ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10446248

HYPERGRAPH TRANSFORMER FOR SEMI-SUPERVISED CLASSIFICATION

Zexi Liu[1], Bohan Tang[2], Ziyuan Ye[3], Xiaowen Dong[2], Siheng Chen[1,4], Yanfeng Wang[4,1]

[1]Shanghai Jiao Tong University, [2]University of Oxford, [3]The Hong Kong Polytechnic University, [4]Shanghai AI Laboratory

ABSTRACT

Hypergraphs play a pivotal role in the modelling of data featuring higher-order relations involving more than two entities. Hypergraph neural networks emerge as a powerful tool for processing hypergraph-structured data, delivering remarkable performance across various tasks, e.g., hypergraph node classification. However, these models struggle to capture global structural information due to their reliance on local message passing. To address this challenge, we propose a novel hypergraph learning framework, HyperGraph Transformer (HyperGT). HyperGT uses a Transformer-based neural network architecture to effectively consider global correlations among all nodes and hyperedges. To incorporate local structural information, HyperGT has two distinct designs: i) a positional encoding based on the hypergraph incidence matrix, offering valuable insights into node-node and hyperedge-hyperedge interactions; and ii) a hypergraph structure regularization in the loss function, capturing connectivities between nodes and hyperedges. Through these designs, HyperGT achieves comprehensive hypergraph representation learning by effectively incorporating global interactions while preserving local connectivity patterns. Extensive experiments conducted on real-world hypergraph node classification tasks showcase that HyperGT consistently outperforms existing methods, establishing new state-of-the-art benchmarks. Ablation studies affirm the effectiveness of the individual designs of our model.

***Index Terms*—** Hypergraph Neural Networks, Transformer, Node Classification

1. INTRODUCTION

Hypergraphs serve as a crucial tool for modelling data with higher-order relationships involving more than two nodes [1, 2, 3, 4, 5]. Hypergraph neural networks leverage message passing over hypergraph structures to enhance node feature learning, leading to remarkable performances in various tasks like node classification [6]. However, these message-passing-based models suffer from the oversmoothing issue, particularly when increasing the model depth to capture global information [7, 8]. Recently, some Transformer-based methods have emerged for hypergraph-structured data [9, 10, 11]. Nonetheless, these methods predominantly use the neighborhood attention to update node features within individual hyperedges, thereby constraining their ability to fully exploit global information. In summary, the existing literature lacks efficient methods for exploiting the global information inherent in hypergraph-structured data.

To fill this gap, we propose HyperGraph Transformer (HyperGT), a novel learning framework that efficiently incorporates both global and local structural information in the hypergraph. HyperGT consists of three key components. First, the hypergraph attention operation explores correlations among all nodes and hyperedges, enabling efficient capture of global information. Second, the positional encoding based on the hypergraph incidence matrix integrates local node-node connectivities into the forward propagation of HyperGT. Third, the hypergraph structure regularization in the loss function, which is derived from the hypergraph star-expansion, encourages HyperGT to capture local node-hyperedge interactions.

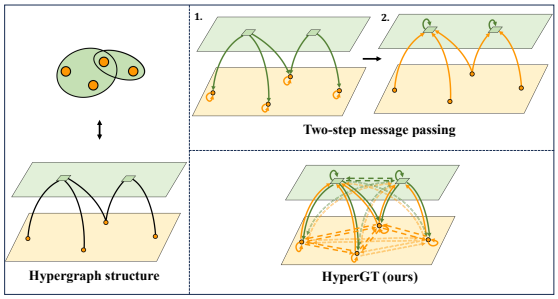

Fig. 1: Left: A hypergraph and the associated hypergraph star-expansion. Hypergraph star-expansion fully preserves the structural information of the hypergraph. Right: Illustration of the previous message-passing method and our HyperGT framework for the hypergraph depicted on the left. We build interactions among all nodes and hyperedges by hypergraph attention.

We extensively evaluate our model on four real-world hypergraph semi-supervised node classification benchmarks. Results are promising: 1) HyperGT achieves state-of-the-art (SOTA) performance across all datasets, notably surpassing previous SOTA hypergraph neural networks by nearly 3% in classification accuracy on the Walmart dataset; and 2) Ablation studies unequivocally demonstrate the effectiveness of each of three proposed components, as an example: the proposed positional encoding mechanism and structure regularization boost the classification accuracy of a vanilla Transformer by approximately 25% on the Walmart dataset.

2. RELATED WORK

Hypergraph neural networks. Numerous studies focus on developing neural networks to process data on hypergraphs. Most existing hypergraph neural networks (HGNNs) [12, 13, 14, 15, 16, 17, 18] follow a two-step message passing paradigm: first, node features are sent to corresponding hyperedges to learn hyperedge embeddings, then the learned hyperedge embeddings are sent back to nodes to learn node embeddings. A recent study [7] highlights a common challenge encountered in message-passing-based hypergraph neural networks: the oversmoothing issue. This issue leads to increasingly homogeneous node embeddings as the depth of models increases, impeding their capacity to effectively capture valuable global information, such as long-range dependencies, through deeper architectures. Consequently, the current landscape lacks hypergraph neural networks capable of efficiently exploiting the inherent global information present in hypergraph-structured data.

Transformer for structured data. Recently, the Transformer

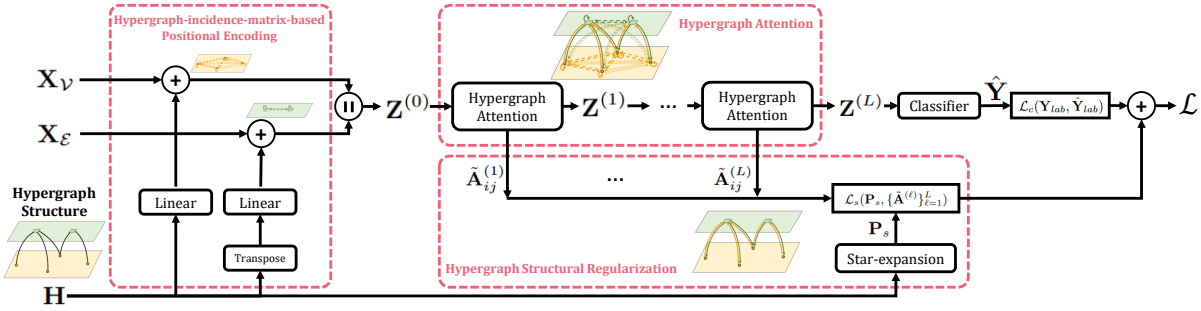

Fig. 2: System architecture of HyperGT, taking node features $\mathbf{X}_{\mathcal{V}}$, hyperedge features $\mathbf{X}_{\mathcal{E}}$, and the hypergraph incidence matrix $\mathbf{H}$ as inputs. There are three key components: 1) Positional encoding using the hypergraph incidence matrix for node-node and hyperedge-hyperedge interactions; 2) Hypergraph attention module creating connections from one node/hyperedge to any other nodes/hyperedges; and 3) Hypergraph structure regularization for node-hyperedge connectivities.

architecture has been adapted to effectively process graph-structured data by incorporating both local and global graph structure information, leading to promising outcomes [19, 20, 21, 22]. However, in the field of hypergraph machine learning, the Transformer is merely used to learn local attention mechanisms inside a hyperedge [9, 10, 11]. To our knowledge, our work represents the first effort in harnessing the Transformer architecture to simultaneously capture both local and global information within hypergraph-structured data.

3. PRELIMINARY

Hypergraphs. We denote a hypergraph as a triplet $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \mathbf{H}\}$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ is the node set with $|\mathcal{V}| = n$, $\mathcal{E} = \{e_1, e_2, \cdots, e_m\}$ is the hyperedge set with $|\mathcal{E}| = m$, and $\mathbf{H} \in \{0,1\}^{n\times m}$ is an incidence matrix in which $\mathbf{H}_{ij} = 1$ indicates that hyperedge j contains node i and $\mathbf{H}_{ij} = 0$ otherwise.

Semi-supervised classification on hypergraphs. We focus on the semi-supervised hypergraph node classification task. Let $\mathbf{X}_{\mathcal{V}} = [\mathbf{x}_{v_1}^\top, \mathbf{x}_{v_2}^\top, \cdots, \mathbf{x}_{v_n}^\top]^\top \in \mathbb{R}^{n\times d}$ be the node features, which is a matrix that contains d-dimensional features, $\mathcal{V}_{lab}$ be the set of labeled nodes with ground truth labels $\mathbf{Y}_{lab} = \{\mathbf{y}_v\}_{v\in\mathcal{V}_{lab}}$, where $\mathbf{y}_{v_i} \in \{0,1\}^c$ be a one-hot label, and $\mathcal{V}_{un} = \mathcal{V} \setminus \mathcal{V}_{lab}$ be a set of unlabeled nodes. Given node features $\mathbf{X}_{\mathcal{V}}$, labels $\mathbf{Y}_{lab}$, and the hypergraph structure $\mathbf{H}$, we aim to classify nodes in $\mathcal{V}_{un}$.

Transformer. In a standard Transformer, the forward propagation begins with a positional encoding mechanism, followed by a sequence of identical Transformer layers, each of which consists of a multi-head self-attention modules (MHSA) and a position-wise fully connected feed-forward network (FFN) [23].

The positional encoding mechanism adds the given structural information to the input embeddings. Let $\mathbf{X} \in \mathbb{R}^{n\times d}$ be the input features and $\mathbf{P} \in \mathbb{R}^{n\times d}$ be features embedding the input structural information. The initial structural-augmented embedding is

$$\mathbf{Z} = \mathbf{X} + \mathbf{P} \in \mathbb{R}^{n\times d},$$

whose i-th row $\mathbf{z}_i \in \mathbb{R}^d$ is the initial embedding of the i-th input.

The MHSA, a pivotal component for information exchange among entities, update embeddings for each entity. Let $\mathbf{Z}$ be the input embedding. Excluding bias terms and multi-head details for simplicity, the self-attention module works as:

$$\begin{aligned}
\mathbf{Q} &= \mathbf{Z}\mathbf{W}_{\mathbf{Q}}, \quad \mathbf{K} = \mathbf{Z}\mathbf{W}_{\mathbf{K}}, \quad \mathbf{V} = \mathbf{Z}\mathbf{W}_{\mathbf{V}}, \\
\tilde{\mathbf{A}} &= \operatorname{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right), \\
\mathbf{Z}' &= \tilde{\mathbf{A}}\mathbf{V},
\end{aligned}$$

where $\mathbf{W}_{\mathbf{Q}}, \mathbf{W}_{\mathbf{K}}, \mathbf{W}_{\mathbf{V}} \in \mathbb{R}^{d\times d_k}$ are learnable matrices, and the weight matrix $\tilde{\mathbf{A}} \in \mathbb{R}^{n\times n}$ is formed from the similarity calculation between any pair of inputs to update their features.

The FFN after the MHSA serves as a pointwise non-linear transformation for $\mathbf{Z}'$. Here we omit the details for brevity.

4. HYPERGRAPH TRANSFORMER

In this section, we introduce our HyperGraph Transformer (HyperGT) tailored for the node classification task, as illustrated in Fig. 2. Leveraging both node and hyperedge features, we enhance these features by incorporating node-node and hyperedge-hyperedge interactions through a positional encoding mechanism driven by the hypergraph incidence matrix, thus generating input embeddings (Section 4.1). These embeddings are thereon fed in the hypergraph attention module, which establishes dense connections between nodes and hyperedges (Section 4.2). Finally, we compute the training loss $\mathcal{L}$ by combining the traditional supervised classification loss $\mathcal{L}_c$ with a hypergraph structure regularization loss $\mathcal{L}_s$, adept at capturing node-hyperedge interrelations (Section 4.3 and Section 4.4).

4.1. Hypergraph incidence matrix based positional encoding

HyperGT simultaneously incorporates node features and hyperedge features as the input. Let $\mathbf{X}_{\mathcal{E}} = [\mathbf{x}_{e_1}^\top, \mathbf{x}_{e_2}^\top, \cdots, \mathbf{x}_{e_m}^\top]^\top \in \mathbb{R}^{m\times d}$ denote the hyperedge features where we let the dimension of hyperedge features be consistent with that of node features. If hyperedge features are not available, to make node and hyperedge features correlated, we initialize hyperedge features by computing the average of features of nodes within that hyperedge. The input features are:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{\mathcal{V}} \\ \mathbf{X}_{\mathcal{E}} \end{bmatrix}, \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{(n+m)\times d}$ means each node and each hyperedge serves as the input of the global attention module.

The functionality of the positional encoding is to provide structural insights for nodes and hyperedges, enabling our model to become aware of the hypergraph structure. To achieve this, we inject structural information to the input features defined in Eq. (1). Specifically, since the hypergraph incidence matrix $\mathbf{H}$ inherently embodies the hypergraph structure itself, we leverage it to design learnable positional encoding for each node and hyperedge.

Let $\mathbf{P}_{\mathcal{V}}$ and $\mathbf{P}_{\mathcal{E}}$ denote the positional encodings for nodes and hyperedges, respectively. Then, our positional encoding works as

$$\mathbf{P}_{\mathcal{V}} = \mathbf{H}\mathbf{W}_{\mathcal{V}}, \mathbf{P}_{\mathcal{E}} = \mathbf{H}^{\top}\mathbf{W}_{\mathcal{E}},$$

where $\mathbf{W}_{\mathcal{V}} \in \mathbb{R}^{n\times d}, \mathbf{W}_{\mathcal{E}} \in \mathbb{R}^{m\times d}$ embeds the nodes and hyperedges structure via a learnable linear projection to generate positional encodings. Then, it is added to the input features directly:

$$\mathbf{Z}^{(0)} = \mathbf{X} + \mathbf{P} = \begin{bmatrix} \mathbf{X}_{\mathcal{V}} + \mathbf{P}_{\mathcal{V}} \\ \mathbf{X}_{\mathcal{E}} + \mathbf{P}_{\mathcal{E}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_{\mathcal{V}} + \mathbf{H}\mathbf{W}_{\mathcal{V}} \\ \mathbf{X}_{\mathcal{E}} + \mathbf{H}^{\top}\mathbf{W}_{\mathcal{E}} \end{bmatrix} \in \mathbb{R}^{(n+m)\times d}.$$

By using the positional encoding, our model can effectively leverage the hypergraph structural information.

To further clarify the advantage of our proposed positional encoding, we show that it can represent the local structure of the hypergraph. Define the structure information of each node to be the hyperedges that contain this node, and the structure information of each hyperedge as the nodes contained within this hyperedge. Then, we have the following result:

Theorem 1. Let $\mathbf{p}_u$, $\mathbf{p}_v$ be the positional encoding of nodes u and v, respectively. Then, $\|\mathbf{p}_u - \mathbf{p}_v\|_2 \leq C\sqrt{N_e}$, where C is a constant and N_e is the number of hyperedges only with either node u or v.

Proof.

$$\begin{aligned} \|\mathbf{p}_u - \mathbf{p}_v\|_2 &= \|\mathbf{h}_u W_{\mathcal{V}} - \mathbf{h}_v W_{\mathcal{V}}\|_2 \\ &= \|(\mathbf{h}_u - \mathbf{h}_v)W_{\mathcal{V}}\|_2 \\ &\leq \|W_{\mathcal{V}}\|_2 \cdot \|(\mathbf{h}_u - \mathbf{h}_v)\|_2 \\ &= \sigma_{max}(W_{\mathcal{V}})\sqrt{|\{e|u \notin e, v \in e \cap u \in e, v \notin e\}|} \\ &= C\sqrt{N_e}, \end{aligned} \tag{2}$$

where $\mathbf{h}_u, \mathbf{h}_v$ are the u-th and v-th row vectors of $\mathbf{H}$, $\|\cdot\|_2$ denotes the 2-norm of a vector or a matrix, and $\sigma_{max}(\cdot)$ denotes the largest singular value of a matrix. From Eq. (2), we know that the distance between the positional encodings of the two nodes is bounded by the number of hyperedges that only contain either node u or v, while the latter represents the similarity between the structural information of node u and v. We have similar results for hyperedges as well.

In summary, positional encoding enhances structural information utilization in our model for nodes and hyperedges.

4.2. Hypergraph attention

The functionality of the hypergraph attention is to explore correlations among all nodes and hyperedges, capturing global information. Mathematically, let $\mathbf{Z}^{(\ell)}$ be the feature embedding at the ℓ-th hypergraph attention layer with $\mathbf{z}_i^{\ell}$ the hidden representation of instance i and $\tilde{\mathbf{A}}^{(\ell)} \in \mathbb{R}^{(n+m)\times(n+m)}$ be the softmax attention matrix at the ℓ-th layer. The (i, j)-element of $\tilde{\mathbf{A}}^{(\ell)}$ is obtained as

$$\tilde{\mathbf{A}}_{ij}^{(\ell)} = \frac{\exp((\mathbf{z}_i^{(\ell)}\mathbf{W}_{\mathbf{Q}}^{(\ell)})(\mathbf{z}_j^{(\ell)}\mathbf{W}_{\mathbf{K}}^{(\ell)})^{\top})}{\sum_{k=1}^{n+m}\exp((\mathbf{z}_i^{(\ell)}\mathbf{W}_{\mathbf{Q}}^{(\ell)})(\mathbf{z}_k^{(\ell)}\mathbf{W}_{\mathbf{K}}^{(\ell)})^{\top})},$$

where $\mathbf{W}_{\mathbf{Q}}^{(\ell)}, \mathbf{W}_{\mathbf{K}}^{(\ell)}, \mathbf{W}_{\mathbf{V}}^{(l)}$ are three learnable matrices at the l-th layer. After attention is calculated between any two inputs, the representation of the i-th instance is updated as

$$\mathbf{z}_i^{(\ell+1)} = \sum_{j=1}^{n+m} \tilde{\mathbf{A}}_{ij}^{(\ell)} \cdot (\mathbf{z}_j^{(\ell)}\mathbf{W}_{\mathbf{V}}^{(\ell)}).$$

This creates a dense connection from one node/hyperedge to any other nodes/hyperedges in only a single step.

This design considers a comprehensive attention among nodes and hyperedges. It has two distinct differences from previous works. First, in previous works, the interaction between two nodes is established along the node-hyperedge-node pathways. Thus, information is propagated slowly through the same local kernel, causing the oversmoothing issue. In comparison, we enable the pairwise interactions between all instances (nodes and hyperedges), which makes information propagation more efficient. Second, previous methods do not consider interactions between hyperedges, while we establish the possible attentions between hyperedges to get better hyperedge embeddings, further crafting better node representations.

4.3. Hypergraph structure regularization

To further encourage the model to leverage local node-hyperedge interaction, we introduce a hypergraph structure regularization. The key idea is to utilize the node-hyperedge connection prior to guide the training of the attention matrix.

To obtain a supervision for the softmax attention matrix, we build a probabilistic transition matrix based on the star-expansion structure of the hypergraph. Mathematically, let $\mathcal{G} = \{\mathcal{V}_s, \mathcal{E}_s, \mathbf{A}_s\}$ be the hypergraph star-expansion, where the vertex set is augmented with hyperedges $\mathcal{V}_s = \mathcal{V} \cup \mathcal{E}$, and vertices and hyperedges are connected by their incident relations $\mathcal{E}_s = \{(v, e) \mid v \in e, v \in \mathcal{V}, e \in \mathcal{E}\}$. The adjacency matrix $\mathbf{A}_s \in \{0, 1\}^{(n+m)\times(n+m)}$ embeds this star-expansion graph structure:

$$\mathbf{A}_s = \begin{bmatrix} 0_{|\mathcal{V}|} & \mathbf{H} \\ \mathbf{H}^{\top} & 0_{|\mathcal{E}|} \end{bmatrix}.$$

Notably, $\mathbf{A}_s$ is uniquely associated with $\mathbf{H}$, and one can reconstruct the original hypergraph from the star-expansion structure without any information loss. Therefore, the star-expansion structure of the hypergraph fully preserves the structural information of the hypergraph (i.e., $\mathbf{H}$). The corresponding probabilistic transition matrix is $\mathbf{P}_s = \mathbf{D}_s^{-1}\mathbf{A}_s \in \mathbb{R}^{(n+m)\times(n+m)}$, where $\mathbf{D}_s$ is a diagonal matrix with $(\mathbf{D}_s)_{ii} = \sum_j (\mathbf{A}_s)_{ij}$. The (i, j)-th element in $\mathbf{P}_s$ represents the probability that the i-th instance transits to the j-th instance.

This transition matrix $\mathbf{P}_s$ is an appropriate supervision for $\tilde{\mathbf{A}}^{(\ell)}$ for three reasons: i) $\mathbf{P}_s$ can reflect the complete relations between nodes and hyperedges; ii) through the star-expansion, $\mathbf{P}_s$ and $\tilde{\mathbf{A}}^{(\ell)}$ have the same dimension; and iii) both $\mathbf{P}_s$ and $\tilde{\mathbf{A}}^{(\ell)}$ have probabilistic meanings. Then, the hypergraph structure loss is defined based on the cross-entropy between $\mathbf{P}_s$ and $\tilde{\mathbf{A}}^{(\ell)}$:

$$\begin{aligned} \mathcal{L}_s(\mathbf{P}_s, \{\tilde{\mathbf{A}}^{(\ell)}\}_{\ell=1}^{L}) &= -\frac{1}{(n+m)L}\sum_{\ell=1}^{L}\sum_{i,j=1}^{n+m}(\mathbf{P}_s)_{i,j}\log\tilde{\mathbf{A}}_{ij}^{(\ell)} \\ &= -\frac{1}{(n+m)L}\sum_{l=1}^{L}\sum_{(i,j)\in\mathcal{E}_s}\frac{1}{d_i}\log\tilde{\mathbf{A}}_{ij}^{(\ell)}, \end{aligned}$$

where L is the number of layers in HyperGT and d_i is the degree of instance i in the star-expansion structure of the hypergraph.

4.4. Training strategy

For node classification, given training labels $\mathbf{Y}_{lab} = \{\mathbf{y}_u\}_{u\in\mathcal{V}_{lab}}$, where $\mathcal{V}_{lab}$ denotes the set of labeled nodes, we train our model with the cross-entropy loss function:

$$\mathcal{L}_c(\mathbf{Y}_{lab}, \hat{\mathbf{Y}}_{lab}) = -\frac{1}{|\mathcal{V}_{lab}|}\sum_{u\in\mathcal{V}_{lab}}\mathbf{y}_u\log(\hat{\mathbf{y}}_u^{\top}),$$

where $\log(\cdot)$ is an element-wise logarithmic function. Considering our hypergraph structure loss, the final objective is: $\mathcal{L} = \mathcal{L}_c + \lambda\mathcal{L}_s$, where λ is a coefficient to balance the two losses. This learning objective is minimized to update trainable parameters in HyperGT.

Table 1: Properties of datasets.

	Congress	Senate	Walmart	House
$\lvert\mathcal{V}\rvert$	1718	282	88860	1290
$\lvert\mathcal{E}\rvert$	83105	315	69906	341
# features	100	100	100	100
# class	2	2	11	2

Table 2: Results for the tested datasets: Mean accuracy (%) ± standard deviation. Experiments show that HyperGT outperforms existing state-of-the-art hypergraph neural networks.

	Congress	Senate	Walmart	House
HGNN	91.26 ± 1.15	48.59 ± 4.52	62.00 ± 0.24	61.39 ± 2.96
HCHA	90.43 ± 1.20	48.62 ± 4.41	62.35 ± 0.26	61.36 ± 2.53
HNHN	53.35 ± 1.45	50.93 ± 6.33	47.18 ± 0.35	67.80 ± 2.59
HyperGCN	55.12 ± 1.96	42.45 ± 3.67	44.74 ± 2.81	48.32 ± 2.93
UniGCNII	94.81 ± 0.81	49.30 ± 4.25	54.45 ± 0.37	67.25 ± 2.57
HyperND	74.63 ± 3.62	52.82 ± 3.20	38.10 ± 3.86	51.70 ± 3.37
AllDeepSets	91.80 ± 1.53	48.17 ± 5.67	64.55 ± 0.33	67.82 ± 2.40
AllSetTransformer	92.16 ± 1.05	51.83 ± 5.22	65.46 ± 0.25	69.33 ± 2.20
ED-HNN	95.00 ± 0.99	64.79 ± 5.14	66.91 ± 0.41	72.45 ± 2.28
HyperGT(Ours)	**95.09 ± 1.16**	**65.49 ± 5.11**	**69.83 ± 0.39**	**74.55 ± 1.99**

5. EXPERIMENTS

5.1. Experimental settings

Datasets. We use four datasets from previous hypergraph neural networks research[12]. In the House dataset, nodes represent US House of Representatives members, with hyperedges corresponding to committee memberships. Node labels indicate the political party. In Walmart, hyperedges represent co-purchased product sets, and nodes are products labeled with categories. In Congress and Senate dataset, nodes are US Congresspersons labeled with political party affiliation, and hyperedges are formed by bill sponsors and co-sponsors from both chambers. Since these datasets lack node attributes, we follow Chien et al. [13] to generate node features from label-dependent Gaussian distribution, where the standard deviation of the added Gaussian features is set as 1.0. The partial statistics of all datasets used are provided in Table 1.

Baselines. We compare our method with top-performing models on these benchmarks, including HGNN [14], HCHA [16], HNHN [15], HyperGCN [18], UniGCNII [17], AllDeepSets [13], AllSetTransformer [13], HyperND [24], and ED-HNN [12]. All the hyperparameters for baselines are followed from [12].

Implementation details. Our model is built upon the implementation of Nodeformer [25], which leverages a kernerlized gumbel-softmax operator to reduce the attention computation comlexity to linearity. We identified λ's optimal value via grid-search. Regarding the hyperparameters of the $\mathcal{O}(N)$ Transformer, we set them to be consistent with NodeFormer. We randomly split the data (node indices) into training/validation/test samples using 50%/25%/25% splitting percentages as in Chien et al. [13]. We report the average prediction accuracy over ten random splits as the evaluation metric. All experiments are conducted on an RTX 3090 utilizing the PyTorch framework. Our code is available at: https://github.com/zeroxleo/HyperGT.

5.2. Results and analysis

Comparison with SOTA hypergraph neural networks. The results of our experiments on real-world hypergraph node classification benchmarks are presented in Table 2. In comparison to previous hypergraph neural networks, our proposed HyperGT demonstrates superior classification accuracy across all datasets. A key distinguishing feature of HyperGT is its incorporation of both global and local hypergraph structural information, setting it apart from prior methods that solely focus on local structural information. These results show the importance of leveraging global information for designing more effective models in processing hypergraph-structured data. We also conducted experiments on the Walmart dataset to obtain the inference speed of popular HGNNs and HyperGT. Results are presented in Table 4, indicating that HyperGT maintains competitive inference speeds, aligning with its theoretically low linear complexity, while also preserving global attention interactions.

Table 3: Ablation study on the Walmart dataset. Results demonstrate the effectiveness of each proposed components.

node PE	hyperedge PE	structure regularization	ACC
-	-	-	45.67
✓	-	-	66.51
✓	✓	-	67.63
✓	✓	✓	69.83

Table 4: Inference speeds (ms/run, mean of 1000 runs) on the Walmart dataset. Results show the competitive inference speed of HyperGT.

HGNN	HCHA	UniGCNII	AllDeepset	AllSetTransformer	HyperGT
20.301	20.942	25.015	32.603	62.788	24.882

Ablation studies. We perform a series of ablation studies on the importance of designs in our proposed HyperGT on the walmart dataset. The ablation results are presented in Table 3. We see that the use of positional encodings for nodes leads to a significant improvement, which shows that the model effectively captures the hypergraph structural information of nodes. Positional encodings of hyperedges also contribute to positive effects. With the addition of regularization, the performance is further improved, which indicates that this regularizer on hypergraph structure helps the Transformer in learning the specific connections between nodes and hyperedges. All of the components designed are helpful to our Transformer architecture for modelling hypergraph data and help to leverage useful information from input hypergraphs.

6. CONCLUSION

In this paper, we present HyperGraph Transformer (HyperGT), a novel learning framework tailored for hypergraph-structured data. HyperGT effectively tackles the challenge of simultaneously capturing global and local structural information within hypergraphs by integrating a Transformer-based architecture with hypergraph-specific components including a hypergraph-incidence-matrix-based positional encoding and a hypergraph structure regularization. Our experiments show that HyperGT outperforms existing state-of-the-art hypergraph neural networks, making it a valuable tool for various applications involving hypergraph-structured data. We leave the development of a theoretical framework to uncover the mechanisms behind the improved performance resulting from the inclusion of global structural information for future research.

Acknowledgment

This research is supported by NSFC under Grant 62171276 and 62211530109, as well as the Science and Technology Commission of Shanghai Municipal under Grant 21511100900, 22511106101, and 22DZ2229005. X.D. acknowledges support from the Oxford-Man Institute of Quantitative Finance, the EPSRC (EP/T023333/1), and the Royal Society (IEC\NSFC\211188).

7. REFERENCES

[1] Christian Bick, Elizabeth Gross, Heather A Harrington, and Michael T Schaub, "What are higher-order networks?," *SIAM Review*, vol. 65, no. 3, pp. 686–731, 2023.

[2] Chenxin Xu, Maosen Li, Zhenyang Ni, Ya Zhang, and Siheng Chen, "Groupnet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning," in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[3] Bohan Tang, Siheng Chen, and Xiaowen Dong, "Learning hypergraphs from signals with dual smoothness prior," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[4] Chenxin Xu, Yuxi Wei, Bohan Tang, Sheng Yin, Ya Zhang, Siheng Chen, and Yanfeng Wang, "Dynamic-group-aware networks for multi-agent trajectory prediction with relational reasoning," *Neural Networks*, 2023.

[5] Bohan Tang, Siheng Chen, and Xiaowen Dong, "Hypergraph structure inference from data under smoothness prior," *arXiv preprint arXiv:2308.14172*, 2023.

[6] Alessia Antelmi, Gennaro Cordasco, Mirko Polato, Vittorio Scarano, Carmine Spagnuolo, and Dingqi Yang, "A survey on hypergraph representation learning," *ACM Computing Surveys*.

[7] Guanzi Chen, Jiying Zhang, Xi Xiao, and Yang Li, "Preventing over-smoothing for hypergraph neural networks," *arXiv preprint arXiv:2203.17159*, 2022.

[8] Zijian Liu, Yang Luo, Xitong Pu, Geyong Min, and Chunbo Luo, "A multi-modal hypergraph neural network via parametric filtering and feature sampling," *IEEE Transactions on Big Data*, 2023.

[9] 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.

[10] Yu-Jung Heo, Eun-Sol Kim, Woo Suk Choi, and Byoung-Tak Zhang, "Hypergraph transformer: Weakly-supervised multi-hop reasoning for knowledge-based visual question answering," *arXiv preprint arXiv:2204.10448*, 2022.

[11] Mengran Li, Yong Zhang, Xiaoyong Li, Yuchen Zhang, and Baocai Yin, "Hypergraph transformer neural networks," *ACM Transactions on Knowledge Discovery from Data*, vol. 17, no. 5, pp. 1–22, 2023.

[12] 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.

[13] 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.

[14] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao, "Hypergraph neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, 2019, vol. 33, pp. 3558–3565.

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

[16] Song Bai, Feihu Zhang, and Philip HS Torr, "Hypergraph convolution and hypergraph attention," *Pattern Recognition*, vol. 110, pp. 107637, 2021.

[17] Jing Huang and Jie Yang, "Unignn: a unified framework for graph and hypergraph neural networks," *arXiv preprint arXiv:2105.00956*, 2021.

[18] 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*, vol. 32, 2019.

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

[20] Jinwoo Kim, Saeyoon Oh, and Seunghoon Hong, "Transformers generalize deepsets and can be extended to graphs & hypergraphs," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28016–28028, 2021.

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

[22] Liheng Ma, Chen Lin, Derek Lim, Adriana Romero-Soriano, K. Dokania, Mark Coates, Philip H.S. Torr, and Ser-Nam Lim, "Graph Inductive Biases in Transformers without Message Passing," in *Proc. Int. Conf. Mach. Learn.*, 2023.

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

[24] Francesco Tudisco, Konstantin Prokopchik, and Austin R Benson, "A nonlinear diffusion method for semi-supervised learning on hypergraphs," *arXiv preprint arXiv:2103.14867*, 2021.

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