Interpretable Hypergraph Neural Additive Networks

Anonymous authors
Paper under double-blind review

Abstract

Hypergraph neural networks have emerged as a powerful framework for learning from higher-order structured data, where relationships among entities extend beyond pairwise connections. However, most current hypergraph neural networks are black-boxes that rely on post-hoc explanation methods to provide model insights. Such post-hoc explanations can be unreliable in high-stakes scenarios and knowledge discovery tasks. We introduce an inherently interpretable hypergraph neural additive network (HGNAN), an extension of generalized additive models that facilitates interpretability in complex, higher-order relational learning settings. HGNAN provides clear visualizations of both global and local behaviors at the node and hyperedge levels while preserving the expressive power of hypergraphs. We evaluate HGNAN on node classification and hyperedge prediction across various datasets, achieving competitive performance compared to state-of-the-art methods. HGNAN also significantly outperforms existing approaches in recovering missing reactions in metabolic networks, while offering interpretable biological insights into metabolic processes.

1 Introduction

Hypergraphs are effective in capturing complex interactions among multiple entities and have been widely applied across various domains, such as metabolic, ecological, and social networks (Battiston et al., 2020; Chen et al., 2023; Grilli et al., 2017; Zhu et al., 2018). Unlike traditional graphs where each edge connects only two nodes, a hyperedge can link any number of nodes, offering greater expressive power and flexibility in modeling multidimensional real-world systems (Berge, 1984). Recent advances have led to significant success in tasks like node classification and hyperedge prediction (Gao et al., 2020; Schölkopf et al., 2007). In particular, hypergraph neural networks (HGNNs) have emerged as a state-of-the-art method for representation learning, effectively leveraging higher-order structural patterns (Bai et al., 2021; Feng et al., 2019). However, like many deep learning models, HGNNs operate as black boxes, providing limited insight of their decision-making processes. This lack of transparency raises concerns about trust, restricts their adoption in high-stakes applications, and impedes scientific knowledge discovery.

Post-hoc explainability techniques, such as HyperEX (Maleki et al., 2023) and SHypX (Su et al., 2024), have been proposed to open the black box. Inspired by graph-based counterparts such as GNNExplainer (Ying et al., 2019), PGExplainer (Luo et al., 2020), and XGNN (Yuan et al., 2020), both methods identify sub-hypergraphs that maximize mutual information with the model's output. These extracted patterns are designed to enhance the predicted probability for a specific class, offering model-level explanations. However, both approaches rely on an auxiliary explanatory model to interpret a trained HGNN. Such explanations can be unreliable and may even undermine trust in the model's decisions (Rudin, 2019). Recently, inherently interpretable graph neural networks have been introduced by using prototype reasoning (Dai & Wang, 2025) and generalized additive models (Bechler-Speicher et al., 2024). Nevertheless, to the best of our knowledge, interpretable HGNNs remain unexplored.

In this paper, we introduce an inherently interpretable neural network designed for hypergraph learning tasks, named hypergraph additive neural network (HGNAN) (see Figure 1). HGNAN

integrates the principles of neural additive models (NAMs) (Agarwal et al., 2021) with HGNN architectures. Specifically, it adds distance-based weights into NAMs to learn structural information, adopting the message-passing mechanism of HGNNs to model higher-order relationships. HGNAN can provide clear interpretations and visualizations of both global and local behaviors at the node and hyperedge levels. Our experiments show that HGNAN achieves performance comparable to various baselines in both node and hyperedge prediction tasks while providing interpretations of exact decision-making process of the underlying model. HGNAN also establishes a new state-of-the-art in identifying missing reactions in genome-scale metabolic networks, and its interpretability enables the discovery of novel biological insights that enhance our knowledge of metabolic processes.

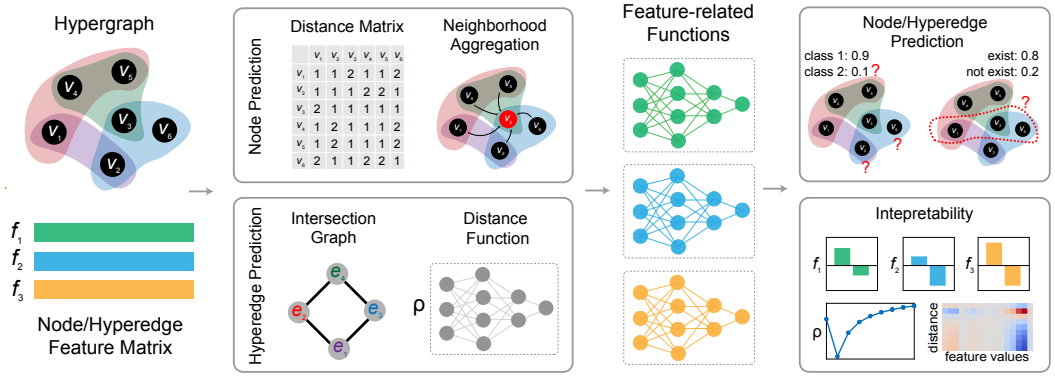


Figure 1: Overview of HGNAN. HGNAN takes a hypergraph and a feature matrix as input (leftmost column) and processes them through two independent components: one that captures distance-related information (mid-left column) and another that captures feature-related patterns (mid-right column). For node prediction, HGNAN computes a distance matrix and performs neighborhood aggregation. For hyperedge prediction, it builds an intersection graph and applies a distance function between nodes. As in NAMs, each feature-related function operates only on a single feature to ensure interpretability.

2 Related Works

Hypergraph neural networks. The concept of hypergraph neural networks (HGNNs) was first introduced by Feng et al. (2019), using a spectral method to generalize graph convolution to hypergraphs via the hypergraph Laplacian. Then various HGNN architectures have been developed to improve expressiveness, scalability, and adaptability. HAN (Chen et al., 2020) incorporates attention mechanisms into message passing to highlight node importance within hyperedges. HyperGCN (Agarwal et al., 2021) approximates hyperedges using pairwise connections via mediator nodes, enabling standard graph neural networks (GNNs) to capture higher-order relationships. UniGNN (Huang & Yang, 2021) mitigates over-smoothing in deep models by extending GCNII with residual connections and identity mappings. AllSetTransformer (Chien et al., 2021) uses Deep Sets and Set Transformers for permutation-invariant multiset message passing. More recently, HGNN+ (Gao et al., 2023) unifies multi-modal features and hyperedge groups for heterogeneous hypergraphs, while ED-HNN (Wang et al., 2023) combines star expansion with message passing to approximate equivariant diffusion. Despite these advances, all these HGNN architectures lack interpretability, limiting their ability to offer transparent insights.

GNN explainability. The two post-hoc HGNN explainability techniques, HyperEX (Maleki et al., 2023) and SHypX (Su et al., 2024), have been proposed, both rooted in established GNN explainability methods. GNNExplainer (Ying et al., 2019) identifies a compact subgraph and a subset of node features that are most influential for a specific prediction, providing instance-level explanations through optimization-based mask learning. Building on this, PGExplainer (Luo et al., 2020) introduces a parametric approach that learns a probabilistic edge mask generator, allowing for faster and more generalizable explanations across different

instances. In contrast, XGNN (Yuan et al., 2020) takes a generative view by learning to synthesize graphs that maximize the confidence of a target prediction class, offering a global and class-level interpretability framework. However, they are post-hoc methods which can be unreliable and, in some cases, may undermine user trust due to inconsistencies or lack of fidelity to the model's true reasoning process (Rudin, 2019).

Additive models. Additive models have long been valued in machine learning for their interpretability and flexibility (Rudin et al., 2022). The most classical form, the generalized additive model (GAM) (Hastie & Tibshirani, 1986), is a linear combination of univariate shape functions which can be easily understood via 2D plots. Neural additive models (NAMs) (Agarwal et al., 2021) were later introduced to learn a linear combination of neural networks, each trained on a single input feature. NAMs leverage the expressive power of deep neural networks to capture complex relationships between inputs and outputs, while ensuring interpretability through a structured architecture. Graph neural additive networks (GNANs) (Bechler-Speicher et al., 2024) extended these ideas to graph-structured data, merging the interpretability of NAMs with the representational power of GNNs. However, additive modeling on hypergraphs with HGNNs has yet to be explored.

3 Methods

A hypergraph generalizes the concept of traditional graphs by allowing each hyperedge to connect an arbitrary subset of nodes, rather than being limited to pairwise connections. Formally, a hypergraph $\mathcal{H} = \{\mathcal{V}, \mathcal{E}\}$ with n nodes and m hyperedges consists of a set of nodes $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ and a set of hyperedges $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$, where each hyperedge $e_j \in \mathcal{E}$ is a non-empty subset of $\mathcal{V}$, i.e., $e_j \subseteq \mathcal{V}$. The structural relationships in a hypergraph can be represented using an incidence matrix $\mathbf{H}$, where each entry $\mathbf{H}_{ij} = 1$ if node v_i is contained in hyperedge e_j, and it is equal to zero otherwise. Given a hypergraph $\mathcal{H}$, its intersection graph is defined upon hyperedge adjacency. Each node in the intersection graph represents a hyperedge in $\mathcal{E}$, and an edge is placed between two nodes if their corresponding hyperedges share at least one common node in $\mathcal{V}$. In other words, an edge between two nodes e_i and e_j exists if $e_i \cap e_j \neq \emptyset$.

Building on the formulation of hypergraphs, we present a novel interpretable hypergraph neural additive network (HGNAN), which consists of two parts: HGNAN-node for node-level prediction and HGNAN-edge for hyperedge-level prediction. Both take the incidence matrix $\mathbf{H}$ and node features as input to learn feature shape functions that capture the effects of individual features. The key distinction is that HGNAN-node leverages the hypergraph structure to perform neighborhood-level aggregation, whereas HGNAN-edge learns a separate distance function defined over higher-order intersection graphs.

3.1 Distance on Hypergraphs

Given a hypergraph $\mathcal{H} = \{\mathcal{V}, \mathcal{E}\}$, a pair of hyperedges e_i, $e_j \in \mathcal{E}$ is called s-adjacent if they share at least s nodes in common, i.e., $|e_i \cap e_j| \geq s$. This allows us to extend intersection graphs to s-intersection graphs by defining the s-hyperedge adjacency matrix $\mathbf{A}(s)$ as

$$\mathbf{A}(s)_{ij} = \mathbf{1}[|e_i \cap e_j| \geq s]$$

For any pair of hyperedges e_i, $e_j \in \mathcal{E}$, we define the s-distance between the two hyperedges as $d_s(e_i, e_j) = \left\{ \min \alpha \in \mathbb{Z}^+ \mid \mathbf{A}^\alpha(s)_{ij} > 0 \right\}$, where α can be interpreted as the shortest distance between e_i and e_j. If no such α exists, we set $d_s(e_i, e_j) = \infty$. In other words, $d_s(e_i, e_j)$ is the minimum number of hops required to connect e_i and e_j via a chain of s-adjacent hyperedges. Additionally, the s-distance between two nodes v_i, $v_j \in \mathcal{V}$ can be defined as

$$d_s(v_i, v_j) = 1 + \min_{v_i \in e_p,\ v_j \in e_q} d_s(e_p, e_q). \tag{1}$$

If no connected pair of hyperedges that contain v_i and v_j can be found, we set $d_s(v_i, v_j) = \infty$. If $d_s(v_i, v_j) \leq 2$, we consider that node v_i is in the neighborhood of node v_j, i.e. $v_j \in \mathcal{N}_i$. This implies that nodes on the same hyperedge and those one hop away are neighbors. We denote $s_{\max}$ as the highest order of adjacency considered and is introduced as a tunable hyperparameter, enabling the model to learn higher-order adjacency from the hypergraph.

3.2 Node Prediction Tasks

HGNAN-node generalizes NAMs to hypergraph-structured data by learning a set of feature-wise shape functions $\{f_k\}_{k=1}^{p}$, where p denotes the number of input features. Each shape function f_k transforms the k^{th} feature independently across a node's neighborhood and contributes to a refined representation via neighborhood-level aggregation. This design effectively suppresses noise from irrelevant neighbors, mitigates oversmoothing from repeated mixing, and enhances both memory and computational efficiency. Let $\mathbf{x}_i \in \mathbb{R}^p$ denote the original features and $\mathbf{h}_i^s \in \mathbb{R}^p$ the updated embedding for node i under s-adjacency. The k^{th} entry of the embedding $\mathbf{h}_i^s$ using HGNAN-node is computed as

$$[\mathbf{h}_i^s]_k = \sum_{v_j \in \mathcal{N}_i^s} \frac{a_{ij}}{\#d_s(v_i, v_j)} f_k\left([\mathbf{x}_j]_k\right) + \lambda ||\mathbf{a}||_1, \tag{2}$$

where $\mathcal{N}_i^s$ denotes the neighborhood of node i under s-adjacency (the definition for neighboorhood of node v_i is the same as described in Section 3.1), and $f_k(\cdot)$ is a feature-dependent shape function associated with the k^{th} feature. Each shape function is parameterized by a multi-layer perceptron (MLP). A neighbor weight a_{ij} is assigned to each neighbor j of the target node i and is calculated using a small neural network similar to graph attention network (Velickovic et al., 2018). To encourage neighborhood-level sparsity, we apply an L1-norm to neighbor weights. The term $\#d_s(v_i, v_j)$ measures the number of nodes sharing the same distance away from node i under s-adjacency, serving as a normalization factor. HGNAN-node follows the formulation of NAMs and GNANs, where $[\mathbf{h}_i^s]_k$ is a linear combination of the transformed features $f_k([\mathbf{x}_j]_k)$, representing the aggregated contribution of the k^{th} channel at node i.

For node-level prediction, we introduce an s-invariant weight vector $\mathbf{w}$ which learns how each entry contributes to the prediction under s-adjacency $[\mathbf{h}_i^s]$, i.e.,

$$[\mathbf{h}_i^s] = \sum_{k=1}^{p} w_k [\mathbf{h}_i^s]_k = \sum_{v_j \in \mathcal{N}_i^s} \frac{a_{ij}}{d_s(v_i, v_j)} \sum_{k=1}^{p} w_k f_k\left([\mathbf{x}_j]_k\right), \tag{3}$$

where w_k is the k^{th} element of $\mathbf{w}$, which acts as a learnable weight for feature k and is normalized using softmax so that $\sum_{k=1}^{K} w_k = 1$. This w_k could act as a relative feature importance for feature k for the whole task. The final embedding for node i is a sum weighted by learnable parameter β over $\{[\mathbf{h}_i^s]\}_{s=1}^{s_{\max}}$, which are embeddings learned under all high-order adjacency:

$$[\mathbf{h}_i] = \sum_{s=1}^{s_{\max}} \beta_s [\mathbf{h}_i^s], \tag{4}$$

Eq (4) allows the model to flexibly learn each neighbor's contribution to node i by considering both high-order adjacency and feature-level importance. This representation can then be passed through a sigmoid or softmax activation function to flexibly adjust for either binary or multiclass classification or regression. Furthermore, HGNAN-node can be naturally extended to hypergraph-level prediction by pooling the node-level aggregated scores into a single hypergraph-level representation $\mathbf{h}$, i.e.,

$$\mathbf{h} = \sum_{i=1}^{N} [\mathbf{h_i}] \tag{5}$$

3.3 Hyperedge Prediction Tasks

Hyperedge prediction methods usually first compute node embeddings for individual nodes and then combine those node embeddings through a pooling layer to get an embedding for the hyperedge. However, this pooling step can lead to several issues. For example, max-pooling may cause non-smoothness and gradient instability, while average pooling can lead to gradient dilution and reduced receptive-field gradients (Boureau et al., 2010). Additionally, it may compromise model interpretability by offering node-level explanations

for hyperedge predictions, rather than providing explanations directly from the hyperedge-level perspective. More importantly, for most node classification datasets, they are highly homophilic. 0-hop and 1-hop neighbors in these datasets usually covers most of the nodes. In this away, deploying neighborhood aggregation can cover most of the valuable information. In contrast, for hyperedge prediction tasks, they are highly heterophilic, which means that neighborhood aggregation cannot capture enough information. To address these issues, we propose HGNAN-edge, which leverages the concept of s-intersection graphs and uses overall aggregation for hyperedge prediction. See Appendix B for more details.

In HGNAN-edge, we first transform the hypergraph into its corresponding s-intersection graph, and then generate hyperedge embeddings directly from features of the nodes associated with each hyperedge. By performing pooling on raw node features, this new method allows us to design interpretable hyperedge embedding based on prior knowledge. As a result, the model can provide hyperedge-level interpretation, which preserves better interpretability compared to those that pool after node embedding. This new formulation turns hyperedge prediction into a node prediction task on the s-intersection graph, thereby HGNAN-edge can easily adapt GNAN to get hyperedge embeddings and make predictions.

Let $\mathbf{x}_l \in \mathbb{R}^p$ denote the original feature vector of node v_l. The initial embedding for hyperedge e_i is $\mathbf{y}_i = \text{Pool}(\{\mathbf{x}_l : v_l \in e_i\})$, where $\text{Pool}(\cdot)$ denotes a pooling function, such as average pooling. HGNAN-edge computes the k^{th} entry of the refined embedding $\mathbf{g}_i^s \in \mathbb{R}^p$ on the s-intersection graph as follows:

$$[\mathbf{g}_i^s]_k = \sum_{j=1}^{m} \frac{1}{d_s(e_i, e_j)} \rho_s \left(\frac{1}{1 + d_s(e_i, e_j)} \right) f_k \left([\mathbf{y}_j]_k \right), \tag{6}$$

where $f_k(\cdot)$ as defined in Eq (2) is a feature-dependent shape function corresponding to the k^{th} feature, and $\rho_s(\cdot)$ is a distance-based weighting function that captures the cumulative influence of hyperedges at varying distances from e_i on the s-intersection graph. Both $f_k(\cdot)$ and $\rho_s(\cdot)$ are parameterized by MLP. To avoid division by zero, we add 1 to each distance value in the denominator. Finally, we compute the refined hyperedge representation using the same aggregation strategy as HGNAN-node, i.e.,

$$[\mathbf{g}_i^s] = \sum_{k=1}^{p} w_k [\mathbf{g}_i]_k = \sum_{j=1}^{m} \frac{1}{d_s(e_i, e_j)} \rho_s \left(\frac{1}{1 + d_s(e_i, e_j)} \right) \sum_{k=1}^{p} w_k f_k \left([\mathbf{y}_j]_k \right), \ [\mathbf{g}_i] = \sum_{s=1}^{s_{\max}} \beta_s [\mathbf{g}_i^s], \tag{7}$$

where w_k and β_s are learnable weights. Similar to HGNAN-node, Eq (7) can also be interpreted through two independent parts: a distance-related part $\sum_{j=1}^{m} \frac{1}{d_s(e_i,e_j)} \rho_s(\frac{1}{1+d_s(e_i,e_j)})$ and a feature-related part $\sum_{k=1}^{p} w_k f_k([\mathbf{y}_j]_k)$. However, unlike HGNAN-node, which employs neighborhood-level aggregation in the distance-related part, HGNAN-edge performs graph-level aggregation using a distance-based weight function $\rho_s(\cdot)$.

Note that for most existing hyperedge prediction datasets, HGNAN-edge requires negative sampling, meaning that we need to generate hypothesized hyperedges (also called negative hyperedges) from existing hypergraph. However, when we calculate the distances, we only use the information of the existing hypergraph structure without taking the new generated negative hyperedge into consideration. Specifically, for a generated negative sample $\tilde{e}_i$, we first identify the positive hyperedges it connects to and then calculate the distance based on those connections. Let E_i and E_j denote the sets of positive hyperedges connected to the negative hyperedges $\tilde{e}_i$ and $\tilde{e}_j$, respectively. The distance between the two sets E_i and E_j is defined as $d_s(E_i, E_j) = \min_{e_i, e_j} d_s(e_i, e_j)$ such that $e_i \in E_i, e_j \in E_j$. Accordingly, the distance between two negative hyperedges $\tilde{e}_i$ and $\tilde{e}_j$ is defined as $d_s(\tilde{e}_i, \tilde{e}_j) = d_s(E_i, E_j) + 2$. If one hyperedge is positive and the other is negative, the distance generalizes naturally as $d_s(\tilde{e}_i, e_j) = d_s(E_i, e_j) + 1$.

4 Experiments

Our experiment aims to address the following two questions: (1) Can HGNAN achieve performance comparable with black-box counterparts in terms of node prediction (Section

4.1) and hyperedge prediction (Section 4.2)? (2) What do the interpretations from HGNAN-node and HGNAN-edge look like (Section 4.3)?

4.1 Node Classification

We compare HGNAN-node with six hypergraph learning methods on four node classification tasks. Specifically, we compare to the following baselines: HGNN (Feng et al., 2019), HyperGCN (Yadati et al., 2019), AllDeepSets (Chien et al., 2021), AllSetTransformer (Chien et al., 2021), UniGCNII (Huang & Yang, 2021), and ED-HNN (Wang et al., 2023). These baselines represent diverse architectural designs for modeling high-order relationships and have demonstrated strong performance on node classification tasks. We also compared with MLP, which does not utilize any hypergraph structure to validate that the model learns and benefits from structural information. We use three widely used homophilic hypergraph-level node classification datasets: Zoo (Forsyth, 1990), Mushroom (mus, 1981), and NTU2012 (Chen et al., 2003). We also tested on Pokec and Actor, two heterophilic datasets introduced by (Li et al., 2025). Details about these datasets are available in Appendix A. Each dataset is randomly split into training, validation, and test sets with a 2:1:1 ratio, and this process is repeated 10 times using different random seeds. To ensure a fair comparison, all models are trained and evaluated using the same data splits and random seeds. We report the mean and standard deviation of the test accuracy across these 10 runs.

Table 1 shows the test accuracy of HGNAN-node and baselines across six different datasets. HGNAN-node achieves accuracy comparable to that of baseline methods. Notably, it outperforms all baselines on two heterophilic node classifications datasets. While some baselines slightly outperform HGNAN-node on homophilic datasets, the performance gap remains relatively small. Crucially, unlike these black-box models, HGNAN offers the added advantage of providing glass-box view of the decision making process, making it a compelling choice in scenarios where both accuracy and transparency are essential.

Table 1: Comparison of test accuracy (mean ± standard deviation) between HGNAN-node and baselines across node classification datasets. Bold indicates the highest test accuracy. HGNAN-node performs comparably to the best-performing models on all these datasets.

Method	Zoo	Mushroom	NTU2012	Pokec	Actor	Avg. Rank
MLP	0.887 ± 0.052	0.965 ± 0.006	0.853 ± 0.012	0.580 ± 0.019	0.827 ± 0.004	6
AllDeepSets	0.942 ± 0.042	**0.999 ± 0.001**	0.876 ± 0.014	0.567 ± 0.008	0.838 ± 0.003	4.8
AllSetTransformer	**0.973 ± 0.032**	**0.999 ± 0.001**	0.890 ± 0.011	0.572 ± 0.010	0.836 ± 0.002	3.4
ED-HNN	0.950 ± 0.035	0.998 ± 0.002	**0.895± 0.013**	0.618 ± 0.020	0.856 ± 0.006	3.2
HGNN	0.957 ± 0.022	0.998 ± 0.001	0.872 ± 0.014	0.553 ± 0.014	0.744 ± 0.004	5.6
HyperGCN	0.423 ± 0.000	0.482 ± 0.000	0.796 ± 0.033	0.538 ± 0.014	0.630 ± 0.000	8
UniGCNII	0.950 ± 0.048	**0.999 ± 0.001**	0.893 ± 0.016	0.570 ± 0.018	0.828 ± 0.003	4.2
HGNAN-node (ours)	0.953 ± 0.030	**0.999 ± 0.001**	0.890 ± 0.011	**0.634 ± 0.012**	**0.857 ± 0.004**	**3**

4.2 Hyperedge Prediction

Genome-scale metabolic models (GEMs) are essential tools for predicting cellular metabolism and physiological states in organisms (Fang et al., 2020). However, due to incomplete knowledge of metabolic processes, even well-curated GEMs often contain knowledge gaps, such as missing reactions. Predicting these missing reactions can be naturally framed as a hyperedge prediction task, where nodes represent metabolites and hyperedges correspond to reactions (Chen et al., 2023; Chen & Liu, 2024). We compare HGNAN with the state-of-the-art hyperedge prediction methods including HyperSAGNN (Zhang et al., 2020), NHP (Yadati et al., 2020) and CHESHIRE (Chen et al., 2023), which have demonstrated strong empirical performance in recovering missing reactions in metabolic networks. We evaluate on four GEMs from the BiGG database (King et al., 2016): iAF1260b, iJR904, iJR904 and iYO844. Appendix B gives a detailed introduction to these datasets.

Since BiGG models do not have inherent features for metabolites, we design a feature generation process by leveraging molecular fingerprints MACCS Keys(RDK, 2025) to represent

the metabolites (see Appendix B). To perform hyperedge prediction, negative sampling is required. For a given reaction in a GEM, a negative reaction is generated by replacing half of its metabolites with random metabolites drawn from a metabolite pool. Other experiment setups are identical to node prediction. We report the mean and standard deviation of the test accuracy across these 10 runs. Additional metrics and further details on the tuning process and parameter search space are available in Appendix C.

Table 2 summarizes the performance of various hyperedge prediction models on four GEM datasets. HGNAN consistently outperforms all state-of-the-art methods across all datasets, achieving the highest mean accuracy with notable margins. Notably, it outperforms the second-best method by over 10% on iAF1260b, iJR904, and iSB619. These results underscore the effectiveness of HGNAN in identifying missing reactions within complex metabolic networks. In addition to its strong predictive performance, HGNAN offers interpretability by linking metabolite chemical structures to prediction outcomes, providing valuable insights into the underlying biochemical mechanisms. An additional experiment in Appendix D on a synthetic hyperedge prediction dataset further validates our model's ability to recover the structural information.

Table 2: Comparison of hyperedge prediction accuracy (mean ± standard deviation) across four BiGG GEM datasets: iAF1260b, iJR904, iSB619, and iYO844. Bold highlights the best result for each dataset. HGNAN-edge outperforms all baseline models across all datasets.

Method	**iAF1260b**	**iJR904**	**iSB619**	**iYO844**
CHESHIRE	0.834 ± 0.050	0.732 ± 0.068	0.730 ± 0.038	0.893 ± 0.047
NHP	0.732 ± 0.076	0.690 ± 0.090	0.687 ± 0.055	0.747 ± 0.043
HyperSAGNN	0.730 ± 0.075	0.753 ± 0.056	0.729 ± 0.162	0.708 ± 0.045
HGNAN-edge (ours)	**0.935 ± 0.069**	**0.958 ± 0.026**	**0.977 ± 0.008**	**0.952 ± 0.181**

4.3 Interpretability

HGNAN provides a glass-box view of its decision-making process. According to Eqs (4) and (7), our model can be fully explained by two parts: the feature-related part and the distance-related part. The feature-related part, or Feature Contribution Score in the context, is $w_k f_k([x_j]_k)$, which reflects how the input features influence the output embedding. The distance-related part differs between HGNAN-node and HGNAN-edge. For HGNAN-node, it is the neighborhood weight a_{ij}, while for HGNAN-edge, there are distance functions $\{\rho_s(\cdot)\}_{s=1}^{s_{\max}}$, which adjust the influence of features with learnt high-order structural information. We can also learn relative feature importance for the whole task from w_k (see Appendix E).

Node-level interpretation. HGNAN-node can offer node-level interpretation for node prediction tasks. Since we deploy neighborhood-level aggregation in HGNAN-node, the feature contribution scores $\{w_k f_k\}_{k=1}^{p}$, provides a global explanation of how each input feature contributes to the predictions. We use the Zoo dataset as an example. It contains 101 animals described by 15 binary features (e.g., whether an animal has feathers) and 1 continuous feature. Animals are grouped into seven classes: "Mammal", "Bird", "Reptile", "Fish", "Amphibian", "Bug", and "Invertebrate". Hyperedges connect animals that share common attributes. The goal is to predict the class to which each animal belongs.

To interpret the contribution of each binary feature to the prediction of the "Mammal" class, we plot the learned values $w_k f_k(1)$ and $w_k f_k(0)$ for each feature k, as shown in Figure 2. The sign of $w_k f_k(x_k), x_k \in \{0, 1\}$ indicates whether the presence or absence of a feature increases or decreases the likelihood of an animal being classified as a mammal. For example, the presence of the "milk" ($f_{\text{milk}}(1)$) strongly supports mammal classification as it's value is positive. However, the presence of "feathers" ($f_{\text{feathers}}(1)$) lowers the predicted probability of being a mammal. Also, the magnitude of these values reflects absolute feature importance for prediction of specific class: larger absolute values correspond to greater influence on the prediction. As shown in Figure 2, the presence of "milk" $f_{\text{milk}}(1)$ has the highest absolute

contribution, making it the most important feature for identifying mammals. This aligns with biological knowledge of mammalian traits.

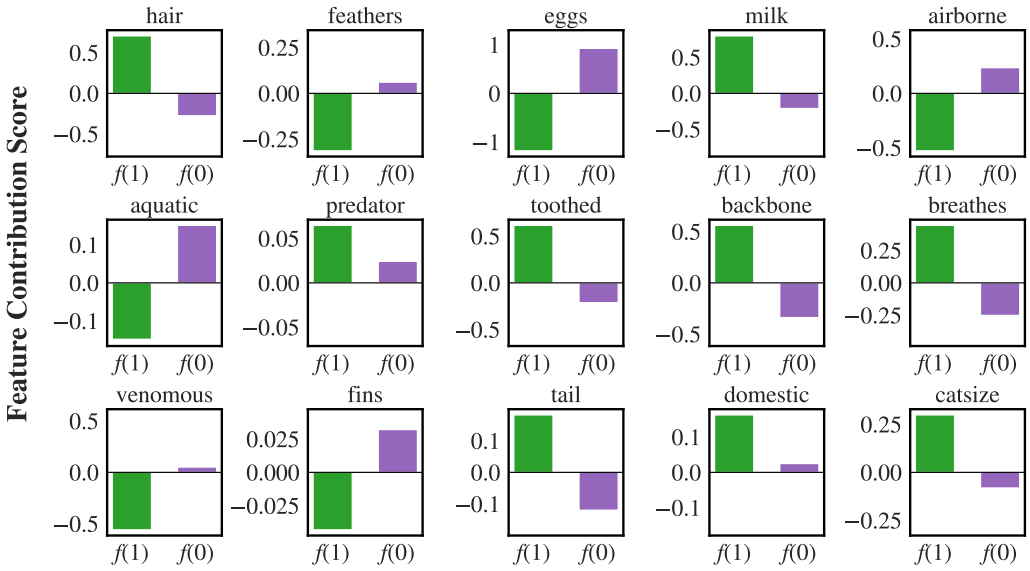


Figure 2: Feature contribution scores for predicting the "Mammal" class in the Zoo dataset. Each barplot shows the effect of a binary feature on the final prediction, where $f(1)$ represents the contribution when the feature is present and $f(0)$ when it is absent.

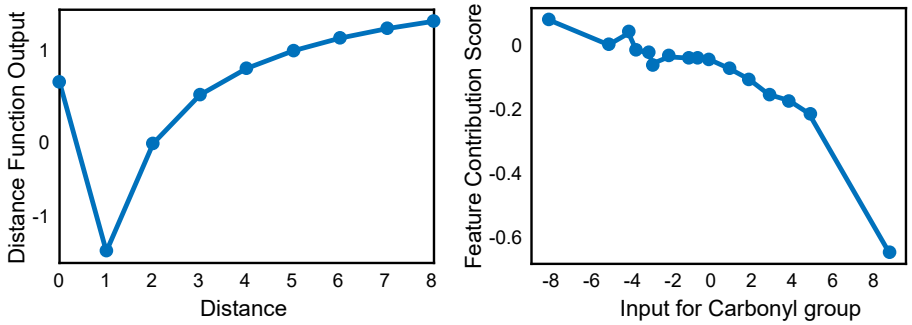


Figure 3: Visualization of the distance function (left) and Carbonyl group contribution score (right) for the iAF1260b dataset. Linear interpolation is used to connect the points.

Hyperedge-level interpretation. HGNAN-edge can provide hyperedge-level interpretation for hyperedge prediction. In this setting, the combination of distance functions $\{\rho_s(\cdot)\}_{s=1}^{s_{\max}}$ and the contribution of each feature, represented by $w_k f_k$ could fully explain the model's prediction. We use the iAF1260b dataset from the BiGG database for illustration, where each node feature corresponds to a certain function group. As mentioned in Section 3.3, we generate an embedding for each hyperedge by using difference pooling of node features on the hyperedge. Therefore, the hyperedge embedding represents the difference in the number of functional groups before and after the reaction.

Figure 3 shows the distance function (left) and contribution of the Carbonyl group (C=O) to prediction (right). The distance function $\rho_1(\cdot)$ indicates structural information from the hypergraph. For this dataset, the optimal $s_{\max}$ is 1. Therefore we just provide one distance function ($\rho_1(\cdot)$). The left subplot in Figure 3 shows that the model exhibits a V-shaped pattern. Features associated with the node on the same hyperedge (distance $= 0$) positively influence prediction, while features from immediate one-hop neighbors (distance $=$ 1) have a strong negative effect. As the distance increases further, the negative influence

diminishes and eventually becomes increasingly positive. This curve is a reasonable global kernel for representation learning and could be justified from biological perspective. At distance 0 (same reaction) stoichiometric coupling in steady state makes participants co-vary, justifying a large positive weight. Distance 1 often captures branch-point competition for a limited precursor; increasing flux down one branch reduces flow down the other, motivating a negative weight. At distance 2 this competitive effect is indirect and attenuated. Beyond two hops, local trade-offs diminish and pairs are more constrained by global objectives (e.g., biomass composition), so average associations become weakly positive, warranting positive weights.

The term $w_k f_k$ quantifies how feature k influences the model's output. The right subplot shows how the number of Carbonyl group (C=O) affect the prediction. The feature has a minor impact when the input of function group "C=O" is below 0, but its influence increases when k is above 0. It means that generating more Carbonyl group after the reaction would have a huge impact on prediction. To compute the overall contribution to prediction, we have to combine the feature contribution ($w_k \cdot f_k$) with the structural weight from the distance function ($\rho_1(\cdot)$). We visualize the combined contributions using heatmaps, as shown in Figure 4. Each cell in the heatmap represents the contribution of a particular feature-distance combination. For example, when a reaction results in the loss of 4 Carbonyl groups, the corresponding contribution is likely to be positive, as most values in the column of -4 are positive. These explanations allow domain experts to apply their knowledge to debug the model. For example, one can manually change the output of the distance function (e.g. $\rho_1(1)$) if it is not aligned with their domain knowledge and immediately get the prediction without retraining the whole model. It can also reveal some biologically meaningful patterns learned from the data, which may be valuable for downstream tasks.

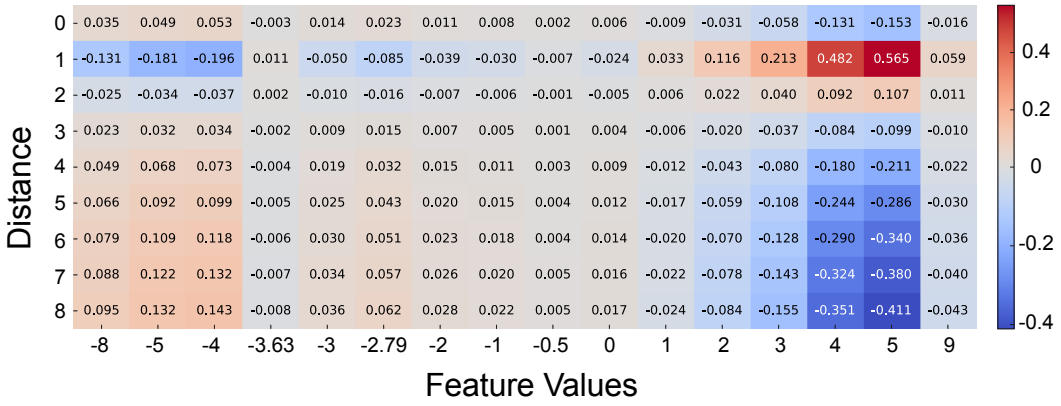


Figure 4: Heatmap for Carbonyl group: the horizontal axis stands for different feature value inputs $[\mathbf{y}_j]_{\text{C=O}}$ and the vertical axis stands for different distance inputs $d_1(e_i, e_j)$.

5 Conclusion

HGNAN provides an inherently interpretable model where the decision-making mechanism can be visualized by 2D plots. In the meantime, HGNAN can achieve competitive or superior performance compared to state-of-the-art hypergraph learning methods across various node and hyperedge prediction tasks. This dual capability of delivering strong predictive performance while offering transparent decision-making makes HGNAN a valuable tool for scientific discovery in hypergraph-based applications, especially in fields such as biotechnology, bioinformatics, and social network analysis. One limitation, however, is that HGNAN constructs a separate neural network for each feature, which can lead to high computational costs on datasets with very large dimensionality. Fortunately, such extreme high-dimensional datasets are relatively uncommon in many real-world applications. Looking forward, future work will explore extensions to dynamic hypergraphs, further broadening its applicability and scalability.

Reproducibility and Ethic Statement

An anonymous code is available at `https://anonymous.4open.science/r/HGNAN-6029`. We strictly adhere to the ICLR Code of Ethics (`https://iclr.cc/public/CodeOfEthics`).

References

Mushroom. UCI Machine Learning Repository, 1981. DOI: https://doi.org/10.24432/C5959T.

RDKit: Open-source cheminformatics. Zenodo DOI:10.5281/zenodo.591637, 2025. `https://www.rdkit.org`.

Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. Neural additive models: Interpretable machine learning with neural nets. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), Advances in Neural Information Processing Systems, volume 34, pp. 4699–4711. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/251bd0442dfcc53b5a761e050f8022b8-Paper.pdf`.

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

Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. Networks beyond pairwise interactions: Structure and dynamics. Physics Reports, 874:1–92, 2020. ISSN 0370-1573. doi: https://doi.org/10.1016/j.physrep.2020.05.004. URL `https://www.sciencedirect.com/science/article/pii/S0370157320302489`. Networks beyond pairwise interactions: Structure and dynamics.

Maya Bechler-Speicher, Amir Globerson, and Ran Gilad-Bachrach. The intelligible and effective graph neural additive networks, 2024. URL `https://arxiv.org/abs/2406.01317`.

Claude Berge. Hypergraphs: combinatorics of finite sets, volume 45. Elsevier, 1984.

Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, pp. 111–118, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Can Chen and Yang-Yu Liu. A survey on hyperlink prediction. IEEE Transactions on Neural Networks and Learning Systems, 35(11):15034–15050, 2024. doi: 10.1109/TNNLS.2023.3286280.

Can Chen, Chen Liao, and Yang-Yu Liu. Teasing out missing reactions in genome-scale metabolic networks through hypergraph learning. Nature Communications, 14(1):2375, 2023. doi: 10.1038/s41467-023-38110-7.

Chaofan Chen, Zelei Cheng, Zuotian Li, and Manyi Wang. Hypergraph attention networks. In 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1560–1565, 2020. doi: 10.1109/TrustCom50675.2020.00215.

Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. Computer Graphics Forum, 22(3):223–232, 2003. doi: 10.1111/1467-8659.00669.

Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. ArXiv, abs/2106.13264, 2021. URL `https://api.semanticscholar.org/CorpusID:235652158`.

Enyan Dai and Suhang Wang. Towards prototype-based self-explainable graph neural network. ACM Trans. Knowl. Discov. Data, 19(2), February 2025. ISSN 1556-4681. doi: 10.1145/3689647. URL https://doi.org/10.1145/3689647.

Xin Fang, Colton J. Lloyd, and Bernhard O. Palsson. Reconstructing organisms in silico: genome-scale models and their emerging applications. Nature Reviews Microbiology, 18 (12):731–743, 2020. doi: 10.1038/s41579-020-00440-4.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019. ISBN 978-1-57735-809-1. doi: 10.1609/aaai.v33i01.33013558. URL https://doi.org/10.1609/aaai.v33i01.33013558.

Richard Forsyth. Zoo. UCI Machine Learning Repository, 1990. DOI: https://doi.org/10.24432/C5R59V.

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.

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, 2023. doi: 10.1109/TPAMI.2022.3182052.

Jacopo Grilli, György Barabás, Matthew J Michalska-Smith, and Stefano Allesina. Higher-order interactions stabilize dynamics in competitive network models. Nature, 548(7666): 210–213, 2017.

Trevor Hastie and Robert Tibshirani. Generalized additive models. Statistical Science, 1(3): 297–318, 1986.

Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. In Zhi-Hua Zhou (ed.), Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, pp. 2563–2569. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/353. URL https://doi.org/10.24963/ijcai.2021/353. Main Track.

Zachary A. King, Justin Lu, Andreas Dräger, Philip Miller, Stephen Federowicz, Joshua A. Lerman, Ali Ebrahim, Bernhard O. Palsson, and Nathan E. Lewis. Bigg models: A platform for integrating, standardizing and sharing genome-scale models. Nucleic Acids Research, 44(D1):D515–D522, 2016. doi: 10.1093/nar/gkv1049. URL https://bigg.ucsd.edu.

Ming Li, Yongchun Gu, Yi Wang, Yujie Fang, Lu Bai, Xiaosheng Zhuang, and Pietro Liò. When hypergraph meets heterophily: New benchmark datasets and baseline. Proceedings of the AAAI Conference on Artificial Intelligence, 39(17):18377–18384, Apr. 2025. doi: 10.1609/aaai.v39i17.34022. URL https://ojs.aaai.org/index.php/AAAI/article/view/34022.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. In Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Sepideh Maleki, Ehsan Hajiramezalani, Gabriele Scalia, Tommaso Biancalani, and Kangway V. Chuang. Learning to explain hypergraph neural networks. In Proceedings of the 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML) at the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA, 2023. PMLR.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence, 1:206–215, 05 2019. doi: 10.1038/s42256-019-0048-x.

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. Statistic Surveys, 16:1–85, 2022.

Bernhard Schölkopf, John Platt, and Thomas Hofmann. Learning with Hypergraphs: Clustering, Classification, and Embedding, pp. 1601–1608. 2007.

Shiye Su, Iulia Duta, Lucie Charlotte Magister, and Pietro Liò. Explaining hypergraph neural networks: From local explanations to global concepts. ArXiv, abs/2410.07764, 2024. URL https://api.semanticscholar.org/CorpusID:273233777.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Peihao Wang, Shenghao Yang, Yunyu Liu, Zhangyang Wang, and Pan Li. Equivariant hypergraph diffusion neural operators. In International Conference on Learning Representations (ICLR), 2023.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/1efa39bcaec6f3900149160693694536-Paper.pdf.

Naganand Yadati, Vikram Nitin, Madhav Nimishakavi, Prateek Yadav, Anand Louis, and Partha Talukdar. Nhp: Neural hypergraph link prediction. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, CIKM '20, pp. 1705–1714, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368599. doi: 10.1145/3340531.3411870. URL https://doi.org/10.1145/3340531.3411870.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/d80b7040b773199015de6d3b4293c8ff-Paper.pdf.

Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20, pp. 430–438, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403085. URL https://doi.org/10.1145/3394486.3403085.

Ruochi Zhang, Yuesong Zou, and Jian Ma. Hyper-SAGNN: a self-attention based graph neural network for hypergraphs. In International Conference on Learning Representations (ICLR), 2020.

Jianming Zhu, Junlei Zhu, Smita Ghosh, Weili Wu, and Jing Yuan. Social influence maximization in hypergraph in social networks. IEEE Transactions on Network Science and Engineering, 6(4):801–811, 2018.

A Additional Details for Node Classification Datasets

We provide detailed information about the node classification datasets in this appendix.

Each node in the Mushroom and Zoo datasets represents a mushroom or an animal, respectively mus (1981); Forsyth (1990). Hyperedges connect samples that share the same feature values, where the features describe the characteristics of each sample and are usually binary or categorical. The target variable for the Mushroom dataset is whether a mushroom is edible or not, while in the Zoo dataset, we predict the type of animal. The NTU2012 dataset Chen et al. (2003) is a 3D shape dataset consisting of 2012 objects across 67 distinct categories (e.g., cars, chairs, chessboards, clocks). Each node represents an object and is described by multi-view visual descriptors. Twitch-Gamers dataset requires binary classification of explicit-content presence per channel with Twitch accounts as nodes and accounts co-created within the same time window as hyperedges. Features include view counts, creation/update timestamps, language, lifetime activity duration, and an inactive flag. Pokec is a binary classification dataset of user gender. Nodes are users while each hyperedge is a user's complete friend set. Features span profile attributes such as age, hobbies/interests, education level, region, and registration time. Actor is a multi-class classification dataset of production role (actor/director/writer). Nodes are film-industry people while each hyperedge contains all collaborators on a single film. Features there are keyword-based attributes extracted from Wikipedia. Table 3 provides a more detailed summary about each dataset.

Table 3: Summary of node classification datasets. $|E|$ stands for the number of nodes on a hyperedge E and d_v stands for the number of hyperedges that contains a node v.

	NTU2012	**Mushroom**	**Zoo**	**Pokec**	**Actor**
$\lvert V\rvert$	2012	8124	101	3200	15761
$\lvert E\rvert$	2012	298	43	2406	10164
# feature	100	22	16	65	50
# class	67	2	7	2	3
$\max \lvert E\rvert$	5	1808	93	7	28
$\min \lvert E\rvert$	5	2	2	2	1
avg $\lvert E\rvert$	4.0	29.5	64.5	2.3	5.3
$\max d_v$	5	1808	61	7	205
$\min d_v$	1	1	1	1	1
avg d_v	2.2	1.8	27.2	1.7	3.4

B Additional Details for Hyperedge Prediction Datasets

The BiGG (Biochemical Genetic and Genomic) dataset King et al. (2016) is a repository of genome-scale metabolic models (GEMs) for various organisms, including bacteria, yeast, and human cells. It provides high-quality, standardized models in the *Systems Biology Markup Language* (SBML) format, facilitating metabolic network reconstruction and simulation. The BiGG dataset consists of multiple components. First, it includes genome-scale metabolic models that represent the biochemical processes of various organisms. It is stored as a stoichiometric matrix. These models define metabolic reactions, describing the biochemical transformations occurring within cells, and metabolites, which are the chemical compounds involved in these reactions. Additionally, the dataset provides gene-protein-reaction (GPR) associations, linking genes to their corresponding enzymes and metabolic functions.

Feature generation. BiGG models do not provide features for each metabolic. Common approaches for feature generation include using node2vec embeddings or extracting molecular fingerprints from SMILES. However, this method generate features that are not interpretable. In our paper, we generate feature for each metabolite using MACCS Keys. MACCS Keys are a type of molecular fingerprint that converts a molecule's structure into a 167-dimension binary vector. Each dimension indicates the presence or absence of a specific predefined substructure or functional group(i.e. F or carbon ring). Because the substructures are predefined by biologists, MACCS Keys offer a fast and interpretable way to vectorize molecules. However, since MACCS Keys are based on molecule fingerprints, which do not necessarily have one-to-one correspondence with each molecule, some molecules do not have MACCS-key-based features. For these molecules, we pad their features using 0. We also report this missing rate in Table 4.

Access. The BiGG dataset is publicly available at `http://bigg.ucsd.edu/`. MACCS Keys are available at `https://github.com/jAniceto/ml-knowledge-base/blob/main/02-data-preparation/feature-engineering/maccs.md`. Table 4 provides details about the four models from the BiGG dataset used in our paper.

Table 4: Summary of four datasets from the BiGG dataset.

	iAF1260b	**iJR904**	**iSB619**	**iYO844**
$\|V\|$	2388	1075	743	1250
$\|E\|$	1668	761	655	990
Missing Rate	0.764	0.912	0.829	0.840
$\max \|E\|$	67	56	61	63
$\min \|E\|$	1	1	1	1
avg $\|E\|$	3.88	4.18	5.14	4.19
$\max d_v$	912	259	362	616
$\min d_v$	1	1	1	1
avg d_v	5.55	4.77	5.83	5.29

To quantify homophily at different distances ddd on a hypergraph, we report the same-label ratio:

$$SLR(d) = \frac{1}{|V_d|} \frac{\sum_{j=1}^{N} \mathbb{1}(y_j = y_i)\mathbb{1}(\text{dist}(i,j) = d)}{\sum_{j=1}^{N} \mathbb{1}(\text{dist}(i,j) = d)}$$

where $V_d = \{i : \exists j \neq i, \text{dist}(i,j) = d\}\}$ is the set of nodes that have at least one neighbor exactly d hops away. Since $SLR(d)$ can be misleading when the number of nodes at that distance is small, which is common when distance is large, we also provide coverage ratio across distances for reference.

For homophilic node classification datasets (e.g., NTU2012, Mushroom): $SLR(d)$ is high at small d and drops sharply with distance, indicating that near-hop neighborhoods are label-consistent while far-hop neighborhoods are not. In contrast, for heterophilic datasets (e.g., Pokec, Actor): there is no early decay. Actor shows a clear increase with distance; Pokec remains roughly flat around the mid-range hops and only declines later, consistent

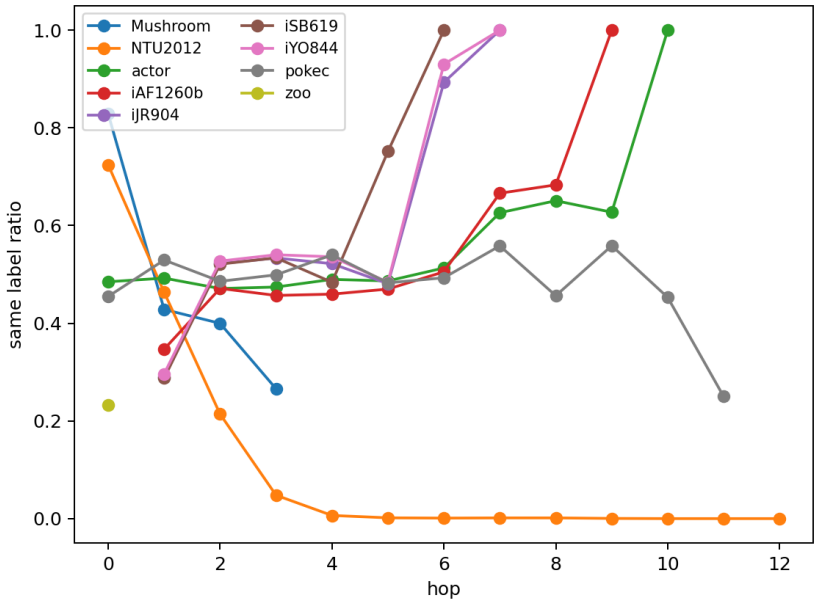


Figure 5: Same-label ratio by distance across all datasets

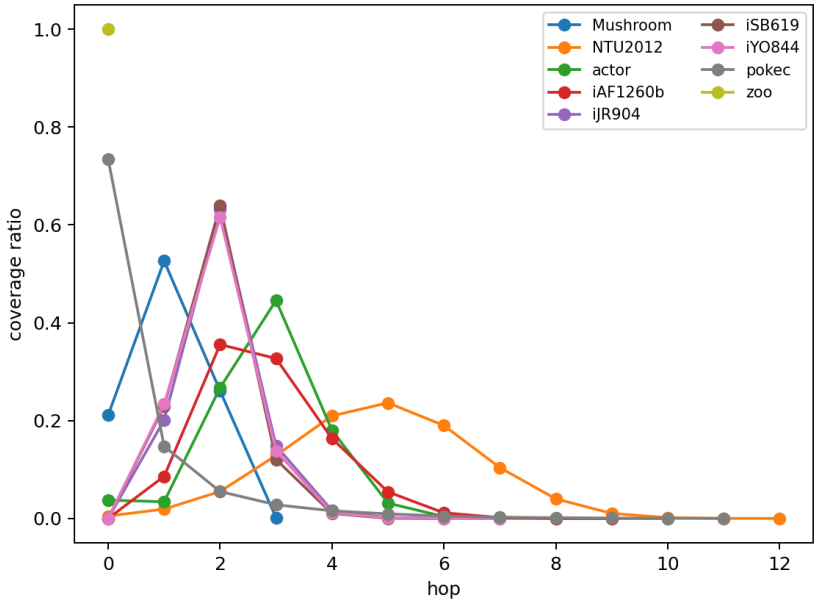


Figure 6: Coverage ratio by distance across all datasets

with weaker local homophily. For BiGG metabolic datasets (iAF1260b, iJR904, iYO844, iSB619), $SLR(d)$ increases with distance, often approaching 1 at larger hops, reflecting strong assortativity among far-hop nodes. Note that while far-hop $SLR(d)$ can be very high, the coverage at those hops is small, so the practical mass of labeled pairs still concentrates in the first few hops.

C Details for Model Training

In this appendix, we provide a detailed description of the experimental setup.

C.1 Node classification tasks

We perform grid search for hyperparameter tuning for each model to ensure optimal performance. We consider the following searching space for all methods: learning rate= $\{0.01, 0.001\}$, weight decay= $\{0, 0.0005\}$ and hidden channels= $\{64, 128, 256\}$. For methods using multi-head attentions, we tune the number of heads = $\{1, 4, 8\}$. For ED-HNN, we tune the number of layers =$\{0, 1\}$ and restart rate =$\{0, 0.5\}$. For our method, we also tune the number of layers =$\{3, 5\}$, dropout rate = $\{0, 0.5\}$ and $s_{\max} = \{1, 2, 3\}$. We select the hyperparameter configuration that achieves the best aggregated validation accuracy across all four datasets to guarantee robustness. The selected hyperparameters are reported in Table 5, and the corresponding evaluation result is in Table 1.

Table 5: Fixed best hyperparameters for node classification tasks

Method	lr	wd	dropout	n_layer	hidden_channel	Classifier_hidden	heads	$s_{\max}$	restart_alpha
HGNAN	0.001	0	0.0	3	128	-	-	1	-
AllDeepSets	0.001	0	-	-	256	64	-	-	-
AllSetTransformer	0.001	0	-	-	256	64	8	-	-
ED-HNN	0.001	0	-	0,0,1	512	512	-	-	0.5
HGNN	0.01	0.0005	-	-	256	128	-	-	-
HyperGCN	0.01	0.0005	-	-	64	128	-	-	-
UniGCNII	0.001	0	-	-	64	256	8	-	-

Following Chien et al. (2021) and Wang et al. (2023), we also select the best-performing hyperparameter configuration for each dataset individually. The optimal hyperparameters for each method–dataset combination and their corresponding performances on test set are reported in Table 6. HGNAN-node achieves accuracy comparable to that of baseline methods on most datasets. All experiments are run on NVIDIA V100 GPU.

Table 6: Best hyperparameters for each dataset for node classification tasks. The last column reports the test accuracy (mean ± standard deviation).

Dataset	Method	lr	wd	dropout	n_layer	hidden_channel	Classifier_hidden	heads	restart_alpha	s_{max}	Test Acc
NTU2012	HGNAN	0.001	0	0.5	3	256	-	-	-	3	0.896 ± 0.010
NTU2012	AllDeepSets	0.001	0	-	-	256	64	-	-	-	0.878 ± 0.014
NTU2012	AllSetTransformer	0.001	0	-	-	256	64	4	-	-	0.887 ± 0.010
NTU2012	HGNN	0.01	0.0005	-	-	256	256	-	-	-	0.873 ± 0.014
NTU2012	HyperGCN	0.01	0.0005	-	-	64	128	-	-	-	0.796 ± 0.033
NTU2012	UniGCNII	0.001	0.0005	-	-	128	64	4	-	-	0.898 ± 0.015
NTU2012	ED-HNN	0.001	0	-	0,0,1	512	256	-	0.5		**0.899 ± 0.011**
Mushroom	HGNAN	0.001	0	0	3	128	-	-	-	1	**0.999 ± 0.001**
Mushroom	AllDeepSets	0.01	0.0005	-	-	64	128	-	-	-	**0.999 ± 0.001**
Mushroom	AllSetTransformer	0.01	0	-	-	64	64	4	-	-	**0.999 ± 0.001**
Mushroom	HGNN	0.01	0.0005	-	-	256	256	-	-	-	0.998 ± 0.001
Mushroom	HyperGCN	0.001	0	-	-	64	64	-	-	-	0.482 ± 0.000
Mushroom	UniGCNII	0.001	0	-	-	64	128	1	-	-	**0.999 ± 0.001**
Mushroom	ED-HNN	0.001	0	-	0,1,1	512	128	-	0	-	0.998 ± 0.002
zoo	HGNAN	0.001	0	0	3	256	-	-	-	1	0.954 ± 0.033
zoo	AllDeepSets	0.001	0	-	-	256	64	-	-	-	0.942 ± 0.042
zoo	AllSetTransformer	0.001	0	-	-	64	64	1	-	-	**0.973 ± 0.036**
zoo	HGNN	0.01	0	-	-	64	128	-	-	-	0.954 ± 0.030
zoo	HyperGCN	0.001	0.0005	-	-	256	256	-	-	-	0.423 ± 0.000
zoo	UniGCNII	0.01	0	-	-	256	64	4	-	-	0.969 ± 0.016
zoo	ED-HNN	0.001	0	-	0,0,1	256	128	-	0	-	0.965 ± 0.028
Actor	HGNAN	0.001	0	0	3	64	-	-	-	1	**0.863 ± 0.007**
Actor	AllDeepSets	0.01	0	-	-	256	64	-	-	-	0.838 ± 0.003
Actor	AllSetTransformer	0.001	0.0005	-	-	256	64	8	-	-	0.836 ± 0.002
Actor	HGNN	0.01	0.0005	-	-	128	64	-	-	-	0.748 ± 0.003
Actor	HyperGCN	0.001	0	-	-	64	64	-	-	-	0.630 ± 0.000
Actor	UniGCNII	0.001	0	-	-	128	128	4	-	-	0.822 ± 0.003
Actor	ED-HNN	0.001	0	-	0,1,1	512	256	-	0	-	0.857 ± 0.005
Pokec	HGNAN	0.001	0	0	3	64	-	-	-	2	**0.636 ± 0.010**
Pokec	AllDeepSets	0.01	0.0005	-	-	128	256	-	-	-	0.578 ± 0.011
Pokec	AllSetTransformer	0.001	0.0005	-	-	256	64	8	-	-	0.564 ± 0.010
Pokec	HGNN	0.001	0	-	-	64	64	-	-	-	0.552 ± 0.020
Pokec	HyperGCN	0.01	0	-	-	256	256	-	-	-	0.534 ± 0.012
Pokec	UniGCNII	0.01	0.0005	-	-	64	64	8	-	-	0.573 ± 0.016
Pokec	ED-HNN	0.001	0	-	0,1,1	512	256	-	0.5	-	0.628 ± 0.017

C.2 Hyperedge prediction tasks

Following Chen & Liu (2024), we consider the following configurations for all methods: learning rate = $\{0.001, 0.01\}$ and weight decay = $\{0, 0.0005\}$.

For HGNAN, we set dropout rate = $\{0.0, 0.5\}$, number of hidden channels = $\{32, 64, 128\}$ and $s_{\max} = \{1, 2, 3\}$.

For CHESHIRE we set embedding dimension = $\{128, 256\}$, convolutional dimension = $\{64, 128\}$, Chebyshev polynomial order $k = \{3, 5\}$, and dropout probability $p = \{0.1, 0.2\}$.

For NHP, we tune the embedding dimension = $\{128, 256\}$ and convolutional dimension = $\{64, 128\}$.

For HyperSAGNN. we tune the embedding dimension = $\{128, 256\}$, convolutional dimension = $\{64, 128\}$, and number of attention heads = $\{1, 3\}$.

Meanwhile, we tune the hyperparameters separately for each dataset. The selected hyperparameters are reported in Table 7. We show the test accuracy in Table 2 and test AUC, AUPRC, and F1-score in Table 8. HGNAN-edge outperforms all baseline models on all evaluation metrics and datasets. All experiments are run on NVIDIA V100 GPU.

Table 7: Best hyperparameters for each dataset for hyperedge prediction tasks

Dataset	Model	lr	wd	emb_dim	k	heads	dropout	layer	hidden_channel	$s_{\max}$	Test Acc
iAF1260b	CHESHIRE	0.001	0.0005	256	5	-	0.5	-	128	-	0.834 ± 0.050
iAF1260b	NHP	0.01	0.0005	128	-	-	-	-	64	-	0.732 ± 0.076
iAF1260b	HyperSAGNN	0.001	0.0005	256	-	4	-	-	128	-	0.730 ± 0.075
iAF1260b	HGNAN	0.001	0	-	-	-	0.5	3	32	1	**0.935 ± 0.069**
iJR904	CHESHIRE	0.001	0.0005	256	5	-	0.5	-	128	-	0.850 ± 0.068
iJR904	NHP	0.001	0	1	-	-	-	-	128	-	0.690 ± 0.090
iJR904	HyperSAGNN	0.001	0.0005	128	-	4	-	-	128	-	0.753 ± 0.056
iJR904	HGNAN	0.001	0	-	-	-	0	5	128	2	**0.959 ± 0.018**
iSB619	CHESHIRE	0.001	0.0005	128	5	-	0.5	-	128	-	0.831 ± 0.038
iSB619	NHP	0.001	0.0005	1	-	-	-	-	128	-	0.687 ± 0.055
iSB619	HyperSAGNN	0.001	0	256	-	4	-	-	64	-	0.729 ± 0.162
iSB619	HGNAN	0.001	0	-	-	-	0	5	64	1	**0.970 ± 0.018**
iYO844	CHESHIRE	0.001	0.0005	256	3	-	0.5	-	64	-	0.893 ± 0.047
iYO844	NHP	0.001	0	1	-	-	-	-	128	-	0.747 ± 0.043
iYO844	HyperSAGNN	0.001	0.0005	256	-	4	-	-	128	-	0.808 ± 0.045
iYO844	HGNAN	0.001	0	-	-	-	0.5	5	128	2	**0.937 ± 0.058**

Table 8: Additional testing metrics for hyperedge prediction tasks. Bold values highlight the best result for each dataset. HGNAN-edge outperforms all baseline models across all datasets.

	iAF1260b			iJR904			iSB619			iYO844		
Model	AUROC	AUPRC	F1	AUROC	AUPRC	F1	AUROC	AUPRC	F1	AUROC	AUPRC	F1
CHESHIRE	0.7844	0.7339	0.7012	0.7802	0.7497	0.7514	0.7591	0.7309	0.7427	0.8312	0.7934	0.7918
NHP	0.7566	0.7323	0.7130	0.7312	0.6901	0.7003	0.7183	0.6872	0.6961	0.7791	0.7470	0.7402
HyperSAGNN	0.7831	0.7303	0.7000	0.7843	0.7529	0.7505	0.7548	0.7290	0.7337	0.8424	0.8084	0.8085
HGNAN-edge	**0.9951**	**0.9961**	**0.9774**	**0.9990**	**0.9991**	**0.9889**	**0.9954**	**0.9957**	**0.9726**	**0.9717**	**0.9717**	**0.9574**

D Experiment on Synthetic Dataset

To validate that our model could learn the complex feature shape and distance functions, we build a hypergraph-level classification dataset whose data-generating process is isomorphic to HGNAN-edge.

Construction of the hypergraph. The hypergraph is consistent of $E = 2C + L$ hyperedges. Hyperedge–hyperedge intersection graph G is created with two sparse clusters of size C each, stitched by a chain of length L whose chain head connects to the left cluster, tail to the right. Each node of G represents a hyperedge in the hypergraph. The node features $z_i \in \mathbb{R}^2$ are sampled i.i.d from $U(-1, 1)$. The hyperedge feature is the mean of its member node features: $x_e = \frac{1}{|V_e|} \sum_{i \in V_e} z_i$. The ground-truth mechanism is comprised of two fixed feature transforms and weights: $f_1(x_1) = -x_1, f_2(x_2) = \sin(\pi x_2)$ and $w_1 = 2, w_2 = 1$. The ground-truth distance function follows an exponential form: $\rho(d) = \exp(-\frac{1}{2}d)$. For each hyperedge, its logit is calculated using the same logic as (7). We standardize the logits using z-score and add Guassian noise $\mathcal{N}(0, 0.1)$ to it. It is then passed through sigmoid and threshold at 0.5 to produce hyperedge-level labels $y \in \{0, 1\}^E$.

Measurements and Results. To align the scales, we deploy an affined transformation to the results of our model. The coefficients of the transformation a and b are obtained by solving this simple linear regression: $\sum_i (a\hat{f}(x_i) + b - f(x_i))^2$, where where f is the ground truth function. We plot the comparison and report the MSE, R^2 and Pearson Correlation on the aligned curve. We can see that HGNAM-edge successfully recovered most of the underlying mechanism of this synthetic dataset.

Table 9: Alignment performance of HGNAM-edge compared with ground-truth. Lower MSE is better; higher R^2 and Pearson correlation are better.

Method	MSE	R^2	Pearson Corr.
Feature_1	0.0013	0.9899	0.9949
Feature_2	0.0014	0.9943	0.9972
Distance	0.0005	0.9905	0.9952

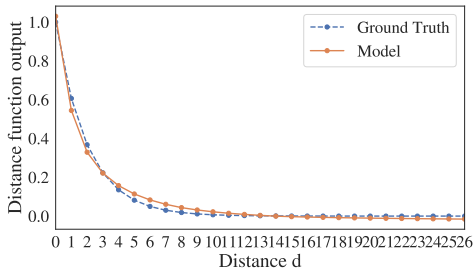
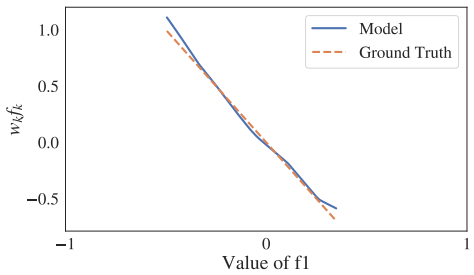
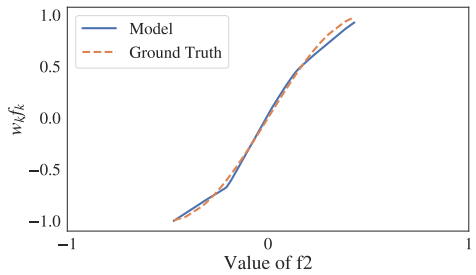


Figure 7: Distance function alignment



(a) Feature 1 shape function alignment



(b) Feature 2 shape function alignment

Figure 8: Feature shape function alignment

E Analysis of Relative Feature Importance

In Equation (4) and (7), we introduce a feature specific weight w_k for each feature k. These weights are passed through an exponential function to ensure positive and finally normalized using softmax so that $\sum_{i=1}^{k} w_k = 1$. In this way, the model could learn the weights in the context of other features. In this way, we can interpret them as dataset-level relative feature importance.

Figure 9 reports global feature importance for predicting the seven classes in the UCI Zoo dataset, and the ordering is reasonable. The top variables—eggs, legs, backbone, breathes, and toothed—create coarse, high-information splits: eggs cleanly separates mammals from most other classes; legs distinguishes insects and some invertebrates from vertebrates ; backbone isolates invertebrates; breathes helps separate fish; and toothed differentiates mammals from birds. Mid-ranked features such as feathers, airborne, fins, aquatic, tail, and catsize are very informative but mainly within specific subsets (e.g., feathers/airborne nearly determine birds), so their global contribution is diluted. Milk or hair score lower because they are strongly correlated with the higher-ranked variables (e.g., eggs/toothed/backbone). Traits like predator and venomous are rare and cross multiple classes, so they naturally receive low importance. Overall, the ranking mirrors both the dataset's structure and known biases of global importance metrics, making the plot reasonable and interpretable.

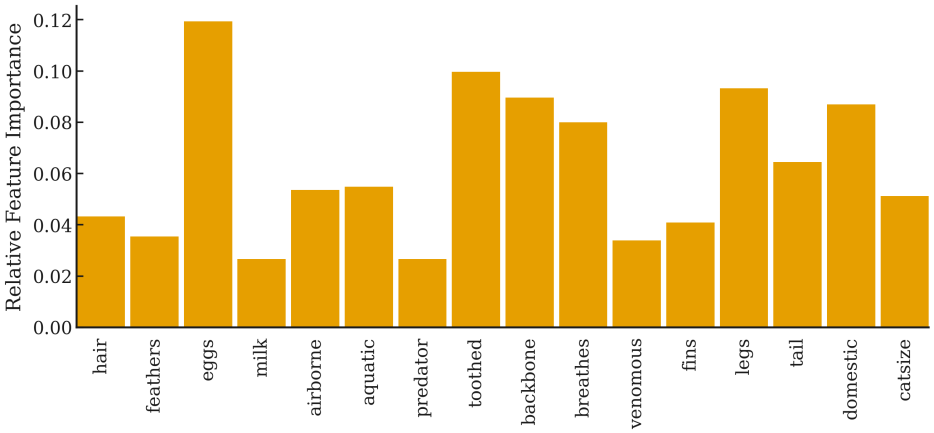


Figure 9: Relative feature importance w_k for Zoo dataset

F Cross-dataset Validation for BiGG Datasets

We conduct cross-dataset validation to evaluate the model's ability to transfer knowledge across different GEM models. Specifically, we apply a model trained on one dataset (e.g., iAF1260b) to test on another. Each model is the best performer selected as described in Appendix C.2. After obtaining the best model trained on one dataset (e.g., iAF1260b), we fine-tune it for one additional epoch using the training set of a target dataset (e.g., iYO844). Since HGNAN is a NAM-based model, its feature-shape and distance functions maintain the same dimension across datasets, allowing direct transfer of the trained model to another dataset. Other baseline methods, however, may encounter input shape mismatches during this procedure. Therefore, for these methods, we first extract each dataset's incidence matrix, compute the union of all metabolites (i.e., nodes) across the four datasets, and then pad each incidence matrix with rows of value 0 for any metabolites that are absent. This ensures that all matrices have the same dimension, enabling fair cross-dataset validation.

According to Table 10, HGNAN outperforms baseline methods across almost all test settings. In one transfer setting, where the model trained on iAF1260b is applied to iJR904, HGNAN-edge achieves near-perfect classification performance with an AUCROC of 0.988. Even for the worst transfer, which is from iAF1260b to iYO844, the performance is above the average among all models. On average, HGNAN-edge achieves an AUROC of approximately 0.818 across different transfer settings, significantly outperforming baseline methods. CHESHIRE and NHP perform pooly on this task. While HyperSAGNN is competitive in some transfer settings, it has higher variance compared to HGNAN-edge. These results demonstrate that HGNAN-edge effectively captures the underlying metabolic network structure, enabling more robust transfer learning performance.

Table 10: Cross-dataset validation results for the BiGG datasets. AUROC is used as the performance metric. Bold values highlight the best result for each pair of transfers.

Training Data	Testing Data	NHP	HyperSAGNN	CHESHIRE	HGNAN-edge
iAF1260b	iJR904	0.6213±0.4679	0.7769±0.2035	0.6785±0.4400	**0.9884±0.0216**
iAF1260b	iYO844	0.6284±0.3968	**0.7479±0.2502**	0.5059±0.3897	0.6254±0.1364
iAF1260b	iSB619	0.6460±0.2731	0.7445±0.3905	0.5455±0.5925	**0.8297±0.1204**
iJR904	iAF1260b	0.6476±0.1190	0.7657±0.0308	0.6717±0.2569	**0.8679±0.3468**
iJR904	iYO844	0.6667±0.3766	0.7641±0.3306	0.6406±0.1578	**0.8967±0.1469**
iJR904	iSB619	0.6291±0.2954	**0.7137±0.4300**	0.5017±0.3161	0.6652±0.1125
iYO844	iAF1260b	0.6551±0.1080	0.7627±0.4770	0.6081±0.5619	**0.9624±0.0153**
iYO844	iJR904	0.6143±0.5227	**0.8191±0.1564**	0.6637±0.5340	0.6534±0.0866
iYO844	iSB619	0.6420±0.2554	**0.7950±0.5768**	0.5888±0.2448	0.7216±0.2165
iSB619	iAF1260b	0.6754±0.4719	0.7333±0.3460	0.5827±0.4503	**0.8345±0.3116**
iSB619	iJR904	0.5798±0.0419	0.8332±0.4215	0.5981±0.1856	**0.8661±0.3416**
iSB619	iYO844	0.6598±0.2153	0.8218±0.5518	0.6404±0.1902	**0.9013±0.3363**

G LLM use declaration

In this paper, we used LLM to help us only with polishing our writing.