hyper-meta-paths can be discovered based on the optimized attention distributions. The merits are twofold. On the one hand, these hyper-meta-paths help the model capture more complex and subtle high-order structural features, boosting the performance of downstream analysis tasks. On the other hand, the rich semantics conveyed by these hyper-meta-paths endow the model with self-interpretability. That is, these semantics can bring some interesting interpretations to the model's predictions.

The main contributions of this work are summarized as follows:

- We are the first to define the concept of hyper-meta-path for heterogeneous hypergraphs. It describes the composite relation among a sequence of hyper-relations, capturing more complex structural features and richer semantics.
- We design a novel network architecture called Heterogeneous Hypergraph Neural Network (HHNN), which can automatically learn the importance of hyper-meta-paths. By discovering and exploiting useful hyper-meta-paths, HHNN is able to achieve higher performance as well as better self-interpretability.
- We conduct extensive experiments to study the effectiveness of the proposed concept of hyper-meta-paths and the proposed HHNN model. It turns out that HHNN can achieve significantly better performance than state-of-the-art baselines, and the discovered hyper-meta-paths bring good interpretability for the model predictions.

2 Related Work

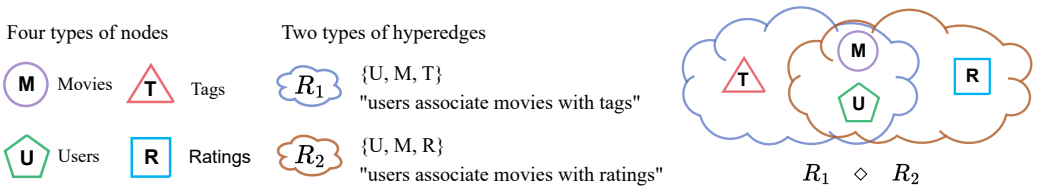
Homogeneous Hypergraph Neural Networks. Inspired by the graph convolutional network (GCN) [24] for homogeneous simple graphs, HGNN [12] is the first work to define the spectral convolution on hypergraphs based on the hypergraph Laplacian, effectively extending GCN to homogeneous hypergraphs. HyperGCN [38] approximates each hyperedge by a set of pairwise edges that connect the nodes in the hyperedge. Inspired by the graph attention network (GAT) [33] for homogeneous simple graphs, Hyper-SAGNN [41] develops a self-attention based on the hypergraph neural network to deal with hypergraphs with variable hyperedge size. Inspired by the GraphSAGE model [17] for homogeneous simple graphs, HyperSAGE [2] proposes an inductive hypergraph learning framework that can capture the intra-relations (within a hyperedge) as well as inter-relations (across hyperedges). HNHN [7] introduces a generalized normalization aggregation scheme of GNN,

which weights the contributions of nodes/hyperedges by a power of their degrees, depending on a real-valued parameter. UniGNN [21] is proposed as a unified hypergraph learning framework, which generalizes several classic GNNs, such as GCN [24], GAT [33], GIN [37], and GraphSAGE [17] to hypergraphs. AllSet [6] implements hypergraph neural network layers as compositions of two multiset functions. ED-HNN [34] is a hypergraph neural network that can approximate any continuous equivariant hypergraph diffusion operators, and thus it can model a wide range of higher-order relations. These methods can effectively address the high-order structural features of hypergraphs, but they have limitations in capturing the heterogeneity of heterogeneous hypergraphs, since there may be multiple types of nodes as well as multiple types of hyperedges.

Heterogeneous Hypergraph Neural Networks. At present, there exists a series of heterogeneous hypergraph neural network methods [4, 30, 14, 25, 26, 10, 22, 19, 5, 18, 36], and the representative ones are introduced as follows. HHNE [4] is a GCN-based hypergraph neural network that can consider the heterogeneity of nodes by projecting different types of node features into a common feature space. HWNN [30] transforms heterogeneous simple graphs into a series of hypergraph snapshots based on a set of user-specified meta-paths. Then it conducts hypergraph convolution based on the Wavelet basis. HGNN+ [14] extends HGNN [12] to heterogeneous hypergraphs by introducing four ways to generate different types of hyperedges. DisenHCN [25] disentangles the user representations into different aspects (location-aware, time-aware, and activity-aware) and aggregates corresponding aspects' features on the constructed hypergraph. These methods not only can capture high-order structural features of hypergraphs but also can address the heterogeneity of nodes and hyperedges by various strategies. However, as we analyzed previously, they fail to capture the complex structural features and rich semantics conveyed by hyper-meta-paths.

3 Preliminaries

In this section, we first define some basic notations and concepts about heterogeneous hypergraphs. Then, we give the formal definition of our proposed concept of hyper-network-schema and hyper-meta-path, as illustrated by Figure 2.



(a) Hyper-network-schema (b) Hyper-meta-path

Figure 2: The illustration of hyper-network-schema and hyper-meta-path of the toy heterogeneous hypergraph shown in Figure 1(d). Note that we use "R" to denote node type "Ratings", and use "R_x" to denote hyperedge type x, i.e., hyper-relation x.

Heterogeneous Hypergraphs. A hypergraph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{H}, \mathbf{W}, \phi, \psi)$, where $\mathcal{V}$ is a set of nodes, $\mathcal{E}$ is a set of hyperedges, and $\mathbf{H} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{E}|}$ is an incidence matrix that describes the belonging relations between node and hyperedges, with the entry $\mathbf{H}_{i,j} = 1$ indicating that node i is in the hyperedge j. The diagonal matrix $\mathbf{W} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$ stores the weights of hyperedges. ϕ: $\mathcal{V} \to \mathcal{A}$ is a node type mapping function, and ψ: $\mathcal{E} \to \mathcal{R}$ is a hyperedge type mapping function, with $|\mathcal{A}| + |\mathcal{R}| > 2$. Let $\mathbf{v}_i$ and $\mathbf{e}_j$ denote the representation vectors of node i and hyperedge j, respectively. Each hyperedge type $R_k \in \mathcal{R}$ conveys a specific multi-ary relation among all the associated, and thus we also call it hyper-relation R_k.

Hyper-network-schema. We define the concept of hyper-network-schema as a tuple $< \mathcal{A}, \mathcal{R} >$, describing the types of the nodes and the types of the hyperedges in a hypergraph, respectively.

A hypergraph $\mathcal{G}$ is homogeneous when $|\mathcal{A}| = 1$ as well as $|\mathcal{R}| = 1$. *In this work, we particularly focus on a more challenging setting by considering heterogeneous hypergraphs with $|\mathcal{A}| > 1$ as well as $|\mathcal{R}| > 1$, i.e., both the nodes and the hyperedges are heterogeneous.*

In Figure 2(a), we show the hyper-network-schema of the toy heterogeneous hypergraph that is previously shown in Figure 1(d). We can clearly see that there are four types of nodes, and two types of hyperedges, i.e., two hyper-relations in the heterogeneous hypergraph.

Hyper-meta-path. For each hyper-relation R_k (the hyperedge type is k), we can denote it as its associated node types, i.e., $R_k = \{\phi(i)|\mathbf{H}_{i,j} = 1, \psi(j) = k, \forall i \in \mathcal{V}, \forall j \in \mathcal{E}, k \in \mathcal{R}\}$ to denote all the associated node types. Then, a hyper-meta-path with length l is defined as a sequence of hyper-relations: $\mathcal{P} = R_1 \diamond R_2 \diamond \cdots \diamond R_l$, where $\diamond$ denotes the composition operator between hyper-relations, and two hyper-relations R_x and R_y can be composed if and only if: $R_x \cap R_y \neq \emptyset$.

In Figure 2(b), we intuitively show a hyper-meta-path $R_1 \diamond R_2$ according to the hyper-network-schema shown in Figure 2(a). As we can see, the hyper-meta-path describes a more complex hyper-relation by compositing the two hyper-relations R_1 and R_2, since they share the same node types, i.e., $R_1 \cap R_2 = \{M, U\}$. Besides, according to the semantics of R_1 and R_2, the hyper-meta-path conveys a richer semantics of "users associate specific movies with specific tags and specific ratings".

4 Methodology

The aggregation scheme of HHNN is shown in Figure 3. Firstly, it projects node features into a hyperedge-specific feature space, and leverages the α-Attention to aggregate features from nodes to hyperedges. Then, it projects hyperedge features back to node-specific feature space, and leverages the β-Attention and γ-Attention to respectively perform intra-hyperedge-type aggregation and inter-hyperedge-type aggregation, resulting in the updated node features.

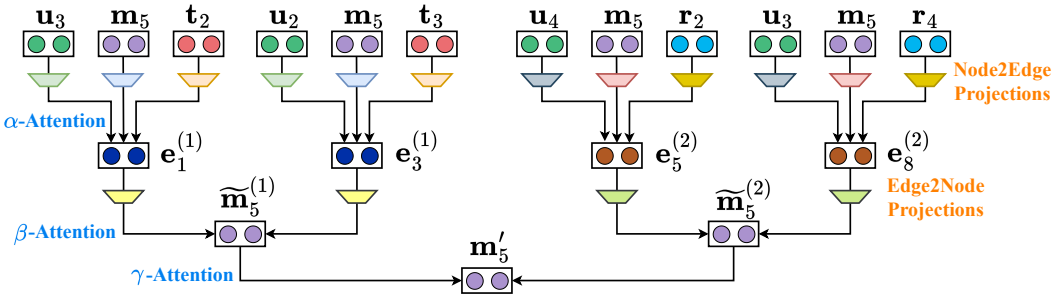


Figure 3: The aggregation scheme of HHNN.

4.1 Aggregation from Nodes to Hyperedges

Each hyperedge contains multiple nodes, and we define a set to describe all the nodes contained in hyperedge j:

$$\mathcal{N}_j^{\mathcal{V}} = \{i|\mathbf{H}_{i,j} = 1, i \in \mathcal{V}, j \in \mathcal{E}\} \tag{1}$$

In a heterogeneous hypergraph, the nodes in $\mathcal{N}_j^{\mathcal{V}}$ may have various types. To aggregate the features of these nodes, we need to first project them into a common feature space through parameter matrices, and these projection parameter matrices are specific to both the involved node type $\phi(i)$ and hyperedge type $\psi(j)$. Then, we leverage the attention mechanism to aggregate the projected node features together, resulting in the updated feature of hyperedge j, which is formally described as follows:

$$\mathbf{e}_j = \sum_{i \in \mathcal{N}_j^{\mathcal{V}}} \alpha_{j,i} \cdot \mathbf{W}^{\psi(j) \leftarrow \phi(i)} \cdot \mathbf{v}_i \tag{2}$$

where $\mathbf{W}^{\psi(j) \leftarrow \phi(i)}$ is a learnable parameter matrix and its superscript "$\psi(j) \leftarrow \phi(i)$" denotes projecting features from nodes of type $\phi(i)$ to hyperedges of type $\psi(j)$, $\alpha_{j,i}$ is the attention aggregation weight from node i to hyperedge j, which is described as follows:

$$\alpha_{j,:} = \textbf{Attention}_{\theta_{\psi(j)}}\left(\{\mathbf{W}^{\psi(j) \leftarrow \phi(i)} \cdot \mathbf{v}_i\}_{i \in \mathcal{N}_j^{\mathcal{V}}}\right) \tag{3}$$

where $\theta_{\psi(j)}$ contains the learnable parameters of the attention function, which are specific to hyperedge type $\psi(j)$ and serve as the attention query. The projected node representations can be mapped into the attention keys and attention values based on different projection parameter matrices.

4.2 Aggregation from Hyperedges to Nodes

In a heterogeneous hypergraph, a node may belong to multiple hyperedges of various types. For the toy example in Figure 1(d), node m_2 belongs to one hyperedge with "blue" type, and one hyperedge with "brown" type. Node r_1 belongs to two hyperedges with "brown" type. To this end, in the hyperedge-to-node aggregation phase, different from most previous approaches, we decompose the aggregation process into two levels, i.e., *intra-type aggregation* and *inter-type aggregation*. In the following, we describe the procedures by taking node i as an example.

4.2.1 Intra-type Aggregation

For node i, assuming that it participates in multiple hyperedges of type k, we define a set to describe these hyperedges:

$$\mathcal{N}_i^{\mathcal{E}_k} = \{j | \mathbf{H}_{i,j} = 1, i \in \mathcal{V}, j \in \mathcal{E}, \psi(j) = k\} \tag{4}$$

Then, we leverage the feature projection and the attention mechanism again to aggregate the features of the hyperedges in $\mathcal{N}_i^{\mathcal{E}_k}$, resulting in a temporary representation for node i:

$$\widetilde{\mathbf{v}}_i^k = \sum_{j \in \mathcal{N}_i^{\mathcal{E}_k}} \beta_{i,j} \cdot \mathbf{W}^{\phi(i) \leftarrow \psi(j)} \cdot \mathbf{e}_j \tag{5}$$

where $\mathbf{W}^{\phi(i) \leftarrow \psi(j)}$ is a projection parameter matrix, and the superscript "$\phi(i) \leftarrow \psi(j)$" denotes projecting features from hyperedges of type $\psi(j)$ to nodes of type $\phi(i)$. $\beta_{i,j}$ is the attention aggregation weight from hyperedge j to node i, which is calculated as follows:

$$\beta_{i,:} = \textbf{Attention}_{\theta_{\phi(i)}}\left(\left\{\mathbf{W}^{\phi(i) \leftarrow \psi(j)} \cdot \mathbf{e}_j\right\}_{j \in \mathcal{N}_i^{\mathcal{E}_k}}\right) \tag{6}$$

where $\theta_{\phi(i)}$ are the attention parameters specific to node type $\phi(i)$. Like the previous attention function, the attention parameters $\theta_{\phi(i)}$ serve as the attention query, and the inputs of the function, i.e., the projected hyperedge features, are mapped into attention keys and values.

Similar aggregation processes are conducted for all the possible hyperedge types that node i participates in. Finally, we can obtain a set of temporary representations for node i, denoted as $\left\{\widetilde{\mathbf{v}}_i^k | \mathcal{N}_i^{\mathcal{E}_k} \neq \emptyset, \forall k \in \mathcal{R}\right\}$, and abbreviated as $\left\{\widetilde{\mathbf{v}}_i^k\right\}$. Its element $\mathbf{v}_i^k$ reflects the property of node i from the aspect of hyper-relation R_k.

4.2.2 Inter-type Aggregation

To obtain a more comprehensive representation for node i, we use the attention mechanism once more to fuse the temporary representations of node i from all the involved hyper-relations:

$$\mathbf{v}_i' = \sum_k \gamma_{i,k} \cdot \widetilde{\mathbf{v}}_i^k \tag{7}$$

where $\gamma_{i,k}, \forall k$ are the attention coefficients that are computed by an attention function as follows:

$$\gamma_{i,:} = \textbf{Attention}_{\omega_{\phi(i)}}\left(\left\{\widetilde{\mathbf{v}}_i^k\right\}\right) \tag{8}$$

where $\omega_{\phi(i)}$ are the attention parameters specific to node type $\phi(i)$, and serve as the attention query. The inputs of the attention function, i.e., the obtained temporary node representations, are mapped into the attention keys and values.

In this way, we conduct this three-level attention aggregation in each model layer to update the node representations and the hyperedge representations. Figure 4 shows the overall architecture of HHNN. We can obtain the final node embeddings $\{\mathbf{v}_i | \forall i \in \mathcal{V}\}$, and the final hyperedge embeddings $\{\mathbf{e}_j | \forall j \in \mathcal{E}\}$ from the last layer.

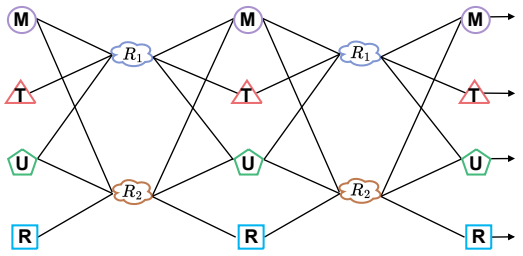


Figure 4: The architecture of HHNN.

4.3 Training Loss

By obtaining the embeddings of nodes and hyperedges, we can implement the loss function according to the task at hand. For example, for the semi-supervised node classification task, we utilize the cross-entropy loss as follows:

$$L = -\sum_{i \in \mathcal{Y}} \mathbf{y}_i \cdot \ln(\mathbf{C} \cdot \mathbf{v}_i) \tag{9}$$

where $\mathbf{v}_i$ and $\mathbf{y}_i$ denote the extracted embedding and the ground-truth label of node i, respectively, $\mathcal{Y}$ denotes the set of node indices that have labels, and $\mathbf{C}$ denotes the parameter matrix of the classifier.

In Appendix D, we show the overall algorithm of HHNN. In Appendix E, we theoretically show that HHNN can automatically discover the most important hyper-meta-path for each node, which helps interpret the model's prediction. In Section 5.3, we verify this ability of HHNN through case studies.

4.4 Time Complexity Analysis

The time complexity of HHNN is analyzed as follows. For each layer, the main cost comes from the three-level attention aggregation operations. The α-attention aggregation involves projecting and attending to nodes within each hyperedge, costing $\mathcal{O}(|E| \cdot d_\alpha \cdot f_\alpha \cdot r)$, where $|E|$ is the number of hyperedges, d_α and f_α are the dimensionalities of the input and output features, respectively, and r denotes the average number of nodes per hyperedge. In the β-attention aggregation phase, each node aggregates its connected hyperedges of each type, costing $\mathcal{O}(|V| \cdot d_\beta \cdot f_\beta \cdot t)$ where $|V|$ is the number of nodes, d_β and f_β are the feature dimensionalities, and t denotes the average number of hyperedges of each type that a node participates in. In the γ-attention aggregation phase, each node fuses its representations across different hyperedge types, costing $\mathcal{O}(|V| \cdot d_\gamma \cdot f_\gamma \cdot k)$, where d_γ and f_γ are the feature dimensionalities, and k is the average number of hyperedge types. Assuming the model has n layers, the total time complexity becomes: $\mathcal{O}(|E| \cdot d_\alpha \cdot f_\alpha \cdot r \cdot n + |V| \cdot d_\beta \cdot f_\beta \cdot t \cdot n + |V| \cdot d_\gamma \cdot f_\gamma \cdot k \cdot n)$. In practical scenarios, $d_\alpha, f_\alpha, d_\beta, f_\beta, d_\gamma, f_\gamma, r, t, k, n$ are usually much smaller constants compared to $|\mathcal{V}|$ and $|\mathcal{E}|$. Therefore, the total time complexity of HHNN is equal to: $\mathcal{O}(|\mathcal{V}| + |\mathcal{E}|)$, which is linear to the size of the hypergraph.

5 Experiment

In this section, we conduct comprehensive experiments on two real-world datasets. Please see Appendix A for dataset details, see Appendix B for the details of the used baseline methods, and see Appendix C for the detailed experimental settings.

5.1 Performance Comparison

Following most previous related studies [12, 14, 38, 7, 6, 21, 34], we compare the performance of all the methods by conducting node classification for the target nodes that are associated with ground-truth labels. The final experimental results are reported in Table 1 and Table 2.

As we can see, our proposed HHNN shows the best performance in all cases. This is due to the fact that HHNN is able to capture more complex and higher-order structural features that are conveyed by the discovered useful hyper-meta-paths. For all the methods, the performance is significantly better when the training ratio is higher, indicating that they can effectively exploit the supervision information. HGNN+ significantly outperforms HGNN in all cases because the former addresses hyperedge heterogeneity in a more nuanced manner. AllSetTsfm performs slightly better than AllDeepSets in most cases, which is also consistent with the similar finding reported in their original paper [6]. For most baselines as well as our HHNN, the experimental results on the Olist dataset are better than those on the Movielens dataset, which is because the target (Product) nodes in Olist are easier to distinguish, and the structural features conveyed by the three types of hyperedges are more informative.

5.2 Ablation Study

As described by Section 4, the key process of HHNN lies in three levels of attention aggregation blocks. As illustrated in the left part of Figure 2(a)), the α-Attention block performs node-level

Table 1: Performance comparison on Movielens.

Metrics	Micro		Macro		AUC	
Training Ratio	20%	40%	20%	40%	20%	40%
CE-GCN	$55.46_{\pm 0.96}$	$57.31_{\pm 0.97}$	$21.99_{\pm 4.32}$	$26.53_{\pm 1.66}$	$67.86_{\pm 2.38}$	$71.26_{\pm 1.68}$
HNHN	$52.53_{\pm 0.79}$	$54.42_{\pm 1.33}$	$15.67_{\pm 1.92}$	$17.36_{\pm 3.00}$	$64.61_{\pm 3.14}$	$67.99_{\pm 2.72}$
HyperGCN	$48.53_{\pm 1.97}$	$49.83_{\pm 1.45}$	$16.55_{\pm 1.48}$	$18.05_{\pm 2.27}$	$57.89_{\pm 1.06}$	$59.87_{\pm 1.96}$
AllDeepSets	$52.39_{\pm 1.13}$	$55.46_{\pm 1.67}$	$15.72_{\pm 2.13}$	$23.13_{\pm 1.69}$	$65.47_{\pm 1.71}$	$70.94_{\pm 1.68}$
AllSetTsfm	$53.39_{\pm 1.84}$	$57.40_{\pm 1.60}$	$18.01_{\pm 5.52}$	$30.32_{\pm 3.13}$	$66.21_{\pm 2.97}$	$73.63_{\pm 2.03}$
UniGNN	$52.35_{\pm 1.08}$	$54.18_{\pm 1.58}$	$16.41_{\pm 2.42}$	$24.11_{\pm 3.43}$	$65.78_{\pm 2.16}$	$71.77_{\pm 1.39}$
EDHNN	$53.41_{\pm 0.60}$	$56.72_{\pm 1.08}$	$18.74_{\pm 2.60}$	$32.56_{\pm 3.22}$	$70.07_{\pm 1.31}$	$77.06_{\pm 1.15}$
HGNN	$52.54_{\pm 1.79}$	$55.30_{\pm 1.69}$	$20.48_{\pm 3.31}$	$25.98_{\pm 4.17}$	$65.25_{\pm 2.50}$	$70.06_{\pm 2.21}$
HGNN+	$53.13_{\pm 2.21}$	$55.44_{\pm 1.76}$	$21.62_{\pm 3.21}$	$26.13_{\pm 2.70}$	$69.62_{\pm 2.60}$	$73.77_{\pm 1.62}$
HHNN	$\mathbf{56.80}_{\pm 1.65}$	$\mathbf{60.65}_{\pm 1.88}$	$\mathbf{26.24}_{\pm 3.90}$	$\mathbf{33.33}_{\pm 3.19}$	$\mathbf{73.21}_{\pm 2.13}$	$\mathbf{78.72}_{\pm 1.60}$

Table 2: Performance comparison on Olist.

Metrics	Micro		Macro		AUC	
Training Ratio	20%	40%	20%	40%	20%	40%
CE-GCN	$32.31_{\pm 0.70}$	$42.97_{\pm 0.62}$	$31.29_{\pm 0.74}$	$42.12_{\pm 0.63}$	$70.20_{\pm 0.30}$	$77.76_{\pm 0.21}$
HNHN	$53.96_{\pm 0.88}$	$60.39_{\pm 0.89}$	$53.40_{\pm 1.30}$	$59.90_{\pm 1.02}$	$84.71_{\pm 0.48}$	$88.05_{\pm 0.49}$
HyperGCN	$21.96_{\pm 0.52}$	$27.96_{\pm 0.79}$	$20.21_{\pm 0.64}$	$25.71_{\pm 1.33}$	$62.08_{\pm 0.62}$	$67.66_{\pm 1.05}$
AllDeepSets	$66.96_{\pm 0.49}$	$75.89_{\pm 0.79}$	$66.95_{\pm 0.46}$	$75.89_{\pm 0.80}$	$90.29_{\pm 0.30}$	$94.44_{\pm 0.24}$
AllSetTsfm	$64.57_{\pm 1.15}$	$75.97_{\pm 0.68}$	$64.82_{\pm 1.18}$	$76.11_{\pm 0.76}$	$88.79_{\pm 0.53}$	$94.31_{\pm 0.25}$
UniGNN	$57.47_{\pm 0.46}$	$68.11_{\pm 0.65}$	$57.13_{\pm 0.47}$	$67.89_{\pm 0.62}$	$87.20_{\pm 0.28}$	$92.36_{\pm 0.28}$
EDHNN	$61.62_{\pm 0.70}$	$70.88_{\pm 1.04}$	$61.35_{\pm 0.68}$	$70.71_{\pm 0.99}$	$89.36_{\pm 0.36}$	$93.47_{\pm 0.35}$
HGNN	$40.66_{\pm 0.61}$	$48.62_{\pm 0.58}$	$40.11_{\pm 0.60}$	$48.38_{\pm 0.62}$	$76.19_{\pm 0.51}$	$80.93_{\pm 0.31}$
HGNN+	$59.03_{\pm 0.80}$	$64.16_{\pm 0.74}$	$59.43_{\pm 0.74}$	$64.63_{\pm 0.75}$	$86.84_{\pm 0.33}$	$89.86_{\pm 0.24}$
HHNN	$\mathbf{72.54}_{\pm 0.54}$	$\mathbf{77.74}_{\pm 0.83}$	$\mathbf{72.89}_{\pm 0.56}$	$\mathbf{77.90}_{\pm 0.80}$	$\mathbf{93.20}_{\pm 0.32}$	$\mathbf{95.39}_{\pm 0.25}$

aggregation to update hyperedge representations. The β-Attention block and γ-Attention block correspond to intra-type hyperedge aggregation and inter-type hyperedge aggregation, respectively. Here, we conduct comprehensive ablation studies to investigate the effectiveness of these three attention blocks. The experimental results are shown in Table 3. We use the binary bit "0" or "1" to indicate whether the corresponding attention blocks are activated. When an attention block is deactivated, we replace the attention aggregation with the mean aggregation. By considering all possible combinations of the three blocks, we obtain eight variants of HHNN. In the first column of Table 3, we also use three-bit binary numbers to denote different variants.

As we can see, in all the cases, the variant "111" achieves the best performance, which corresponds to the full version of our HHNN. This indicates that the power of HHNN can only be fully unleashed when all three attention blocks are assembled together, which implies that these three attention blocks can enhance each other. Besides, variant "010" performs better than variants "001" and "100", and variant "101" performs worse than variants "011" and "110", indicating that the β-Attention block is very effective. This indicates that the different hyperedges a node belongs to have significantly different levels of importance for it.

5.3 Hyper-meta-path Discovery and Interpretability Study

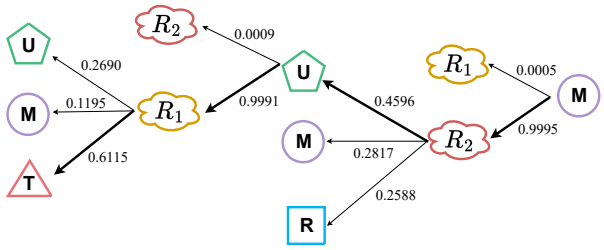
One of the merits of our HHNN is that it can discover useful hyper-meta-paths for the concerned task (see Appendix E). Here, we show this ability of HHNN by analyzing the learned attention coefficients after finishing the model training. In Figure 5(a) and Figure 5(b), we show the learned attention weights that are associated with the movie node m_{2934} on Movielens, and the product node

Table 3: The performance of different variants of our HHNN.

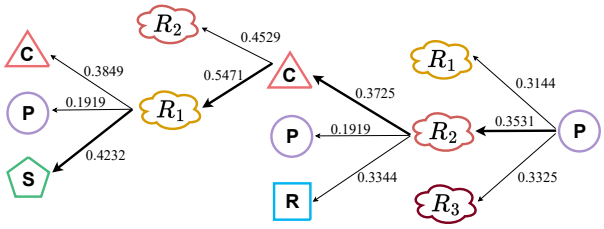
Variants Modes	Movielens Micro	Movielens Macro	Movielens AUC	Olist Micro	Olist Macro	Olist AUC
000	$55.37_{\pm 1.81}$	$27.87_{\pm 4.63}$	$74.09_{\pm 1.78}$	$71.79_{\pm 0.87}$	$71.79_{\pm 0.81}$	$93.33_{\pm 0.25}$
001	$56.81_{\pm 1.70}$	$30.63_{\pm 5.41}$	$75.60_{\pm 1.61}$	$68.44_{\pm 0.69}$	$68.72_{\pm 0.67}$	$92.17_{\pm 0.34}$
010	$58.97_{\pm 1.73}$	$34.21_{\pm 4.60}$	$78.44_{\pm 1.74}$	$76.96_{\pm 0.66}$	$77.03_{\pm 0.66}$	$95.38_{\pm 0.18}$
011	$60.29_{\pm 1.42}$	$35.45_{\pm 4.78}$	$79.28_{\pm 1.58}$	$74.74_{\pm 0.63}$	$74.89_{\pm 0.62}$	$94.87_{\pm 0.22}$
100	$57.34_{\pm 2.66}$	$30.55_{\pm 5.14}$	$75.32_{\pm 1.75}$	$63.60_{\pm 0.90}$	$64.02_{\pm 0.79}$	$90.58_{\pm 0.36}$
101	$59.26_{\pm 2.40}$	$35.45_{\pm 3.04}$	$77.87_{\pm 1.88}$	$71.19_{\pm 0.96}$	$71.38_{\pm 0.91}$	$93.24_{\pm 0.42}$
110	$61.42_{\pm 1.63}$	$36.22_{\pm 4.79}$	$80.12_{\pm 2.24}$	$77.28_{\pm 0.61}$	$77.41_{\pm 0.66}$	$95.73_{\pm 0.18}$
111	$\mathbf{62.95}_{\pm 1.94}$	$\mathbf{39.03}_{\pm 2.32}$	$\mathbf{81.87}_{\pm 1.85}$	$\mathbf{79.82}_{\pm 0.71}$	$\mathbf{79.91}_{\pm 0.78}$	$\mathbf{96.32}_{\pm 0.21}$

p_{4593} on Olist, respectively. In the figures, the magnitude of the attention weights is reflected by the thickness of the corresponding arrows, and the notations in the figures are described in Table 4.

In Figure 5(a), for the target movie node, the most important path can be denoted as "T $\xleftarrow{R_1}$ U $\xleftarrow{R_2}$ M". Referring to the definition in Section 3, in the path, the hyper-relation R_1 (i.e., hyperedge type {UMT}) and the hyper-relation R_2 (i.e., hyperedge type {UMR}) are composited based on the movie node shared between them, forming the hyper-meta-path "$R_1 \diamond R_2$". Observing the path again, hyper-meta-path "$R_1 \diamond R_2$" connects the leftmost tag node to the rightmost movie (i.e., target) node. The full semantics conveyed by "$R_1 \diamond R_2$" can be described as: "the leftmost tag node participates in a {UMT} hyperedge" and "the rightmost movie node participates in a {UMR} hyperedge" and "the two hyperedges share the same user node", which can serve as the interpretation for the prediction result of the target movie node.



(a) Movielens



(b) Olist

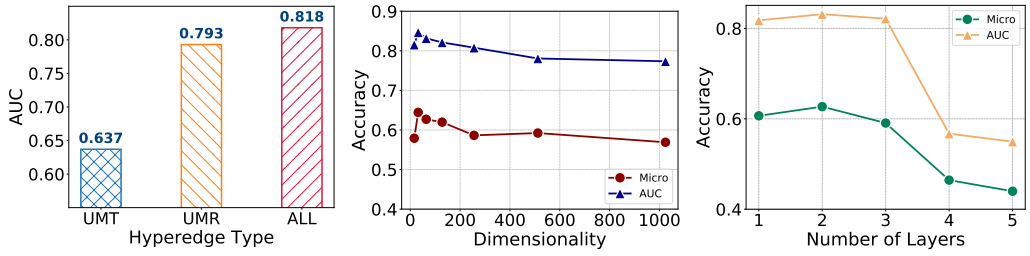
Figure 5: Discovered hyper-meta-paths by HHNN.

Similarly, in Figure 5(b), the discovered most important hyper-meta-path can be denoted as "S $\xleftarrow{R_1}$ C $\xleftarrow{R_2}$ P". In the path, the hyper-relation R_1 (i.e., hyperedge type {CPS}) and the hyper-relation R_2 (i.e., hyperedge type {CPR}) are composited based on the customer node shared between them, forming the hyper-meta-path "$R_1 \diamond R_2$". The semantics of the hyper-meta-path can be described as: "the leftmost seller node participates in a {CPS} hyperedge" and "the rightmost product node participates in a {CPR} hyperedge" and "the two hyperedges share the same customer node". This semantics can help interpret the prediction result of the target product node.

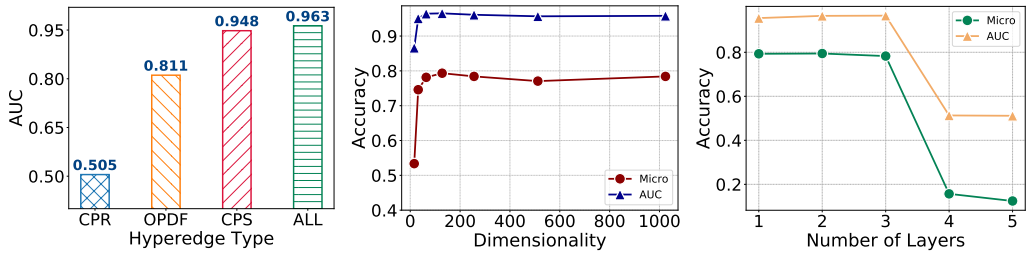
5.4 Analysis of Different Hyperedges

We investigate the impact of the number of hyperedge types on HHNN performance. In Figure 6(a) and Figure 6(d), we show the performance of HHNN by exploiting each single type of hyperedges, as well as the performance by exploiting all types of hyperedges (marked by "ALL" in the legend).

As we can see, on both datasets, HHNN achieves the best performance when exploiting all types of hyperedges, which is due to the fact that HHNN can well handle the heterogeneity of hyperedges in an adaptive way, and it can compose more complex structural features by combining these hyperedges.



(a) Movielens (b) Movielens (c) Movielens



(d) Olist (e) Olist (f) Olist

Figure 6: Hyper-parameter study on Movielens and Olist.

5.5 Hyper-parameter Study

Our proposed HHNN does not have special hyper-parameters. Here, we study the sensitivity of the hidden dimensionality and the number of model layers on the two datasets, and the results are shown in Figure 6.

Referring to Figure 6(b) and Figure 6(e), the model performance is poor when the hidden dimensionality is too small, which indicates that the model suffers from underfitting. HHNN shows the best results when the dimensionalities are equal to 64 on Movielens and 128 on Olist, respectively. After that, the model's performance gradually decreases as the dimensionality increases, which is caused by the overfitting issue.

Referring to Figure 6(c) and Figure 6(f), HHNN achieves its best performance when it has two model layers, and the model maintains strong performance even at 3 layers, with no significant performance drop, indicating robustness against over-smoothing. After that, the model's performance decreases significantly when the model goes deeper, which is caused by the overfitting issue of the hypergraph neural network. Notably, each HHNN layer includes two projection and aggregation steps (node-to-hyperedge and back), which actually correspond to two layers in standard GNN models. Therefore, a 3-layer HHNN is equivalent to a 6-layer standard GNN, making it sufficiently deep to capture complex structural patterns without degradation.

6 Conclusion

In this work, we first define a novel concept called hyper-meta-path for heterogeneous hypergraphs. It not only describes more complex structural features but also conveys richer semantic information. Then, we design a three-level attention-based heterogeneous hypergraph neural network called HHNN to automatically learn the importance of hyper-meta-paths. By discovering and exploiting useful ones, HHNN can achieve higher performance, and the semantics conveyed by these hyper-meta-paths can enhance the model interpretability of HHNN.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 62303366, 62425605, and 62133012, in part by the Key Research and Development Program of Shaanxi under Grants 2025CY-YBXM-041, 2022ZDLGY01-10, and 2024CY2-GJHX-15, and in part by the Fundamental Research Funds for the Central Universities under Grants ZYTS25211 and ZYTS25086.

References

[1] Alessia Antelmi, Gennaro Cordasco, Mirko Polato, Vittorio Scarano, Carmine Spagnuolo, and Dingqi Yang. A survey on hypergraph representation learning. *ACM Computing Surveys*, 56(1):1–38, 2023.

[2] Devanshu Arya, Deepak K Gupta, Stevan Rudinac, and Marcel Worring. Hypersage: Generalizing inductive representation learning on hypergraphs. *arXiv preprint arXiv:2010.04558*, 2020.

[3] Song Bai, Feihu Zhang, and Philip HS Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

[4] Inci M Baytas, Cao Xiao, Fei Wang, Anil K Jain, and Jiayu Zhou. Heterogeneous hyper-network embedding. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 875–880. IEEE, 2018.

[5] Rui Bing, Guan Yuan, Yanmei Zhang, Senzhang Wang, Bohan Li, and Yong Zhou. An efficient multi-view heterogeneous hypergraph convolutional network for heterogeneous information network representation learning. *IEEE Transactions on Big Data*, 2024.

[6] Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. In *International Conference on Learning Representations*, 2022.

[7] Yihe Dong, Will Sawin, and Yoshua Bengio. Hnhn: Hypergraph networks with hyperedge neurons. *arXiv preprint arXiv:2006.12278*, 2020.

[8] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *SIGKDD*, pages 135–144. ACM, 2017.

[9] Iulia Duta, Giulia Cassarà, Fabrizio Silvestri, and Pietro Liò. Sheaf hypergraph networks. *Advances in Neural Information Processing Systems*, 36:12087–12099, 2023.

[10] Haoyi Fan, Fengbin Zhang, Yuxuan Wei, Zuoyong Li, Changqing Zou, Yue Gao, and Qionghai Dai. Heterogeneous hypergraph variational autoencoder for link prediction. *IEEE transactions on pattern analysis and machine intelligence*, 44(8):4125–4138, 2021.

[11] Li Feng, Huiying Gong, Shen Zhang, Xiang Liu, Yu Wang, Jincan Che, Ang Dong, Christopher H Griffin, Claudia Gragnoli, Jie Wu, et al. Hypernetwork modeling and topology of high-order interactions for complex systems. *Proceedings of the National Academy of Sciences*, 121(40):e2412220121, 2024.

[12] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3558–3565, 2019.

[13] Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Position: Relational deep learning-graph representation learning on relational databases. In *International Conference on Machine Learning*, pages 13592–13607. PMLR, 2024.

[14] Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3181–3199, 2022.

[15] Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2548–2566, 2020.

[16] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.

[17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, pages 1024–1034, 2017.

[18] Malik Khizar Hayat, Shan Xue, Jia Wu, and Jian Yang. Heterogeneous hypergraph embedding for node classification in dynamic networks. *IEEE Transactions on Artificial Intelligence*, 2024.

[19] Aiping Huang, Zihan Fang, Zhihao Wu, Yanchao Tan, Peng Han, Shiping Wang, and Le Zhang. Multi-view heterogeneous graph learning with compressed hypergraph neural networks. *Neural Networks*, 179:106562, 2024.

[20] Jie Huang, Xin Liu, and Yangqiu Song. Hyper-path-based representation learning for hypernetworks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 449–458, 2019.

[21] Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 2021.

[22] Bilal Khan, Jia Wu, Jian Yang, and Xiaoxiao Ma. Heterogeneous hypergraph neural network for social recommendation using attention network. *ACM Transactions on Recommender Systems*, 2023.

[23] Sunwoo Kim, Soo Yong Lee, Yue Gao, Alessia Antelmi, Mirko Polato, and Kijung Shin. A survey on hypergraph neural networks: an in-depth and step-by-step guide. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6534–6544, 2024.

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

[25] Yinfeng Li, Chen Gao, Quanming Yao, Tong Li, Depeng Jin, and Yong Li. Disenhcn: Disentangled hypergraph convolutional networks for spatiotemporal activity prediction. *arXiv preprint arXiv:2208.06794*, 2022.

[26] Yongkang Li, Zipei Fan, Jixiao Zhang, Dengheng Shi, Tianqi Xu, Du Yin, Jinliang Deng, and Xuan Song. Heterogeneous hypergraph neural network for friend recommendation with human mobility. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4209–4213, 2022.

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pages 3111–3119, 2013.

[28] Mathilde Papillon, Sophia Sanborn, Mustafa Hajij, and Nina Miolane. Architectures of topological deep learning: A survey of message-passing topological neural networks. *arXiv preprint arXiv:2304.10031*, 2023.

[29] Aravind Sankar, Yozen Liu, Jun Yu, and Neil Shah. Graph neural networks for friend ranking in large-scale social platforms. In *Proceedings of the Web Conference 2021*, pages 2535–2546, 2021.

[30] Xiangguo Sun, Hongzhi Yin, Bo Liu, Hongxu Chen, Jiuxin Cao, Yingxia Shao, and Nguyen Quoc Viet Hung. Heterogeneous hypergraph embedding for graph classification. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 725–733, 2021.

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

[32] Hao Tian and Reza Zafarani. Higher-order networks representation and learning: A survey. *ACM SIGKDD Explorations Newsletter*, 26(1):1–18, 2024.

[33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[34] Peihao Wang, Shenghao Yang, Yunyu Liu, Zhangyang Wang, and Pan Li. Equivariant hypergraph diffusion neural operators. In *The Eleventh International Conference on Learning Representations*, 2023.

[35] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The World Wide Web Conference*, pages 2022–2032, 2019.

[36] Ying Wang, Yingji Li, Yue Wu, and Xin Wang. Exploring multiple hypergraphs for heterogeneous graph neural networks. *Expert Systems with Applications*, 236:121230, 2024.

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

[38] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. *Advances in neural information processing systems*, 32, 2019.

[39] Haoran Yang, Hongxu Chen, Lin Li, S Yu Philip, and Guandong Xu. Hyper meta-path contrastive learning for multi-behavior recommendation. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 787–796. IEEE, 2021.

[40] Yaming Yang, Ziyu Guan, Jianxin Li, Wei Zhao, Jiangtao Cui, and Quan Wang. Interpretable and efficient heterogeneous graph convolutional network. *TKDE*, 2021.

[41] Ruochi Zhang, Yuesong Zou, and Jian Ma. Hyper-sagnn: a self-attention based graph neural network for hypergraphs. *arXiv preprint arXiv:1911.02613*, 2019.

[42] Xiao-Meng Zhang, Li Liang, Lin Liu, and Ming-Jing Tang. Graph neural networks and their current applications in bioinformatics. *Frontiers in genetics*, 12:690049, 2021.

[43] Jun Zhao, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2347–2357, 2019.

[44] Minhao Zou, Zhongxue Gan, Yutong Wang, Junheng Zhang, Dongyan Sui, Chun Guan, and Siyang Leng. Unig-encoder: A universal feature encoder for graph and hypergraph node classification. *Pattern Recognition*, 147:110115, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: See the abstract and Section 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.

- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: See Appendix K.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes] .

   Justification: See Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.

- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes] .

   Justification: See Appendix C, and see our source code at: `https://github.com/zhengziyu77/HHNN`.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] .

   Justification: See Appendix A, and see our source code at: `https://github.com/zhengziyu77/HHNN`.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes] .

   Justification: See Appendix C, and see our source code at: `https://github.com/zhengziyu77/HHNN`.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes] .

   Justification: See the experimental results in Table 1, Table 2, and Table 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes] .

Justification: See Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes] .

Justification: All the experiments of this work conform, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes] .

Justification: See Appendix L.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes] .

    Justification: We have properly cited existing assets in our paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes] .

    Justification: We have provided our well-documented source code at: `https://github.com/zhengziyu77/HHNN`.

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA] .

    Justification: The core method development in this research does not involve LLMs.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Datasets

Most previous studies [12, 7, 2, 6, 18, 21, 14, 38, 30, 3] construct their hypergraphs based on Cora, PubMed, and Citeseer, which originally belong to homogeneous simple graphs (as shown in Figure 1(a)). Several methods [5, 10, 22] also construct their hypergraphs based on DBLP, ACM, MAG, and IMDB, which originally belong to heterogeneous simple graphs (as shown in Figure 1(b)). *In other words, these hypergraphs are transformed from simple graphs, and they essentially describe binary relations rather than multi-ary relations.*

In this work, to fully reflect the effectiveness of our proposed HHNN in capturing complex structural features conveyed by hyper-meta-paths, we have tried our best to construct two genuine heterogeneous hypergraphs, where there are not only multiple types of nodes but also multiple types of real hyper-relations, as illustrated by Figure 1(d). To this end, we have tried our best to construct two new heterogeneous hypergraph benchmark datasets, and their key statistics are listed in Table 4.

Table 4: Dataset statistics.

Datasets	Node Type	Number	Hyperedge Type	Hyperedge Semantics	Number
Movielens	Movies (M)	3439	UMT (R_1)	"a user tags a movie with a specific tag"	9681
	User (U)	2106			
	Tag (T)	3108	UMR (R_2)	"a user rates a movie"	158897
	Rating (R)	10			
Olist	Customer (C)	54321	CPS (R_1)	"a customer purchases a product from a seller"	55630
	Product (P)	18454			
	Seller (S)	2090	CPR (R_2)	"a customer provides a rating for a product"	55556
	Rating (R)	5			
	Origin (O)	489	OPDF (R_3)	"an order is shipped from an origin to a destination with a freight value"	55630
	Destination (D)	3477			
	Freight (F)	14			

•**MovieLens** is a real-world movie dataset, which was originally released by GroupLens[2], a research laboratory at University of Minnesota. Based on the raw dataset, we construct a heterogeneous hypergraph that contains four types of nodes and two types of hyperedges. As we can see, the two hyperedge types (hyper-relations R_1 and R_2) are natural ternary relations. Movie nodes are associated with ground-truth labels, describing their genres, such as comedy, action, animation, etc.

•**Olist** is a real-world e-commercial dataset. It contains the orders of Olist Store, a Brazilian e-commerce platform. The raw dataset was originally released at Kaggle[3], a data science competition platform. Based on the raw dataset, we construct a heterogeneous hypergraph that contains six types of nodes and three types of hyperedges. As shown in the table, the hyper-relations R_1 and R_1 are ternary relations, and the hyper-relation R_1 is a quaternary relation. The product nodes are associated with ground-truth labels that describe their categories, such as perfumery, telephone, automotive, etc.

B Baselines

We select ten representative hypergraph neural network methods as our comparative baselines, including: (1) CE-GCN; (2) HNHN [7]; (3) HyperGCN [38]; (4) AllSetTsfm [6]; (5) AllDeepSets [6]; (6) UniGNN [21]; (7) ED-HNN [34]; (8) HGNN [12]; (9) HGNN+ [14].

Here, CE-GCN is a naive baseline, which first transforms a hypergraph into a simple graph through clique expansion [1], where two nodes form an edge if and only if they are in the same hyperedge. Then, the GCN [24] model is utilized to encode the resulting simple graph. For baseline UniGNN, we adopt its most powerful variant, i.e., UniGCNII. Baselines AllSetTsfm (AllSetTransformer) and AllDeepSets are two variants of the AllSet method [6].

[2] `https://files.grouplens.org/datasets/hetrec2011/hetrec2011-movielens-2k-v2.zip`

[3] `https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce`

C Experimental Settings

We use the Pytorch framework to implement our proposed HHNN. The number of model layers is searched in {1, 2, 3, 4, 5}, the dimensionality of hidden node/hyperedge representations is searched in {8, 16, 32, 64, 128}, the number of attention heads is searched in {1, 2, 4, 8}. We use the Adam optimizer to optimize all the trainable model parameters, which are randomly initialized by the Xavier uniform distribution [16]. For ease of tuning, the optimizer settings are the same for both datasets. Specifically, the learning rate is set to 0.001, the weight decay is set to 0.0, and the attention dropout rate is set to 0.5. For more details about hyper-parameter settings, please see our source code at: `https://github.com/zhengziyu77/HHNN`.

Regarding baselines, we reproduce the experimental results on the two benchmark datasets, based on their official source codes. For fairness, we also try our best to search for the optimal hyper-parameters for them, starting from the default settings of their code or the settings reported in their papers.

For fairness, some of the settings are shared among all the methods. Firstly, they use the same input features. Secondly, they use the same evaluation metrics of Micro F1 (Micro) score, Macro F1 (Macro) score, and Area Under the Curve (AUC) score. Finally, they use the same training/validation/test sets. Specifically, on each dataset, we randomly select $\tau\%$ ground-truth labels as the training set, and the rest $(1-\tau)\%$ are divided equally as the validation set and the test set, where $\tau \in \{20, 40\}$. We randomly repeat all the evaluation tasks ten times and report the mean results with the standard deviation. All the experiments are conducted on Intel(R) Core(TM) i9-10980XE CPU and NVIDIA TITAN RTX GPU with 24GB GPU memory.

D Algorithm

The overall training process of our proposed HHNN is shown in Algorithm 1.

Algorithm 1: The training process of HHNN.

Input : The heterogeneous hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{H}, \mathbf{W}, \phi, \psi)$,
The number of model layers N.
Output : The embeddings of all the nodes and hyperedges.

```
Randomly initialize all the trainable model parameters;
for n = 1, ..., N do
    # from nodes to hyperedges;
    Perform α-Attention aggregation according to Eqs. (1-3);
    # from hyperedges to nodes;
    Perform β-Attention (intra-type) aggregation according to Eqs. (4-6);
    Perform γ-Attention (inter-type) aggregation according to Eqs. (7-8);
end
Compute loss according to Eq. (9);
Update model parameters by gradient descent;
```

E Theoretical Analysis of Hyper-meta-paths

The HHNN model introduces a hierarchical attention mechanism at three levels, i.e., node-to-hyperedge, intra-type hyperedge-to-node, and inter-type hyperedge-to-node, which correspond to the α-Attention, the β-Attention, and the γ-Attention, respectively. Here, we can theoretically show that this attention structure enables the model to automatically discover important hyper-meta-paths by back-tracing the learned attention coefficients.

E.1 Hyper-meta-path Importance Calculation

Consider a target node $i \in \mathcal{V}$, its final embedding is output by the last layer N, which reflects the information that flows through a sequence of interconnected nodes and hyperedges across the N layers. Theoretically, the importance is the product of the involved attention weights along this

sequence. Thus, regarding the hyper-meta-path $\mathcal{P} = R_1 \diamond R_2 \diamond \cdots \diamond R_N$ involved in this sequence, its importance can be computed as follows:

$$I(\mathcal{P}) = \prod_{l=1}^{N} \alpha^{[l]} \cdot \beta^{[l]} \cdot \gamma^{[l]} \tag{10}$$

A higher value of $I(\mathcal{P})$ indicates that the information that propagates along the hyper-meta-path $\mathcal{P}$ contributes more significantly to the target nodes i's final embedding $v_i^{[N]}$.

E.2 Important Hyper-meta-path Discovering

By calculating the importance $I(\mathcal{P})$ for all relevant paths that lead to the target node i, we can identify the most important hyper-meta-path $\mathcal{P}^*$ for node i as follows:

$$\mathcal{P}^* = \arg\max_{\mathcal{P}} I(\mathcal{P}) \tag{11}$$

$\mathcal{P}^*$ represents the most significant hyper-meta-path for the target node i, as learned by the model.

E.3 Interpretability of Discoverd Hyper-meta-path

The automatically discovered hyper-meta-path $\mathcal{P}^*$ carries specific semantic meaning derived from the composition of the involved individual hyper-relations. This semantic interpretation can be used to explain the model's prediction for the target node i. For instance, if the task is node classification, the semantics of $\mathcal{P}^*$ can provide insight into why the model assigned a particular class label to the target node i.

E.4 Discussion and Outlook of Attention-based Hyper-meta-path Discovery

HHNN discovers hypermeta-paths based on the attention distribution, and this attention-based heuristic reveals correlations rather than causal relationships. In the future work, formalization of causal interpretability is an important direction, and we plan to explore techniques like counterfactual reasoning in our future work. Here, we also conduct a preliminary experiment of masking information along high-weight paths and observing the change in the model's prediction, and the results are shown in the Table 5. As we can observe, by masking information along high-weight paths, the model's performance drops sharply, which clearly indicates that the high-weight paths identified by the HHNN model are crucial.

Table 5: Comparison between HHNN performance with high-weight paths masked (right) and the original performance (left) under the 20% training ratio.

	Movielen	Olist
Micro	$56.80_{\pm 1.65} \rightarrow 28.63_{\pm 5.32}$	$72.54_{\pm 0.54} \rightarrow 13.24_{\pm 0.81}$
Macro	$26.24_{\pm 3.90} \rightarrow 11.83_{\pm 2.80}$	$72.89_{\pm 0.56} \rightarrow 6.38_{\pm 0.73}$
AUC	$73.21_{\pm 2.13} \rightarrow 54.04_{\pm 1.31}$	$93.20_{\pm 0.32} \rightarrow 50.13_{\pm 0.51}$

F Comparison between HAN and HHNN

We compare our HHNN to HAN, which is based on a two-level attention aggregation mechanism. The results are shown in Table 6. As we can see, our HHNN can achieve higher performance against HAN on both ACM and DBLP, which demonstrates the superiority of our HHNN.

G Experiments on More Datasets

Most existing heterogeneous hypergraph datasets are transformed from simple graphs with binary relations, which lack natural multi-way interactions and cannot fully showcase the advantages of

Table 6: Dataset statistics.

	ACM	DBLP
HAN	$85.09_{\pm 2.48}$	$90.02_{\pm 0.87}$
HHNN	$\mathbf{87.28}_{\pm 0.88}$	$\mathbf{91.29}_{\pm 0.92}$

hypergraph neural networks. In contrast, in our work, we have tried our best to construct two new real-world datasets from the movie domain and e-commerce domain, both containing multiple types of naturally occurring heterogeneous hyperedges based on genuine multi-ary relations. The two datasets are more challenging to fully showcase the power of our HHNN. Here, we also conduct a performance comparison on three widely-used hypergraph datasets, i.e., Cora, PuMmed, and DBLP, and the results are shown in Table 7. We can observe that since these three datasets do not contain complex and heterogeneous structural features, the three hypergraph methods achieve comparable performance, with our HHNN performing slightly better.

Table 7: The performance comparison between three hypergraph methods on three new hypergraph datasets, under the 20% training ratio.

	Cora	PubMed	DBLP
UniGNN	$76.93_{\pm 1.73}$	$84.05_{\pm 0.63}$	$90.49_{\pm 0.18}$
ED-HNN	$77.07_{\pm 1.22}$	$84.12_{\pm 0.68}$	$90.50_{\pm 0.21}$
HHNN	$\mathbf{77.42}_{\pm 1.67}$	$\mathbf{84.14}_{\pm 0.74}$	$\mathbf{90.50}_{\pm 0.27}$

Table 8: The performance comparison on Movielens, under the 20% training ratio.

	Micro	Macro	AUC
SheafHyperGNN [9]	$46.59_{\pm 1.24}$	$18.85_{\pm 1.74}$	$61.68_{\pm 1.48}$
UniG-Encoder [44]	$41.79_{\pm 1.17}$	$16.15_{\pm 0.96}$	$58.66_{\pm 1.32}$
HHNN	$\mathbf{56.80}_{\pm 1.65}$	$\mathbf{26.24}_{\pm 3.90}$	$\mathbf{73.21}_{\pm 2.13}$

Table 9: The performance comparison on Olist, under the 20% training ratio.

	Micro	Macro	AUC
SheafHyperGNN [9]	$51.71_{\pm 1.54}$	$51.56_{\pm 1.38}$	$85.60_{\pm 1.02}$
UniG-Encoder [44]	$24.40_{\pm 0.39}$	$23.68_{\pm 0.49}$	$64.96_{\pm 0.28}$
HHNN	$\mathbf{72.54}_{\pm 0.54}$	$\mathbf{72.89}_{\pm 0.56}$	$\mathbf{93.20}_{\pm 0.32}$

Table 10: The running time (seconds of 10 epochs) of HHNN on heterogeneous hypergraphs of different sizes using varying ratios of nodes.

	Movielens	Olist
20%	0.730	2.361
40%	0.763	2.449
60%	1.114	2.839
80%	1.205	2.883
100%	1.231	3.194

H Experiments with More Baselines

We compare our HHNN against two recent hypergraph models, i.e., SheafHyperGNN [9] and UniG-Encoder [44], as the new baselines, and their experimental results are shown in Table 8 and Table 9.

Table 11: The running time (seconds of 10 epochs) of HHNN on heterogeneous hypergraphs of different sizes using varying ratios of hyperedges.

	Movielens	Olist
20%	0.653	2.413
40%	0.805	2.528
60%	0.927	2.809
80%	1.111	2.978
100%	1.231	3.194

Table 12: The running time (seconds of 1 epoch) of HHNN in comparison with baselines UniGNN and EDHNN.

	HyperGCN	UniGNN	ED-HNN	HHNN
Movielens	81.57	0.13	0.18	0.15
Olist	90.22	0.17	0.63	0.19

We can see that our HHNN significantly outperforms the two hypergraph baseline methods on both datasets.

I Efficiency Study

To further verify the efficiency of HHNN, we have added the scalability experiment as well as the running time comparison experiment. The results are shown in Table 10, Table 11, and Table 12. As we can see, the running time of HHNN increases approximately linearly with the size of the graph. Besides, the running time of HHNN is on par with two baselines, UniGNN and EDHNN, and is much less than the running time of the baseline HyperGCN.

J More Related Work

In the past decade, the representation learning methods on hypergraphs have become a research surge. In the recent two years, several surveys [1, 15, 32, 23, 28] have been published to comprehensively review existing hypergraph representation learning methods from different perspectives.

Regarding the heterogeneity of nodes and hyperedges, some of the existing methods, e.g., [4], only consider node heterogeneity, and some methods generally consider the hyperedge heterogeneity based on operations like concatenation [30] and adaptive fusion [14]. RelBench [13] leverages graph neural networks to learn directly from relational databases, which can also be viewed as a heterogeneous hypergraph neural network. In contrast, HHNN focuses on hyper-meta-paths to capture more complex structural features.

There are also some other path-related methods. Pathsim [31] is the first work to define the concept of meta-path as a sequence of binary relations among two nodes, which has shown extraordinary effectiveness in capturing heterogeneous structural features and rich semantics contained in heterogeneous simple graphs. The subsequent study, metapath2vec [8], leverages a set of user-specified meta-paths to guide the random walk on heterogeneous simple graphs, transforming heterogeneous structural features into sequences. Then, it develops a word2vec [27]-like encoder to obtain the node representations. Considering that existing methods need users to specify useful meta-paths, which is not practical, ie-HGCN [40] is proposed to automatically discover and exploit useful meta-paths in heterogeneous simple graphs. HAN [35] is a well-known GNN method for heterogeneous simple graphs, which is based on a set of user-specified meta-paths. While our model and HAN both adopt hierarchical attention, they are fundamentally different in scope and purpose. Firstly, HAN operates on heterogeneous simple graphs, where all relations are binary between node pairs, whereas HHNN is designed for heterogeneous hypergraphs that naturally encode multi-ary relations of various types. Moreover, HAN's two-level attention relies on manually specified meta-paths, while HHNN introduces a three-level attention mechanism that automatically discovers hyper-meta-paths without

manual intervention. HMG-CR [39] proposes the concept of hyper meta-path, which is a composition of multiple meta-paths between two specified end nodes in a heterogeneous simple graph. HPHG [20] defines the concept of hyper-path, which is utilized to preserve the complex information about the indecomposability of hypergraphs.

Different from these existing methods above, in this work, we define a novel concept called hyper-meta-path, which describes a complex hyper-relation by compositing a sequence of hyper-relations.

K Limitations

While the proposed HHNN demonstrates strong performance in modeling heterogeneous hypergraphs and good ability to automatically discover semantically meaningful hyper-meta-paths, several limitations should be acknowledged: (1) Although the attention-based aggregation enhances interpretability, it reveals correlations rather than causal relationships; (2) Although the model can learn to ignore some meaningless combinations by the attention optimization, the current definition of hyper-meta-path may be two broad in real-world scenarios, particularly when the shared node type is generic or high-frequency; (3) HHNN is particularly designed for complex heterogeneous hypergraphs, and it is most pronounced on hypergraphs with natural multi-ary relations, and its advantages may not be fully realized when such structures are not prominent. In future work, we would like to further explore alternative definitions and discovery mechanisms for hyper-meta-paths, and the direction of utilizing causality is very promising.

L Broader Impacts

The proposed HHNN model contributes to the broader field of graph representation learning by enabling improved performance and understanding for heterogeneous hypergraphs. This advancement can benefit many applications, such as recommendation systems, bioinformatics, and social network analysis, by simultaneously improving task performance and enhancing model interpretability through the modeling of more complex and higher-order relational patterns.