Lifting Hypergraph Machine Learning with Gaussian Markov Random Field

Bohan Tang*, Keyue Jiang*, Siheng Chen, Xiaowen Dong

***Abstract*—Understanding the data-generating process is essential for building machine learning models that generalize well while ensuring robustness and interpretability. This paper addresses the fundamental challenge of modelling the data generation processes on hypergraphs and explores how such models can inform the design of machine learning algorithms for hypergraph data. The key of our solution is the development the hypergraph Markov random field that models the joint distribution of the node features and hyperedge features on a hypergraph through a multivariate Gaussian distribution whose covariance matrix is uniquely determined by the hypergraph structure. The proposed data-generating process provides a valuable inductive bias for various hypergraph machine learning tasks, thus enhancing the algorithm design. In this paper, we focus on two representative downstream tasks: structure inference and node classification. Accordingly, we introduce two novel frameworks: 1) an original hypergraph structure inference framework named HGSI, and 2) a novel learning framework entitled Hypergraph-MLP for classifying nodes on hypergraphs. Finally, we empirically evaluate HGSI and Hypergraph-MLP. The results are promising: 1) HGSI outperforms existing hypergraph structure inference methods on both synthetic and real-world data; and 2) On seven hypergraph node classification benchmarks, Hypergraph-MLP outperforms baselines, and exhibits remarkable runtime efficiency and robustness against structural perturbations during inference.**

***Index Terms*—Hypergraph, data-generating process, graph signal processing, graph machine learning.**

I. Introduction

Higher-order interactions that involve more than two entities widely exist in various domains, such as the co-authorships in social science [2], the spreading phenomena in epidemiology [3], and the multi-tissue gene expression in biology [4]. Hypergraphs, characterized by nodes representing entities and hyperedges denoting the higher-order interactions among these entities, serve as a valuable tool for modelling real-world data with complex higher-order interactions [5]. Recently, the hypergraph structural data have gained increasing interest from the machine learning community [6], [7].

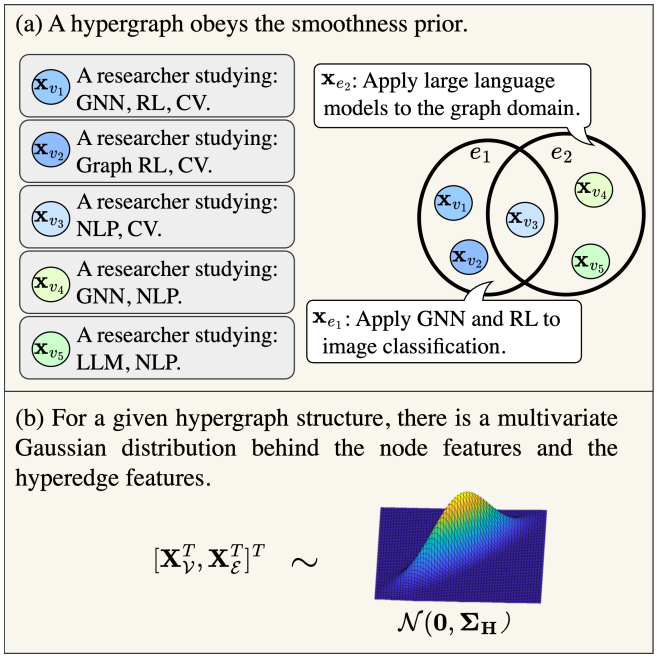


Fig. 1: (a) An academic hypergraph, where nodes are authors, hyperedges are papers, node features are authors' research interests, and hyperedge features are paper topics. Co-authors tend to share similar research interests aligned with their paper's topic. (b) For a hypergraph structure $\boldsymbol{H}$, there is a multivariate Gaussian distribution with a unique $\boldsymbol{\Sigma}_{\boldsymbol{H}}$ behind node features $\boldsymbol{X}_{\mathcal{V}}$ and hyperedge features $\boldsymbol{X}_{\mathcal{E}}$. For conciseness, we set the feature dimension as one.

The performance of a solution for hypergraph machine learning tasks, like any other machine learning (ML) tasks, depends on the compatibility between the predictive model and the underlying data-generating process (DGP) [8]–[10]. A clear understanding of the data-generating process brings meaningful solutions for hypergraph ML tasks. The current research on hypergraph machine learning intensively focuses on designing predictive models, such as hypergraph neural networks [11], [12], without meticulously investigating the inductive biases induced by the data-generating process. In parallel to these studies, we seek to advance hypergraph machine learning from a different perspective - by explicitly modelling the generation process for hypergraph data. In a nutshell, we propose a data-generating process that incorporates our belief on the hypergraph data formulation and obtain

Bohan Tang is with the Department of Engineering Science, University of Oxford, Oxford, UK. E-mail: bohan.tang@eng.ox.ac.uk.

Keyue Jiang is with the Department of Electronic and Electrical Engineering, University College London, London, UK. E-mail: uclqjia@ucl.ac.uk.

Siheng Chen is with Shanghai Jiao Tong University and Shanghai AI laboratory, Shanghai, China. E-mail: sihengc@sjtu.edu.cn.

Xiaowen Dong is with the Oxford-Man Institute and the Department of Engineering Science, University of Oxford, Oxford, UK. E-mail: xdong@robots.ox.ac.uk.

A preliminary version of this work is published as [1]. The present version includes both methodological and empirical innovations. Methodologically, we develop a framework that exploits our proposed hypergraph smoothness prior to address the hypergraph structure inference task. Empirically, we present three new findings: 1) We conduct experiments on the hypergraph structure inference task; 2) We study the influence of the hyperparameter on the performance of our Hypergraph-MLP; and 3) We study the impact of homophily levels in the dataset on the efficacy of our Hypergraph-MLP.

a model named hypergraph Markov random fields (HMRF) borrowing the idea from Gaussian Markov random fields [13]. Furthermore, we demonstrate that the HMRF motivates effective solutions through constraining the optimisation solution space and evoking new architecture for multiple hypergraph machine learning tasks.

The proposed HMRF model is rooted in the assumption that the features of nodes in a hyperedge are highly correlated by the features of the hyperedge connecting them (See fig. 1a for the illustration of a hypergraph obeying this assumption). Under such an assumption, the data-generating process of a hypergraph is modelled as a multivariate Gaussian distribution whose energy function is uniquely determined by the given hypergraph structure and the joint node-hyperedge features (see fig. 1b for a visualisation). According to previous research [11], [12], hyperedge features are typically absent in real-world applications. Therefore, to enhance the applicability of the proposed HMRF, we approximate the energy function with an estimator that can be computed without hyperedge features, which is defined as the sum of the maximum l_2 distances between any two node features in each hyperedge of a hypergraph.

To demonstrate the value of our proposed hypergraph Markov random field in practical applications, we develop novel frameworks for two crucial downstream tasks, *hypergraph structure inference* and *node classification*. Within both of the tasks, the energy estimator can be easily incorporated into the optimisation objective to constrain the solution. Specifically, the hypergraph structure inference task aims to infer hypergraph structures when they are unavailable in the application domains. To this task, we introduce an original framework named HGSI. Utilising observed node features as inputs, this framework infers probabilities of potential hyperedges by solving a maximum a-posteriori problem whose data likelihood term is evaluated by the energy estimator. Compared with previous hypergraph structure inference methods, the key novelty of our proposed approach lies in its capability to directly learn probabilities for potential hyperedges, eliminating the reliance on pre-defined hypergraphs for training supervision. The goal of hypergraph node classification is to classify nodes on a pre-defined hypergraph structure. For this task, we design a novel learning framework called Hypergraph-MLP. This framework consists of two primary components: a multi-layer perceptron (MLP) and a loss function rooted in the energy estimator which regularises the embedding optimisation through the prior belief adhering to the data-generating process. Training with the energy-based loss function enables the MLP to leverage structural information efficiently without message passing. This leads to lower inference latency in Hypergraph-MLP compared to traditional message-passing hypergraph neural networks. Moreover, removing the reliance on the hypergraph structure at inference makes Hypergraph-MLP robust to structural perturbations.

We conduct extensive experiments to show that: 1) On both synthetic and real-world datasets, the proposed HGSI significantly outperforms existing hypergraph structure inference methods; and 2) On seven hypergraph node classification benchmarks, compared to existing hypergraph neural networks, Hypergraph-MLP achieves competitive accuracy, fastest inference, and better robustness against structural perturbations at inference.

The contributions of this work are as follows:

- We generalise Markov random field to model the data-generating process on hypergraphs, yielding a model named hypergraph Markov random field. We extend the applicability of the HMRF to the scenarios when hyperedge features are unobservable through an energy estimator.
- We develop an original unsupervised approach for hypergraph structure inference using the proposed HMRF. The unsupervised nature of our method enables it to be applied to scenarios where pre-defined hypergraphs are unavailable.
- We design Hypergraph-MLP, a novel learning framework for learning node embeddings with pre-defined hypergraph structures based on the proposed HMRF. It provides a new paradigm for designing neural networks to process hypergraph-structured data deviating from the traditional message-passing frameworks.
- We carry out extensive experiments on hypergraph structure inference and hypergraph node classification tasks. The results confirm the practical value of our proposed HMRF in processing data with higher-order interactions.

The rest of this paper is structured as follows. In section II, we introduce the necessary notations and formalise the hypergraph machine learning tasks from a probabilistic perspective. In section III, we propose a novel hypergraph Markov random field and leverage this model to develop an energy estimator that could be utilised to construct feature likelihood in hypergraph structure inference task and embedding prior in hypergraph node classification tasks. In section IV, we apply the energy estimator to design an original framework for hypergraph structure inference. In section V, we use the energy estimator to design a novel framework for classifying nodes on pre-defined hypergraphs. In section VI, we review related works. In section VII, we conduct experiments on hypergraph structure inference and hypergraph node classification. Finally, we conclude this paper in section VIII.

II. Notation

In this section, we introduce the notations for hypergraphs, and the features of nodes and hyperedges.

Hypergraphs. A hypergraph can be represented as $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \boldsymbol{H}\}$. Here, $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ 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 $\boldsymbol{H} = [w_1\boldsymbol{h}_1, w_2\boldsymbol{h}_2, \cdots, w_m\boldsymbol{h}_m] \in \mathbb{R}^{n\times m}$ is an incidence matrix embedding a hypergraph structure, in which $w_j \in [0,1]$ is the weight of the j-th hyperedge and $\boldsymbol{h}_j \in \{0,1\}^n$ represents the j-th hyperedge: $\boldsymbol{H}_{ij}=1$ indicates that hyperedge j contains node i and $\boldsymbol{H}_{ij}=0$ otherwise. Usually, the hyperedge weight is set to 1 in the real-world applications. The size of a hyperedge is the number of nodes contained in a hyperedge.

Node and hyperedge features. For a hypergraph $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \boldsymbol{H}\}$, let $\boldsymbol{X}_{\mathcal{V}} = [\boldsymbol{x}_{v_1}, \boldsymbol{x}_{v_2}, \cdots, \boldsymbol{x}_{v_n}]^T \in \mathbb{R}^{n\times d}$ denote the node features and $\boldsymbol{X}_{\mathcal{E}} = [\boldsymbol{x}_{e_1}, \boldsymbol{x}_{e_2}, \cdots, \boldsymbol{x}_{e_m}]^T \in \mathbb{R}^{m\times d}$ denote the hyperedge features, which are two matrices that contain d-dimensional features. Notably, when $d = 1$, these

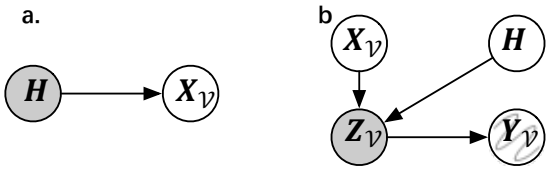


Fig. 2: The graphical models for (a) the hypergraph structure inference and (b) hypergraph node classification. The shaded variables are unobserved and the curve-crossed variables are partially observed.

feature matrices reduce to feature vectors. To keep the notation concise, we focus our discussion on the case with $d = 1$. However, our results are directly applicable to cases with $d > 1$.

Hypergraph Machine Learning and Data-generating Process. The design of the hypergraph ML algorithm is highly related to the underlying data-generating process. We will illustrate the relationship through two tasks: *hypergraph structure inference* and *hypergraph node classification*. This section only introduces the general formalisation and the detailed solution is left in section IV and section V.

Specifically, the *hypergraph structure inference* problem aims at estimating the ground truth incidence matrix through the node features, which can be formalized as a maximum a-posteriori (MAP) estimation problem:

$$\begin{aligned}\hat{\boldsymbol{H}} &= \arg\max_{\boldsymbol{H}} P(\boldsymbol{H} \mid \boldsymbol{X}_\mathcal{V}) \\ &= \arg\max_{\boldsymbol{H}} \underbrace{\log P(\boldsymbol{X}_\mathcal{V} \mid \boldsymbol{H})}_{\text{Feature likelihood given structure}} + \underbrace{\log P(\boldsymbol{H})}_{\text{Prior on structure}}\end{aligned} \tag{1}$$

where the knowledge of the data-generating process would inform the analytical form of likelihood term $P(\boldsymbol{X}_\mathcal{V} \mid \boldsymbol{H})$. In section IV we will further introduce how to turn the MAP problem into a meaningful optimisation problem.

Meanwhile, the *node classification* task on hypergraph aims to find the optimal class estimation for the unlabelled nodes. Let $\boldsymbol{X}_\mathcal{V} \in \mathbb{R}^{n\times d}$ denote the observed node features, $\mathcal{V}_{lab}$ be the set of nodes with observed labels $\boldsymbol{Y}_{lab} = \{\mathbf{y}_v\}_{v\in\mathcal{V}_{lab}}$, where $\mathbf{y}_{v_i} \in \{0,1\}^c$ be a one-hot label, and $\mathcal{V}_{un} = \mathcal{V}\backslash\mathcal{V}_{lab}$ be a set of unlabeled nodes. The hypergraph node classification task requires a model to classify nodes within $\mathcal{V}_{un}$ based on $\boldsymbol{X}_\mathcal{V}$, known labels $\boldsymbol{Y}_{lab}$, and the given hypergraph structure $\boldsymbol{H}$. In general, it has two steps: 1) learning the model parameter Θ with the objective:

$$\begin{aligned}\Theta^* &= \arg\max_{\Theta} P(\Theta \mid \boldsymbol{Y}_{lab}, \boldsymbol{X}_\mathcal{V}, \boldsymbol{H}) \\ &= \arg\max_{\Theta} \underbrace{\log P(\boldsymbol{Y}_{lab} \mid \Theta, \boldsymbol{X}_\mathcal{V}, \boldsymbol{H})}_{\text{log-likelihood}} + \underbrace{\log P(\Theta \mid \boldsymbol{X}_\mathcal{V}, \boldsymbol{H})}_{\text{log-prior}}\end{aligned} \tag{2}$$

and 2) prediction based on the optimised model:

$$\hat{\boldsymbol{Y}}_{un} = \arg\max_{\boldsymbol{Y}_{un}} P(\boldsymbol{Y}_{un} \mid \boldsymbol{X}_\mathcal{V}, \boldsymbol{H}, \Theta^*). \tag{3}$$

If we consider the model parameter to be the node embeddings denoted as $\boldsymbol{Z}_\mathcal{V}$ and consider the graphical model as in fig. 2b, the training objective in eq. (2) becomes,

$$\underbrace{\log P(\boldsymbol{Y}_{lab} \mid \boldsymbol{Z}_\mathcal{V})}_{\text{log-likelihood}} + \underbrace{\log P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{H})}_{\text{prior from the structure}} + \underbrace{\log P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{X}_\mathcal{V})}_{\text{Prior for the features}}. \tag{4}$$

Similarly, the DGP can guide the training through impacting the term $P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{H})$. In section V, we will first introduce how we design the hypergraph DGP that helps in parmaterise $P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{H})$ and then dive into how such method can lift the hypergraph machine learning algorithms.

III. Methodology

In this section, we first introduce how we utilise Markov random field for modelling the generation process of hypergraph data. Specifically, the data-generating process of a hypergraph can be formalised as,

$$\underbrace{\log P(\boldsymbol{Y}_{lab} \mid \boldsymbol{Z}_\mathcal{V})}_{\text{log-likelihood}} + \underbrace{\log P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{H})}_{\text{prior from the structure}} + \underbrace{\log P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{X}_\mathcal{V})}_{\text{Prior for the features}}. \tag{5}$$

Then, we develop an energy estimator for the DGP that helps in tackling the challenge of missing hyperedge features. We further leverage the energy estimator to establish both the likelihood function in structure inference task and the prior in node classification task.

A. Hypergraph Markov Random Field

In this subsection, we first introduce the fundamental assumption of our hypergraph Markov random field. Building upon this assumption, we represent the relationship between nodes and hyperedges in the feature space through an incidence graph. Finally, leveraging this graph, we present the probabilistic formulation of our hypergraph Markov random field[1]. This formulation captures the distribution of node features and hyperedge features within a hypergraph as a multivariate Gaussian distribution.

Assumption. To characterise the criteria by which nodes with certain features can be connected by a hyperedge, we presume that: *the features of nodes in a hyperedge are highly correlated by the features of the hyperedge connecting them.* One real-world example obeying this assumption is the academic hypergraph, where nodes are authors, hyperedges are papers, node features are authors' research interests, and hyperedge features are paper topics. In this hypergraph, co-authors (nodes) tend to share similar research interests (node features) aligned with their paper's topic (hyperedge features); See fig. 1 (panel a) for an illustration of this example.

Incidence graph. Our assumption indicates that, in the feature space, nodes are close to each other and also to their corresponding hyperedges. In this case, we capture the relation between nodes and hyperedges by using an incidence graph [14]–[16] corresponding to the hypergraph structure $\boldsymbol{H}$.

Specifically, for the hypergraph $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \boldsymbol{H}\}$, we construct a unique incidence graph that is a bipartite graph $\mathcal{G} = \{\mathcal{V} \cup \mathcal{V}', \mathcal{E}_\mathcal{G}, \boldsymbol{L}_\mathcal{H}\}$, where $v_i \in \mathcal{V}$ is a node in $\mathcal{H}$, $v_{e_j} \in \mathcal{V}'$ corresponds to the hyperedge e_j in $\mathcal{H}$, and there is an edge between v_i and v_{e_j} if and only if e_j contains v_i in $\mathcal{H}$. The structure of $\mathcal{G}$ can be represented as a graph Laplacian matrix:

$$\boldsymbol{L}_\mathcal{H} = \begin{bmatrix} \operatorname{diag}(\boldsymbol{H}\mathbf{1}_m) & -\boldsymbol{H} \\ -\boldsymbol{H}^T & \operatorname{diag}(\boldsymbol{H}^T\mathbf{1}_n) \end{bmatrix}, \tag{6}$$

[1]The model is "Markovian" in the sense that the nodes/hyperedges are conditional independent of the other nodes/hyperedges given their directly connected components.

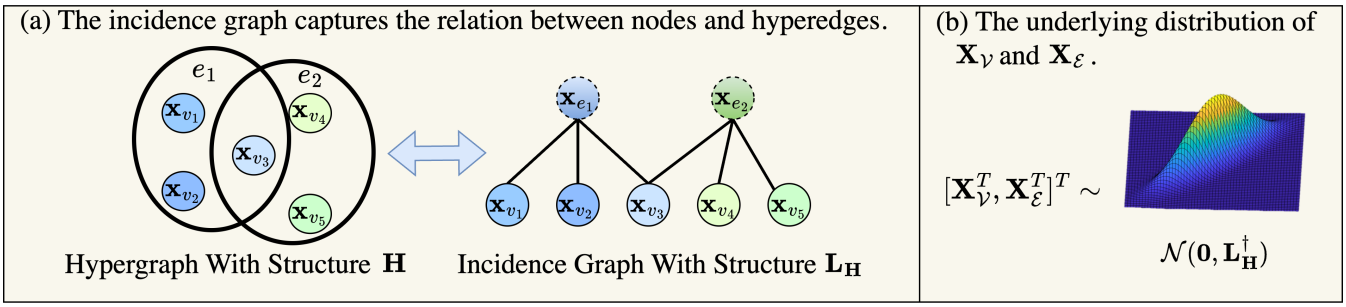


Fig. 3: $\boldsymbol{X}_{\mathcal{V}} = [\boldsymbol{x}_{v_1}, \boldsymbol{x}_{v_2}, \cdots, \boldsymbol{x}_{v_n}]^T \in \mathbb{R}^{n\times d}$ and $\boldsymbol{X}_{\mathcal{E}} = [\boldsymbol{x}_{e_1}, \boldsymbol{x}_{e_2}, \cdots, \boldsymbol{x}_{e_m}]^T \in \mathbb{R}^{m\times d}$ denote node and hyperedge features respectively. (a) For smooth data on a hypergraph $\boldsymbol{H}$, the relationship between $\boldsymbol{X}_{\mathcal{V}}$ and $\boldsymbol{X}_{\mathcal{E}}$ can be modelled by an incidence graph corresponding to $\boldsymbol{H}$ with the graph Laplacian $\boldsymbol{L}_{\mathcal{H}}$. (b) With the incidence graph, $\boldsymbol{X}_{\mathcal{V}}$ and $\boldsymbol{X}_{\mathcal{E}}$ can be associated with a multivariate Gaussian distribution whose covariance matrix is the pseudoinverse of $\boldsymbol{L}_{\mathcal{H}}$.

where $\boldsymbol{L}_{\mathcal{H}} \in \mathbb{R}^{(n+m)\times(n+m)}$, $\mathrm{diag}(\cdot)$ is a mapping that converts a vector into a diagonal matrix, and $\mathbf{1}_n \in \{1\}^n$ and $\mathbf{1}_m \in \{1\}^m$ are two all-one vectors. Further, the features of nodes in $\mathcal{V}$ are $\boldsymbol{X}_{\mathcal{V}}$, and the features of hyperedges in $\mathcal{V}'$ are $\boldsymbol{X}_{\mathcal{E}}$. fig. 3 (panel a) shows an example of the incidence graph.

Probabilistic formulation. We model the conditional distribution of node and hyperdge features $\boldsymbol{X}_{\mathcal{H}} = [\boldsymbol{X}_{\mathcal{V}}^T, \boldsymbol{X}_{\mathcal{E}}^T]^T \in \mathbb{R}^{(n+m)\times d}$ given incidence graph $\boldsymbol{H}$, $P(\boldsymbol{X}_{\mathcal{H}} \mid \boldsymbol{H})$, following the literature of graph signal processing [17]–[20]. Specifically, under the assumption of smoothness on a hypergraph, the distribution of node features can be modelled by a multivariate Gaussian distribution whose covariance matrix is the pseudoinverse of the corresponding graph Laplacian matrix. In our incidence graph, nodes are connected to their corresponding hyperedge and by our assumption they all have similar features. Therefore the conditional distribution of $P(\boldsymbol{X}_{\mathcal{H}} \mid \boldsymbol{H})$ can be modelled as:

$$\boldsymbol{X}_{\mathcal{H}} = [\boldsymbol{X}_{\mathcal{V}}^T, \boldsymbol{X}_{\mathcal{E}}^T]^T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{L}_{\mathcal{H}}^{\dagger}), \tag{7}$$

where $\boldsymbol{L}_{\mathcal{H}}^{\dagger}$ is the pseudoinverse of $\boldsymbol{L}_{\mathcal{H}}$. As $\boldsymbol{L}_{\mathcal{H}}$ is uniquely associated with $\boldsymbol{H}$, the relationship between $\boldsymbol{X}_{\mathcal{V}}$ and $\boldsymbol{H}$ is captured byeq. (7). The probabilistic formulation is summarised in fig. 3 (panel b). We name the model introduced in eq. (7)) as the hypergraph Markov random field analogising to the concept of Markov networks [13] used to model the graph generation process [13], [21]. Notably, the energy function in this model is captured by the quadratic form on the hypergraph features with respective of incidence matrices,

$$f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}}) = \boldsymbol{X}_{\mathcal{H}}^{\top} \boldsymbol{L}_{\mathcal{H}} \boldsymbol{X}_{\mathcal{H}} \tag{8}$$

The likelihood can be defined through the energy function,

$$P(\boldsymbol{X}_{\mathcal{H}} \mid \boldsymbol{H}) \propto \exp\left(-f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}})\right). \tag{9}$$

B. Energy Estimator

Based on prior research [11], [12], hyperedge features are generally not present in practical applications. Consequently, to improve the applicability of the proposed hypergraph Markov random field, we develop an energy estimator that serves as a lower bound of the energy function in the optimisation stage. The estimator is capable of establishing the node feature likelihood function given the incidence matrix of a hypergraph structure inference problem and the prior function of the node embeddings given a hypergraph structure in the node classification task.

To derive the estimator of the energy function and eliminate the dependency on hyperedge features, we expand eq. (8) as:

$$f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}}) = \sum_{i=1}^{m_p} \sum_{v_j \in e_i} w_i \, \|\boldsymbol{x}_{e_i} - \boldsymbol{x}_{v_j}\|_2^2 = \boldsymbol{w}^T \boldsymbol{s}_{\boldsymbol{H}}, \tag{10}$$

where $\boldsymbol{w} = [w_1, w_2, \cdots, w_m]^T \in [0,1]^m$ is the vectorised weights, $\boldsymbol{s}_{\boldsymbol{H}} = [s_{e_1}, s_{e_2}, \cdots, s_{e_m}]^T \in \mathbb{R}^m$, and $s_{e_i} = \sum_{v_j \in e_i} \|\boldsymbol{x}_{e_i} - \boldsymbol{x}_{v_j}\|_2^2$. The key challenge of directly using $f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}})$ to estimate $P(\boldsymbol{X}_{\mathcal{H}} \mid \boldsymbol{H})$ is that $\boldsymbol{X}_{\mathcal{E}}$ is usually implicit in real-world applications. Therefore, we further design our energy estimator as:

$$g(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{V}}) = \sum_{e_i \in \mathcal{E}} w_i \max_{v_j, v_k \in e_i} (\|\boldsymbol{x}_{v_j} - \boldsymbol{x}_{v_k}\|_2) = \mathbf{w}^T \boldsymbol{s}'_{\boldsymbol{H}}, \tag{11}$$

where $\boldsymbol{s}'_{\boldsymbol{H}} = [s'_{e_1}, s'_{e_2}, \cdots, s'_{e_m}]^T \in \mathbb{R}^m$, and $s'_{e_i} = \max_{v_j, v_k \in e_i} (\|\boldsymbol{x}_{v_j} - \boldsymbol{x}_{v_k}\|_2^2)$ is the largest squared ℓ_2 distance between any two nodes in e_i. We prove that $g(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{V}})$ is a lower bound for $f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}})$ as follows:

Theorem 1. *For any hypergraph structure $\boldsymbol{H} \in \{0,1\}^{n\times m}$, given node features $\boldsymbol{X}_{\mathcal{V}} = [\boldsymbol{x}_{v_1}, \boldsymbol{x}_{v_2}, \cdots, \boldsymbol{x}_{v_n}]^T \in \mathbb{R}^{n\times d}$, and hyperedge features $\boldsymbol{X}_{\mathcal{E}} = [\boldsymbol{x}_{e_1}, \boldsymbol{x}_{e_2}, \cdots, \boldsymbol{x}_{e_m}]^T \in \mathbb{R}^{m\times d}$, $g(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{V}})$ is a lower bound for $f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}})$.*

Proof. To prove $g(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{V}})$ is a lower bound for $f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}})$, it suffices to prove, for any hyperedge e_i in $\boldsymbol{H}$, the following inequality holds:

$$\sum_{v_j \in e_i} \|\boldsymbol{x}_{e_i} - \boldsymbol{x}_{v_j}\|_2 \geq \max_{v_j, v_k \in e_i} (\|\boldsymbol{x}_{v_j} - \boldsymbol{x}_{v_k}\|_2). \tag{12}$$

Without the loss of generality, in the feature space, let v_a and v_b be the two most distant nodes within e_i. By triangle inequality [22], we have:

$$\begin{aligned} \sum_{v_j \in e_i} \|\boldsymbol{x}_{e_i} - \boldsymbol{x}_{v_j}\|_2 &\geq \|\boldsymbol{x}_{e_i} - \boldsymbol{x}_{v_a}\|_2 + \|\boldsymbol{x}_{e_i} - \boldsymbol{x}_{v_b}\|_2 \\ &\geq \|\boldsymbol{x}_{v_a} - \boldsymbol{x}_{v_b}\|_2. \end{aligned}$$

Hence, eq. (12) holds, which proves the theorem. □

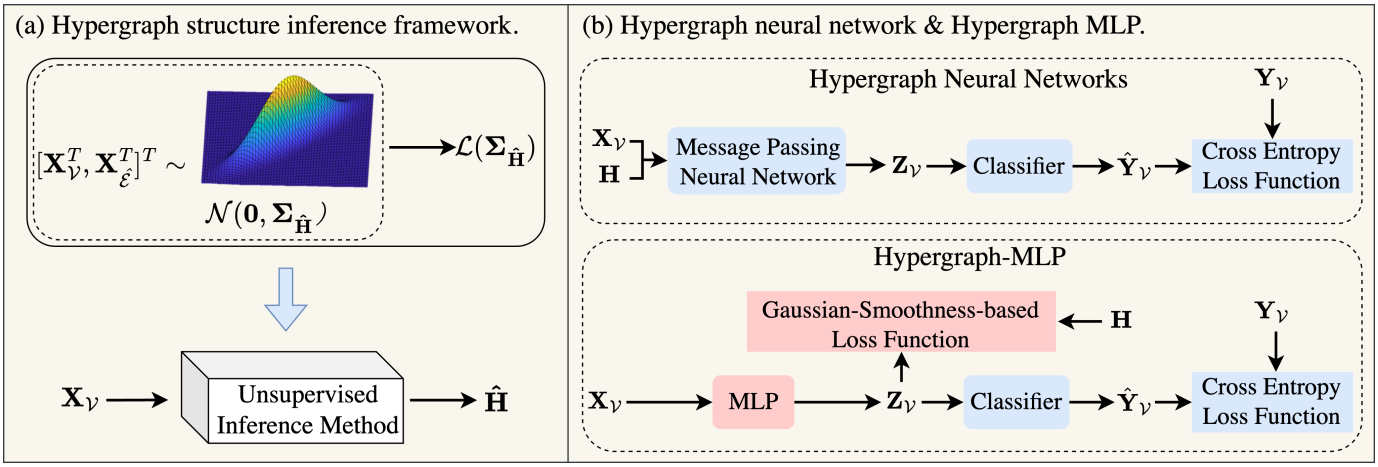


Fig. 4: (a) The proposed unsupervised inference method, which infers the incidence matrix of a potential hypergraph structure $\hat{\boldsymbol{H}}$ from given node features $\boldsymbol{X}_{\mathcal{V}}$, is based on minimising the negative log-likelihood $\mathcal{L}(\mathbf{\Sigma}_{\hat{\boldsymbol{H}}})$. $\boldsymbol{X}_{\hat{\mathcal{E}}}$ denotes the features of potential hyperedges in $\hat{\boldsymbol{H}}$. (b) The training paradigm of the Hypergraph-MLP and hypergraph neural network for hypergraph node classification, where $\boldsymbol{X}_{\mathcal{V}}$ denotes node features, $\boldsymbol{H}$ is a pre-defined incidence matrix, and $\boldsymbol{Y}_{\mathcal{V}}$ denotes node labels.

Feature Likelihood for Hypergraph Structure learning. The optimisation objective of hypergraph structure inference in eq. (1) contains the term maximising the feature likelihood $P(\boldsymbol{X}_{\mathcal{V}} \mid \boldsymbol{H})$. Such an objective can be reparameterise as optimising over the energy estimator as follows,

$$\begin{aligned} \arg\max \log P(\boldsymbol{X}_{\mathcal{V}} \mid \boldsymbol{H}) &= \arg\min f(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{H}}) \\ &\rightarrow \arg\min g(\boldsymbol{H}, \boldsymbol{X}_{\mathcal{V}}). \end{aligned} \tag{13}$$

In section IV, we will show how we solve the hypergraph structure inference task through incorporating our energy estimator into eq. (1).

Embedding Prior for Hypergraph Node Classification. As introduced in eq. (4), we aim at injecting the inductive bias when designing the embedding prior $P(\boldsymbol{Z}_{\mathcal{V}} \mid \boldsymbol{H})$. With the data-generating process introduced in eq. (7), we are able to explicitly constraint the embedding to follow a similar constraint as we did in HGSI. Specifically, we design the probability density function of $P(\boldsymbol{Z}_{\mathcal{V}} \mid \boldsymbol{H})$ following that,

$$\begin{aligned} &P(\boldsymbol{Z}_{\mathcal{H}}|\boldsymbol{H}) \propto \exp\big(-f(\boldsymbol{H}, \boldsymbol{Z}_{\mathcal{H}})\big), \\ &\text{with } f(\boldsymbol{H}, \boldsymbol{Z}_{\mathcal{H}}) = \boldsymbol{Z}_{\boldsymbol{H}}^{\top} \boldsymbol{L}_{\mathcal{H}} \boldsymbol{Z}_{\boldsymbol{H}}. \end{aligned} \tag{14}$$

As discussed before, $\boldsymbol{Z}_{\boldsymbol{H}}$ usually does not exist in real-world applications. Therefore, based on theorem 1, we further replace the energy function $f(\boldsymbol{H}, \boldsymbol{Z}_{\mathcal{H}})$ with $g(\boldsymbol{H}, \boldsymbol{Z}_{\mathcal{V}}) = \sum_{e_i \in \mathcal{E}} w_i \max_{v_j, v_k \in e_i} (\|\boldsymbol{z}_{v_j} - \boldsymbol{z}_{v_k}\|_2)$ defined in eq. (11). In section V, we demonstrate that such an embedding prior can lead to an efficient and accurate algorithm in the hypergraph node classification task.

IV. Hypergraph Structure Inference

In this section, we start by introducing the problem formulation of the hypergraph structure inference task. Then, we leverage the hypergraph Markov random field introduced in section III-B to design a novel hypergraph structure inference framework entitled HGSI. See fig. 4 (panel a) for the illustration of this hypergraph structure inference framework.

A. Problem Formulation

The hypergraph structure inference task aims to take the observed node features $\boldsymbol{X}_{\mathcal{V}} \in \mathbb{R}^{n \times d}$ as inputs to infer a ground-truth binary incidence matrix $\boldsymbol{H}_{gt} \in \{0,1\}^{n \times m}$ for the hypergraph structure associated with $\boldsymbol{X}_{\mathcal{V}}$. We assume $\boldsymbol{H}_{gt}$ is drawn from a distribution that can be characterised by a weighted incidence matrix $\hat{\boldsymbol{H}} = [w_1 \hat{\mathbf{h}}_1, w_2 \hat{\mathbf{h}}_2, \cdots, w_{m_p} \hat{\mathbf{h}}_{m_p}] \in \mathbb{R}^{n \times m_p}$, where $\hat{\mathbf{h}}_i \in \{0,1\}^n$ is a potential hyperedge, m_p is the number of potential hyperedges, and $w_i \in [0,1]$ defines the probability of the existence of $\hat{\mathbf{h}}_i$ in $\boldsymbol{H}_{gt}$. As $\hat{\mathbf{h}}_i$ is pre-defined, we aim to infer a vector $\boldsymbol{w} = [w_1, w_2, \cdots, w_{m_p}]^T \in [0,1]^{m_p}$ that contains probabilities of all potential hyperedges taking node features $\boldsymbol{X}_{\mathcal{V}}$ as inputs, and we do so without training on known ground-truth hyperedges. After $\boldsymbol{w}$ is inferred, we construct $\boldsymbol{H}_{gt}$ by m most likely potential hyperedges. Without any constraints, $\hat{\boldsymbol{H}}$ would contain all the combinations of n nodes as potential hyperedges, and in this case $m_p = 2^n$.

B. Hypergraph Structure Inference through the Data-generating Process

Unsupervised inference method. Inferring $\boldsymbol{H}_{gt}$ from $\boldsymbol{X}_{\mathcal{V}}$ is to infer $\boldsymbol{w}$ from $\boldsymbol{X}_{\mathcal{V}}$. Substituting eq. (13) into eq. (1), we can formulate the inference as the following optimisation problem with proper prior:

$$\min_{\boldsymbol{w}} \ \boldsymbol{w}^T \boldsymbol{s}'_{\hat{\boldsymbol{H}}} - \alpha \mathbf{1}_{m_p}^T \log(\boldsymbol{w}) + \beta \|\boldsymbol{w}\|_1 \quad \text{s.t.}\, w_i \in (0,1]. \tag{15}$$

where $\mathbf{1}_{m_p} \in \{1\}^{m_p}$ is an all-one vector, and α and β are two hyperparameters. Here the first term is the energy estimator for the hypergraph Markov random field as defined in eq. (11). The second term in eq. (15) is a log barrier that enforces the positivity of the learned probabilities, which can prevent w_i from being zero and thereby avoid the creation of an empty hypergraph structure. The third term is used to ensure the sparsity of the target hypergraph structure, i.e., we hope that only a small number of potential hyperedges have significant probabilities.

To solve the optimisation problem, we take the derivative of the objective function with respect to each $w_i \in (0, 1]$:

$$\frac{\partial f_{wv}}{\partial w_i} = s'_{\hat{e}_i} - \frac{\alpha}{w_i} + \beta,$$

and set it to zero:

$$w_i^\star = \frac{1}{\alpha s'_{\hat{e}_i} + \beta}, \tag{16}$$

$$\frac{\partial^2 f_{wv}(w_i^\star, \boldsymbol{X}_\mathcal{V})}{\partial^2 w_i^\star} = \frac{1}{w_i^{\star 2}} > 0,$$

hence, $\boldsymbol{w}^\star = [w_1^\star, w_2^\star, \cdots, w_{m_p}^\star]^T \in (0, 1]^{m_p}$ is the analytical solution for eq. (15) under the constraints each $w_i \in (0, 1]$. In practice, we use eq. (16) to infer the probabilities for potential hyperedges. Additionally, to guarantee that $w_i \in (0, 1]$, we set $\alpha = 1$ and $\beta = 1$. Solving eq. (15) does not require labelled data, so the proposed inference method is completely unsupervised.

Hypergraph structure construction. Notably, without any constraints, $m_p = 2^n$ which makes solving eq. (15) extremely time-consuming. To address this issue, we propose to constrain the set of potential hyperedges in $\hat{\boldsymbol{H}}$ in a way similar to that in [17]: *for a given list $\mathcal{K}_S = [k_1, k_2, \cdots, k_S]$ that collects S desired hyperedge sizes in descending order, each potential k_s-hyperedge is formed by a node with its $k_s - 1$ nearest neighbours in the feature space.* By doing so, we constrain the number of potential hyperedges always not greater than Ln, and ensure that all the potential hyperedges consist of nodes with similar features. We then use this constrained set to solve eq. (15) based on Algorithm. 16. Finally, we construct $\boldsymbol{H}_{gt}$ with $\boldsymbol{w}^\star$. Details of our approach are in Algorithm. 1.

Algorithm 1: Hypergraph Structure Inference Under HMRF (**HGSI**)

Input: $\boldsymbol{X}_\mathcal{V}$, $\mathcal{K}_S$, m, and $\mathcal{M}_S$ a list with the number of target hyperedges in each size (optional).
Output: Binary incidence matrix $\boldsymbol{H}_{gt}$.
Initialisation: An empty set used to save inferred hyperedges $\mathcal{S}$.
if *$\mathcal{M}_S$ is given* **then**
  for $i = S : 1$ **do**
    For k_i in $\mathcal{K}_S$, generate potential k_i-hyperedges, which are formed by nodes with their $k_i - 1$ nearest neighbours in the feature space.
    Compute $\boldsymbol{s}'_{\hat{\boldsymbol{H}}}$ by $\boldsymbol{X}_\mathcal{V}$ for the generated k_i-hyperedges, and use $\boldsymbol{s}'_{\hat{\boldsymbol{H}}}$ to generate $\boldsymbol{w}^\star$ based on eq. (16).
    Delete the generated k_i-hyperedges that are subsets of hyperedges in $\mathcal{S}$, and add top-$\mathcal{M}_i$ hyperedges in the remaining generated k_i-hyperedges to $\mathcal{S}$ based on $\boldsymbol{w}^\star$.
  end
  Form $\boldsymbol{H}_{gt}$ with hyperedges in $\mathcal{S}$.
end
else
  for $i = 1 : S$ **do**
    For k_i in $\mathcal{K}_S$, introduce potential k_i-hyperedges to $\mathcal{S}$, which are formed by nodes with their $k_i - 1$ nearest neighbours in the feature space.
  end
  Compute $\boldsymbol{s}'_{\hat{\boldsymbol{H}}}$ by $\boldsymbol{X}_\mathcal{V}$ for the potential hyperedges in $\mathcal{S}$, and use $\boldsymbol{s}'_{\hat{\boldsymbol{H}}}$ to generate $\boldsymbol{w}^\star$ based on eq. (16).
  Form $\boldsymbol{H}_{gt}$ with top-m potential hyperedges in $\mathcal{S}$ based on $\boldsymbol{w}^\star$.
end

V. Hypergraph Node Classification

In this section, we first present the problem formulation of the hypergraph node classification task. Then, we demonstrate a solution inspired by the hypergraph Markov random field for this task and propose a learning framework named Hypergraph-MLP.

A. Problem Formulation

Let $\boldsymbol{X}_\mathcal{V} \in \mathbb{R}^{n \times d}$ denote the observed node features, $\mathcal{V}_{lab}$ be the set of labelled nodes with ground truth labels $\boldsymbol{Y}_{lab} = \{\mathbf{y}_v\}_{v \in \mathcal{V}_{lab}}$, where $\mathbf{y}_{v_i} \in \{0, 1\}^c$ be a one-hot label, and $\mathcal{V}_{un} = \mathcal{V} \backslash \mathcal{V}_{lab}$ be a set of unlabeled nodes. The hypergraph node classification task requires a model to classify nodes within $\mathcal{V}_{un}$ based on $\boldsymbol{X}_\mathcal{V}$, known labels $\boldsymbol{Y}_{lab}$, and the given hypergraph structure $\boldsymbol{H}$.

B. Hypergraph-MLP informed by Data-generating Process

In this section, we utilize the hypergraph Markov random field and the related energy estimator introduced in section III-B to design a framework, named Hypergraph-MLP, for classifying nodes on hypergraphs. We start by introducing the model architecture used in our Hypergraph-MLP. Then, we elaborate on how to train the MLP-based model to use the structural information from $\boldsymbol{H}$ without requiring it during inference. Finally, we compare Hypergraph-MLP with existing hypergraph neural networks. To make the following discussion accessible, we denote the node embeddings in the layer l of the MLP-based model as $\mathbf{Z}_\mathcal{V}^{(l)} = [\mathbf{z}_{v_1}^{(l)}, \mathbf{z}_{v_2}^{(l)}, \cdots, \mathbf{z}_{v_n}^{(l)}]^T \in \mathbb{R}^{n \times d}$. Following previous papers [11], [12], [23], [24], we initialize $\mathbf{Z}_\mathcal{V}^{(0)}$ with $\boldsymbol{X}_\mathcal{V}$.

Model architecture. The training objective in eq. (4) suggests that the model design should consist of three components: 1) a predictor (classifier) that models $P(\boldsymbol{Y}_{lab} \mid \boldsymbol{Z}_\mathcal{V})$, 2) the prior to encoding hypergraph structure $P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{H})$ and 3) the prior mapping between node features and embedding $P(\boldsymbol{Z}_\mathcal{V} \mid \boldsymbol{X}_\mathcal{V})$. The whole architecture is visualized in fig. 4. Our model first utilises a multilayer perceptron (MLP) for the node embedding generation [25], [26] from node features, which has the form,

$$\mathbf{Z}_\mathcal{V}^{(l)} = D(LN(\sigma(\mathbf{Z}_\mathcal{V}^{(l-1)} \boldsymbol{\Theta}^{(l)}))), \tag{17}$$

where $D(\cdot)$ is the dropout function, $LN(\cdot)$ is the layer normalization function, $\sigma(\cdot)$ is the activation function, and $\boldsymbol{\Theta}^{(l)} \in \mathbb{R}^{d \times d}$ denotes the learnable parameters for layer l. We then develop a nodes classifier on hypergraphs that generate the node label logits formulated as:

$$\hat{\boldsymbol{Y}}_\mathcal{V} = softmax(\mathbf{Z}_\mathcal{V}^{(L)} \boldsymbol{W}), \tag{18}$$

Algorithm 2: Hypergraph-MLP for Classifying Nodes on Hypergraphs

/*Training*/
Input: $\boldsymbol{X}_{\mathcal{V}}$, $\boldsymbol{Y}_{\mathcal{V}}$, $\boldsymbol{H}$, $\mathcal{V}'$, the number of layers L, the number of the training iterations T.
Output: The optimal model parameters $[\boldsymbol{\Theta}^{(1)^\star}, \cdots, \boldsymbol{\Theta}^{(L)^\star}, \boldsymbol{W}^\star]$.
Initialisation: Randomly initialise the model parameters $[\boldsymbol{\Theta}^{(1)}, \cdots, \boldsymbol{\Theta}^{(L)}, \boldsymbol{W}]$, and set $\mathbf{Z}_{\mathcal{V}}^{(0)}$ as $\boldsymbol{X}_{\mathcal{V}}$.
for $t = 1 : T$ **do**
  for $l = 1 : L$ **do**
    Update the node embeddings by eq. (17).
  end
  Generate the node label logits $\hat{\boldsymbol{Y}}_{\mathcal{V}}$ by eq. (18).
  Update $[\boldsymbol{\Theta}^{(1)}, \cdots, \boldsymbol{\Theta}^{(L)}, \boldsymbol{W}]$ by minimizing eq. (21) with the gradient descent.
end
/*Inference*/
Input: $\boldsymbol{X}_{\mathcal{V}}$ and $[\boldsymbol{\Theta}^{(1)^\star}, \cdots, \boldsymbol{\Theta}^{(L)^\star}, \boldsymbol{W}^\star]$.
Output: Predicted node label logits $\hat{\boldsymbol{Y}}_{\mathcal{V}}$.
Initialisation: Set $\mathbf{Z}_{\mathcal{V}}^{(0)}$ as $\boldsymbol{X}_{\mathcal{V}}$.
for $l = 1 : L$ **do**
  Update the node embeddings by eq. (17).
end
Generate the label logits $\hat{\boldsymbol{Y}}_{\mathcal{V}}$ by eq. (18).

where $\mathbf{Z}_{\mathcal{V}}^{(L)} \in \mathbb{R}^{n\times d}$ denotes node embeddings generated by an L-layer Hypergraph-MLP, $\hat{\boldsymbol{Y}}_{\mathcal{V}} \in (0,1)^{n\times c}$ denotes the node label logits, $\boldsymbol{W} \in \mathbb{R}^{d\times c}$ denotes some learnable parameters, and $softmax(\cdot)$ is the softmax function. Finally, the optimisation objective also incorporated the energy estimator that is defined in eq. (11).

Training. Minimising the energy function ensures that the output embedding of an L-layer MLP, $\mathbf{Z}_{\mathcal{V}}^{(L)}$, follows the data generating process thus maintains a high probability as formulated eq. (7). In the mean time, it enables the MLP to perform inference using the structural information of the hypergraph without explicitly relying on a message passing module. To this end, we train the MLP to generate $\mathbf{Z}_{\mathcal{V}}^{(L)}$ that tends to minimise the energy function discussed in section III-B. Specifically, we design loss function based on the energy estimator $g(\boldsymbol{H}, \boldsymbol{Z}_{\mathcal{V}})$, as defined in eq. (11). Consequently, we formulate this loss function as:

$$\ell_{smooth} = \frac{1}{m}\sum_{i=1}^{m} w_i \max_{v_j, v_k \in e_i} (\|\mathbf{z}_{v_j}^{(L)} - \mathbf{z}_{v_k}^{(L)}\|_2^2). \tag{19}$$

Furthermore, following the previous literature [11], [12], we employ the cross-entropy loss function to train the overall architecture for node classification, which is defined as:

$$\ell_{CE} = -\frac{1}{|\mathcal{V}'|}\sum_{v_i \in \mathcal{V}'} \mathbf{y}_{v_i}^T \log(\hat{\mathbf{y}}_{v_i}), \tag{20}$$

where $\mathcal{V}'$ is a set containing selected training nodes, and $|\mathcal{V}'|$ is the number of the selected training nodes.

As a result, the overall loss function is defined as:

$$\ell_{overall} = \ell_{CE} + \alpha \ell_{smooth}. \tag{21}$$

where α is a coefficient to balance the two losses. In practice, our model is trained by using gradient descent to minimize the eq. (21). The overall framework of our proposed Hypergraph-MLP is summarized in **??** 2.

Compared with existing hypergraph neural networks. Our Hypergraph-MLP has two key benefits. Firstly, *inference speed.* The application of the message-passing-based hypergraph neural networks to real-world scenarios faces challenges due to high inference latency [6], [25], [26]. Let n be the number of nodes, m be the number of hyperedges, and L be the number of layers. The computational complexity of a hypergraph neural network is $\mathcal{O}(Ln + Lm)$, as it involves feature aggregation for every node and hyperedge in each layer. In contrast, the Hypergraph-MLP performs inference solely via feed-forward propagation, as formulated in eq. (17). Consequently, its computational complexity is $\mathcal{O}(Ln)$, which is significantly lower especially when dealing with datasets rich in hyperedges, such as DBLP as demonstrated in table V. Secondly, *inference robustness.* The significant dependence on the hypergraph structure for message passing renders current hypergraph neural networks vulnerable to structural perturbations at inference. For instance, the introduction of fake hyperedges during inference can lead well-trained hypergraph neural networks to generate baffling results [27], [28]. In contrast, Hypergraph-MLP implicitly takes into account the hypergraph structure, thus removing its dependence on the structure during inference. In section VII, we present empirical evidence to demonstrate that this property enhances the robustness of Hypergraph-MLP compared to existing hypergraph neural networks towards structural perturbations at inference. fig. 4 (panel b) compares the architectures of the Hypergraph-MLP and existing hypergraph neural networks.

VI. Related Work

A. Hypergraph Smoothness

The concept of smoothness plays a pivotal role in analyzing hypergraph-structured data that contain higher-order interactions. This concept, as defined in graph signal processing and machine learning [18]–[20], [29]–[31], posits that node features are smooth on a hypergraph structure when connected nodes have similar features. Current research on hypergraph smoothness mostly focuses on formulating metrics to quantify the level of smoothness present in node features on a given hypergraph [32]–[35], thereby enhancing the understanding of node feature patterns on hypergraphs. Nonetheless, there exists a notable gap in the exploration of the smoothness prior that aims to disclose the generation process of node features and hyperedge features on a hypergraph. To our knowledge, before this work, only one study [36] has attempted to investigate such a prior. The present study differs from [36] in two principal aspects. Firstly, there is a divergence in the formulation of the smoothness prior. In [36], the prior presumes that each hyperedge corresponds to a subgraph with both node feature and edge feature smoothness in the learnable graph structure. Conversely, this work defines the prior based on the assumption that the features of nodes in a hyperedge are highly correlated by the features of the hyperedge connecting them. Secondly, in this paper, our proposed

smoothness prior is formulated as a rigorous probabilistic model that mathematically describes the feature distribution of nodes and hyperedges on a hypergraph structure. In contrast, the smoothness prior presented in [36] lacks a well-defined mathematical foundation.

B. Hypergraph Structure Inference.

Existing hypergraph structure inference approaches can be divided into two categories: rule-based and supervised-learning-based approaches. The rule-based approach either assumes that nodes in a hyperedge have similar features, or that a hyperedge can be decomposed as a set of pairwise edges. The methods built upon the first assumption usually construct a hypergraph by setting hyperedges as clusters determined by the k-means algorithm [37]–[39]. Approaches under the second assumption typically create a hypergraph with specific connected components, e.g., cliques or communities, in a graph structure [40]. These rule-based methods can provide hypergraph structures with some desirable properties, but they fail to disclose the distribution behind the generated hypergraph structure, which limits their robustness and inference accuracy. On the other hand, recent works [41], [42] use ground-truth hypergraph structures associated with node features to train a neural network that learns a mapping between node features and ground-truth structures. After the mapping is learned, the neural network can estimate the probabilities of hyperedges and form a hypergraph structure accordingly. However, these supervised-learning-based approaches often require a significant amount of ground-truth hyperedges and associated node features for reliable training, hence limiting their application in scenarios where labelled data are scarce. We address the limitations of both categories in this paper. Leveraging the proposed Gaussian smoothness prior, we design an unsupervised hypergraph structure inference method that can estimate the probability for each potential hyperedge without training on labelled data.

C. Hypergraph Node Classification

There exists a body of works for classifying nodes residing on pre-defined hypergraphs by utilising node features and the hypergraph incidence matrix as inputs [6]. Most of them [11], [12], [23], [24], [43] extend the message-passing paradigm in graph machine learning [44] to develop hypergraph neural networks. This adaptation employs a two-stage message-passing paradigm: node features are first aggregated to hyperedges to update hyperedge embeddings, which are then aggregated back to nodes to update node embeddings. Despite the promising performance of these message-passing-based models, they have several limitations including high inference latency [6], [25], and sensitivity to structural perturbations [27], [28]. Leveraging our Gaussian smoothness prior, we design a framework named Hypergraph-MLP for hypergraph node classification without message passing. Removing the reliance on the hypergraph structure at inference not only makes Hypergraph-MLP have lower inference latency than the message-passing-based models, but also makes it robust to structural perturbations.

VII. Experiments

In this section, we empirically evaluate our smoothness-based approaches, namely HGSI and Hypergraph-MLP. We assess HGSI on the hypergraph structure inference task and Hypergraph-MLP on the hypergraph node classification task.

A. Hypergraph Structure Inference

Datasets. In the *synthetic dataset*, a hypergraph with node features is generated in two steps: 1) generate the ground-truth hypergraph structures based on the algorithm in [46]; 2) use the generated structures to get node features by $[\boldsymbol{X}_{\mathcal{V}}^T, \boldsymbol{X}_{\mathcal{E}}^T]^T \sim \mathcal{N}(\mathbf{0}, (\boldsymbol{L}_{\mathcal{H}} + \sigma^2\mathbf{I})^{-1})$, where we set the dimension of the node features as 1000, and σ is a small positive constant which is set as 10^{-3} in our experiments. Although we have the features of hyperedges in the synthetic dataset, these features are not used in our inference approach. The *real-world dataset* involves three real-world hypergraphs: SubCora, SubDBLP, and SubYelp, which are obtained from the pre-processed data in [11]. SubCora and SubDBLP are two co-authorship hypergraphs, where nodes are authors, a hyperedge containing authors of a specific paper, and the node features are formulated by the bag-of-words model with keywords related to the authors' research interests. SubYelp is a customership hypergraph, where a node is a customer, a hyperedge including customers of a specific restaurant, and node features are formulated by the bag-of-words model with keywords describing the dining preferences of each customer. SubCora contains 479 nodes each with a feature dimension of 1433 and 220 hyperedges whose sizes are 3 or 8. SubDBLP has 626 nodes each with a feature dimension of 1425 and 212 hyperedges whose sizes are 4. SubYelp includes 688 nodes each with a feature dimension of 1860 and 120 hyperedges whose sizes are 6. On the real-world data, we assume that we can only get the number of the overall target hyperedges. On the synthetic data, we assume that we can obtain the number of the target hyperedges belonging to different sizes.

Baselines. We choose GroupNet [38], HGSL [36], and NEO [45] as our baselines. These are unsupervised approaches. GroupNet assumes that each node contributes to at least one hyperedge whose internal nodes are highly correlated in terms of cosine similarity. HGSL learn the hypergraph structure from node features by doing community detection on a learnable line graph. NEO is an overlapping k-means algorithm, in which generated clusters are set as hyperedges in the hypergraph.

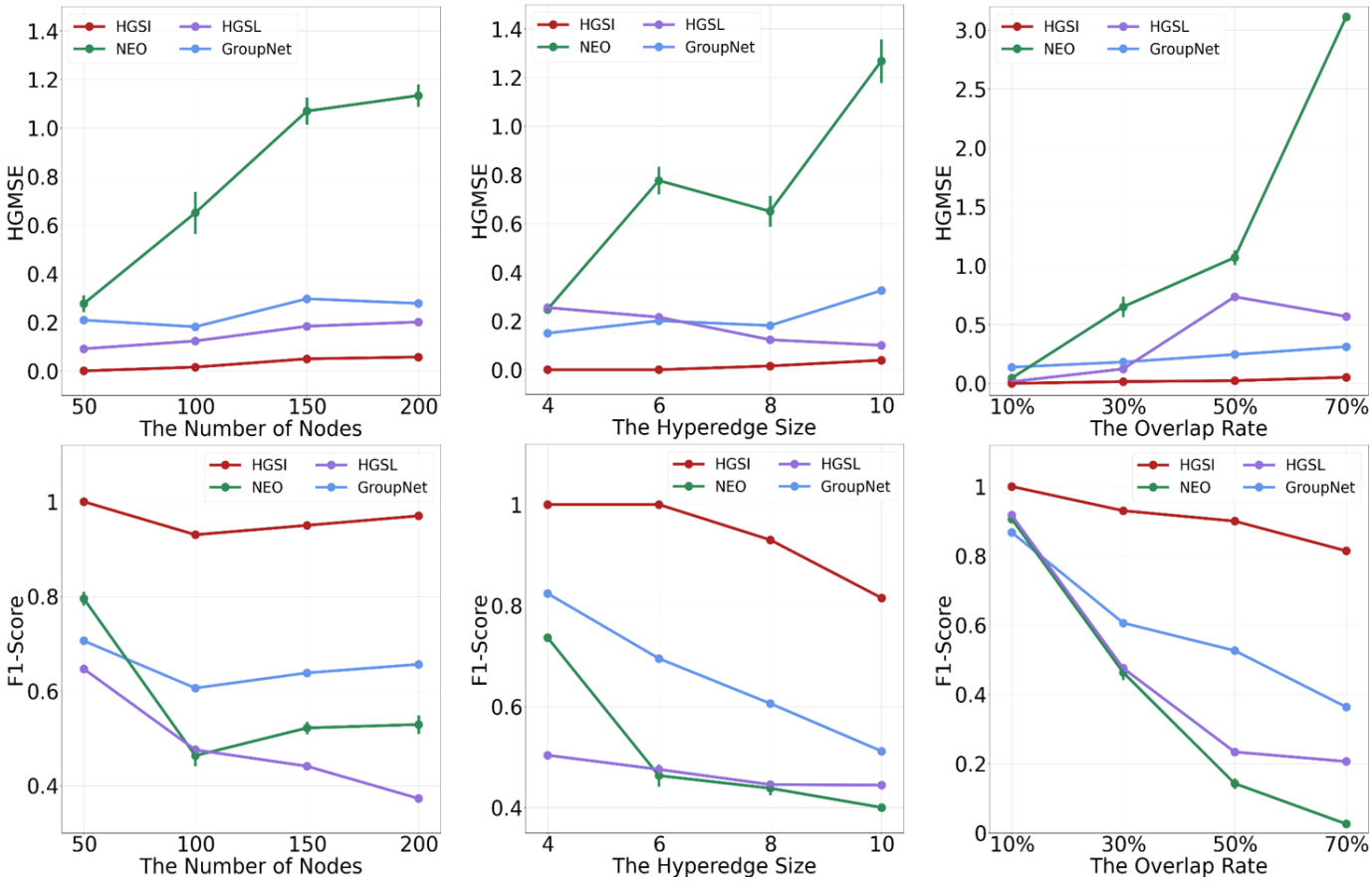
Metrics. We compare the proposed approach with its variation and baselines in finding the binary incidence matrix embedding the ground-truth hyperedges. We use F1-score and the normalised mean squared error for hypergraph recovery (HGMSE) to measure the performance of these methods. The higher the F1 score, the more accurate the binary incidence matrix is generated by the method. The smaller the HGMSE, the closer the generated binary incidence matrix and ground-truth incidence matrix are. In the following, we show the results of the average F1-Score/HGMSE and the associated std in 10 runs. Note that the performances of NEO and HGSL vary according to different random seeds, because NEO needs

TABLE I: F1-Score of methods on synthetic datasets with different numbers of hyperedge sizes and overlapping rates.

Models / Datasets	Overlap Rate10%		Overlap Rate 30%		Overlap Rate 50%	
	UniHG	MultiHG	UniHG	MultiHG	UniHG	MultiHG
GroupNet [38]	0.8676	0.2166	0.6059	0.1985	0.5260	0.1935
HGSL [36]	0.9173 ± 0.0024	0.8144 ± 0.0000	0.4754 ± 0.0044	0.4494 ± 0.0024	0.2335 ± 0.0018	0.1883 ± 0.0028
NEO [45]	0.9063 ± 0.0482	0.8666 ± 0.0142	0.4631 ± 0.0220	0.4105 ± 0.0132	0.1578 ± 0.0155	0.1425 ± 0.0094
HGSI	**1.0000**	**1.0000**	**0.9301**	**0.9112**	**0.9049**	**0.8984**

TABLE II: HGMSE of methods on synthetic datasets with different numbers of hyperedge sizes and overlapping rates.

Models / Datasets	Overlap Rate10%		Overlap Rate 30%		Overlap Rate 50%	
	UniHG	MultiHG	UniHG	MultiHG	UniHG	MultiHG
GroupNet [38]	0.1376	0.5711	0.1815	0.6047	0.3118	0.6218
HGSL [36]	0.0146 ± 0.0007	0.0306 ± 0.0000	0.1231 ± 0.0018	0.1891 ± 0.0015	0.7357 ± 0.0004	0.6341 ± 0.0007
NEO [45]	0.0445 ± 0.0004	0.0406 ± 0.0024	0.6504 ± 0.0872	0.8826 ± 0.0336	1.0691 ± 0.0634	2.1412 ± 0.0413
HGSI	**0.0000**	**0.0000**	**0.0155**	**0.0193**	**0.0230**	**0.0277**



(a) Impacted by the number of nodes. (b) Impacted by the hyperedge size. (c) Impacted by the overlap rate.

Fig. 5: The impacts brought by the properties of the ground-truth hypergraph structure.

to randomly initialize the cluster centres and HGSL needs to do random selection in its community detection step. The inference processes of GroupNet and HGSI are deterministic, so their performances do not vary according to random seeds.

Implementation details. All experiments were performed on 96 Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz using PyTorch. Our code is available at: https://github.com/tbh-98/A-Novel-Smoothness-Prior-for-Hypergraph-Machine-Learning/tree/main/HGSI.

Comparison with baselines on synthetic data. We use the synthetic data to show that HGSI can learn ground-truth hypergraph structures with varying properties from data fitting the hypergraph smoothness prior. We focus on four key structural metrics: the number of nodes, the hyperedge

TABLE III: F1-score of different methods in inferring real-world hypergraph structures.

Models / Datasets	SubCora	SubDBLP	SubYelp
GroupNet [38]	0.5310	0.8849	0.8898
HGSL [36]	0.4297 ± 0.0001	0.4698 ± 0.0000	0.7215 ± 0.0010
NEO [45]	0.4338 ± 0.0138	0.4474 ± 0.0103	0.7283 ± 0.0085
HGSI	**0.8909**	**0.9151**	**0.9250**

TABLE IV: HGMSE of different methods in inferring real-world hypergraph structures.

Models / Datasets	SubCora	SubDBLP	SubYelp
GroupNet [38]	0.4015	0.0758	0.0771
HGSL [36]	0.2001 ± 0.0001	0.0895 ± 0.0000	0.1401± 0.0000
NEO [45]	0.6069 ± 0.0206	0.4431 ± 0.0164	0.0915 ± 0.0140
HGSI	**0.0947**	**0.0425**	**0.0333**

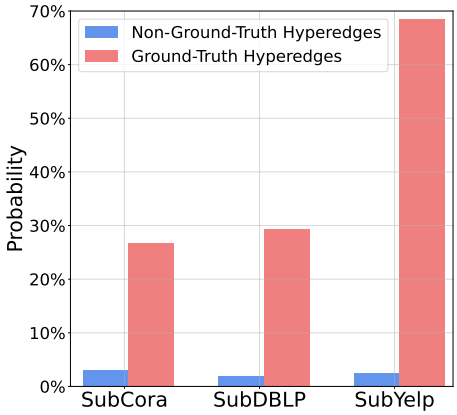


Fig. 6: The red/blue bar represents the average probabilities generated by HGSI for the ground-truth/non-ground-truth potential hyperedges in the potential hypergraph structure.

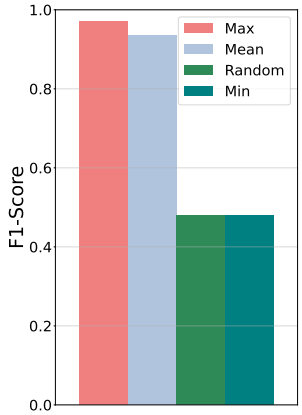


Fig. 7: The proposed smoothness criterion enables HGSI to achieve optimal performance on synthetic data.

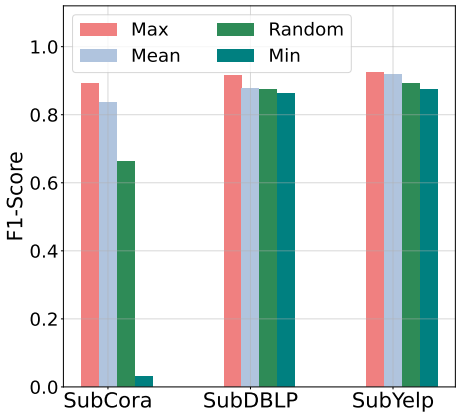


Fig. 8: The proposed smoothness criterion enables HGSI to achieve optimal performance on real-world data.

size, the overlap rate[2], and the number of hyperedge sizes. Firstly, fig. 5a shows the impact of the number of nodes in the ground-truth hypergraphs. Here we let all hyperedges in a hypergraph be 8-hyperedges and fix the overlap rate of each hypergraph as 30%. The increase in the number of nodes would cause an increase in the search space of the tested methods. According to the structure construction constraint proposed in section IV-B, the search space of our methods grows linearly with the number of nodes. Therefore, HGSI is hardly influenced by the increase in the number of nodes. Secondly, fig. 5b exhibits the impact brought by the size of hyperedges in the ground-truth hypergraphs. Here we fix the number of nodes, the overlap rate, and the number of hyperedge sizes of each hypergraph as 100, 30%, and 1 respectively. The growing sizes of hyperedges necessitate inference techniques that incorporate a more significant number of node features to identify relevant hyperedges, resulting in heightened inference complexity. Thus, the F1-Score for all employed methods declines as the hypergraph size within the ground-truth structure expands. Thirdly, fig. 5c displays the impact from the overlap rate in the ground-truth hypergraphs. Here we fix the number of nodes of each hypergraph as 100 and let all hyperedges in a hypergraph be 8-hyperedges. The escalating overlap rate in ground-truth hypergraphs can make nodes in different hyperedges have similar features, thus leading to a blurring of the boundaries between different hyperedges. Therefore, this exacerbates the challenges encountered during the inference process. Accordingly, the performances of all the methods drop when the overlap rate increases. Finally, table I and table II show the impact brought by the number of hyperedge sizes under different overlap rates. Here each hypergraph consists of 100 nodes. Moreover, UniHG denotes hypergraphs containing hyperedges in only size 8, and MultiHG represents hypergraphs including hyperedges in three different sizes: 7, 8, and 9. We note that this metric mainly influences the performance of GroupNet. This is because the number of hyperedges in its output increases significantly when the number of hyperedge sizes grows, which results in a substantial decrease in its precision.

Comparison with baselines on real-world data. The performance of each method in inferring the real-world hypergraph structures from node features is presented in table III and table IV. These results show that the proposed HGSI achieve state-of-the-art performance in inferring every real-world hypergraph structure with respect to both F1-Score and HGMSE. Furthermore, fig. 6 visualises the probabilities generated by HGSI for the ground-truth and non-ground-truth hyperedges in the potential hypergraph structure. These figures illustrate that, for every real-world hypergraph, HGSI generates much higher probabilities for ground-truth hyperedges compared to non-ground-truth hyperedges in the potential hypergraph structure. Both the quantitative results and the visualisation reflect that the proposed inference method can use the given node features to generate reliable probabilities for potential hyperedges in the real world. Additionally, the proposed hypergraph smoothness prior demonstrates the

[2]The overlap rate of a hyperedge is defined as the ratio of nodes in the hyperedge that are involved in more than one hyperedge to the total number of nodes. The average overlapping rate of a hypergraph is the mean of its hyperedges' overlapping rates [36]. For instance, the overlap rate of the hypergraph structure shown in fig. 3 is around 33.3%.

TABLE V: Properties of datasets.

	Cora	Citeseer	Pubmed	DBLP	20News	NTU2012	House
$\lvert\mathcal{V}\rvert$	2708	3312	19717	41302	16242	2012	1290
$\lvert\mathcal{E}\rvert$	1579	1079	7963	22363	100	2012	341
# features	1433	3703	500	1425	100	100	100
# class	7	6	3	6	4	67	2
Homophily	0.84	0.78	0.79	0.88	0.49	0.81	0.52

TABLE VI: Comparison with baselines on clean datasets. Mean testing **ACC** (%) ± standard deviation from 20 runs.

	Cora	Citeseer	Pubmed	DBLP	20News	NTU2012	House	Avg Mean
HyperGCN	78.45 ± 1.26	71.28 ± 0.82	82.84 ± 8.67	89.38 ± 0.25	81.05 ± 0.59	56.36 ± 4.86	78.22 ± 2.46	76.80
HGNN	79.39 ± 1.36	72.45 ± 1.16	86.44 ± 0.44	91.03 ± 0.20	80.33 ± 0.42	87.72 ± 1.35	66.16 ± 1.80	80.50
HCHA	79.14 ± 1.02	72.42 ± 1.42	86.41 ± 0.36	90.92 ± 0.22	80.33 ± 0.80	87.48 ± 1.87	67.91 ± 2.26	80.66
UniGCNII	78.81 ± 1.05	73.05 ± 2.21	88.25 ± 0.40	**91.69 ± 0.19**	81.12 ± 0.67	**89.30 ± 1.33**	80.65 ± 1.96	83.27
AllDeepSets	76.88 ± 1.80	70.83 ± 1.63	**88.75 ± 0.33**	91.27 ± 0.27	81.06 ± 0.54	88.09 ± 1.52	80.70 ± 1.59	82.51
AllSetTransformer	78.59 ± 1.47	73.08 ± 1.20	88.72 ± 0.37	91.53 ± 0.23	81.38 ± 0.58	88.69 ± 1.24	83.14 ± 1.92	83.59
MLP	74.99 ± 1.49	72.31 ± 1.28	87.69 ± 0.59	85.53 ± 0.27	81.70 ± 0.49	87.89 ± 1.36	83.78 ± 1.96	81.98
Hypergraph-MLP	**79.80 ± 1.82**	**73.90 ± 1.57**	87.89 ± 0.55	90.29 ± 0.26	**81.75 ± 0.41**	88.42 ± 1.32	**84.03 ± 1.75**	**83.72**

TABLE VII: Comparison with baselines on clean datasets. Mean **inference time** (ms) ± standard deviation from 60000 runs.

	Cora	Citeseer	Pubmed	DBLP	20News	NTU2012	House	Avg Mean
HyperGCN	0.46 ± 0.10	0.49 ± 0.10	0.60 ± 0.07	1.19 ± 0.21	0.82 ± 0.30	0.47 ± 0.10	0.50 ± 0.10	0.65
HGNN	1.40 ± 0.11	1.41 ± 0.33	3.33 ± 0.26	4.36 ± 0.41	1.39 ± 0.33	1.26 ± 0.11	1.43 ± 0.14	2.08
HCHA	1.33 ± 0.38	1.37 ± 0.14	3.43 ± 0.17	8.08 ± 0.76	1.45 ± 0.14	1.29 ± 0.15	1.46 ± 0.16	2.63
UniGCNII	3.01 ± 0.14	0.71 ± 0.10	0.85 ± 0.07	21.15 ± 0.64	3.79 ± 0.18	1.48 ± 0.08	0.77 ± 0.13	4.54
AllDeepSets	2.02 ± 0.15	2.40 ± 0.14	8.90 ± 1.09	21.51 ± 0.43	7.68 ± 0.57	1.96 ± 0.09	1.39 ± 0.13	6.55
AllSetTransformer	1.91 ± 1.34	3.01 ± 1.32	4.75 ± 1.33	21.21 ± 1.42	4.30 ± 1.57	1.90 ± 1.32	2.14 ± 1.29	5.60
Hypergraph-MLP	**0.45 ± 0.11**	**0.21 ± 0.07**	**0.19 ± 0.04**	**0.47 ± 0.17**	**0.28 ± 0.05**	**0.36 ± 0.08**	**0.28 ± 0.05**	**0.32**

ability to capture the relationship between the real-world hypergraph structure and the node features associated with it.

Ablation study on the probabilistic criterion. We empirically study how the probabilistic criterion based on the Gaussian-smoothness-based estimator, as defined in eq. (11), influences HGSI. We denote the proposed probabilistic criterion as Max and compare it with three other probabilistic criteria with different bases: Mean, Random and Min. Mean calculates the probabilistic criterion based on the average squared ℓ_2 distance between nodes in a hyperedge, Random uses the squared ℓ_2 distance of a random node pair, and Min considers the smallest squared ℓ_2 distance between nodes in the hyperedge. We conduct experiments on both synthetic and real-world data. For the synthetic data, we use the 200-node dataset with hyperedges of size 8 and a 30% overlap rate. For the real-world data, we use SubCora, SubDBLP, and SubYelp. The results are summarized as in fig. 7 and fig. 8, which show that the proposed probabilistic criterion enables HGSI to achieve its optimal performance. These findings confirm the crucial role of the proposed probabilistic criterion in HGSI.

B. Hypergraph Node Classification

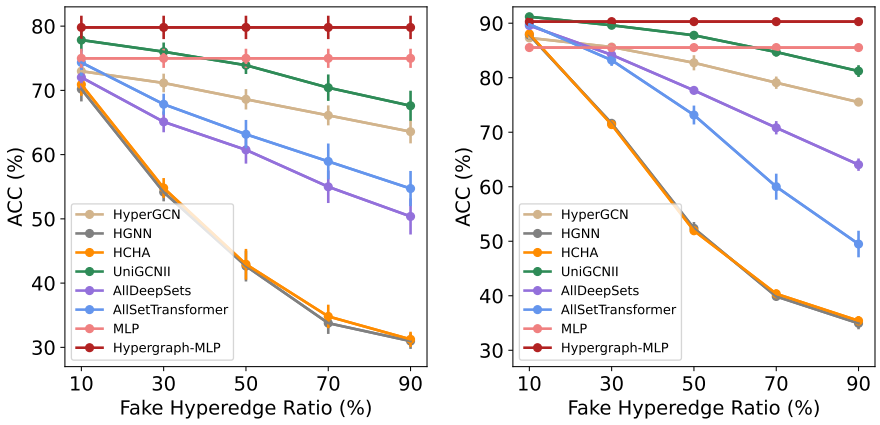
Datasets. We use seven public datasets, including academic hypergraphs (Cora, Citeseer, Pubmed, and DBLP), adapted from [23], 20News from UCI's Categorical Machine Learning Repository [47], NTU2012 from computer vision [48], and House from politics [49]. For the House dataset, lacking node features, we follow [11] and use Gaussian random vectors instead, where the standard deviation of the added Gaussian features is set as 0.6. Notably, the datasets 20News and House represent examples of heterophilic hypergraphs, while Cora, Citeseer, Pubmed, DBLP, and NTU2012 exemplify homophilic hypergraphs. Details of the datasets used are in table V.

Baselines. We compare Hypergraph-MLP with six message-passing hypergraph neural networks: HyperGCN [23], HGNN [12], HCHA [24], UniGCNI [43], AllDeepSets [11], and AllsetTransformer [11]; and a standard MLP, differing from our Hypergraph-MLP only in the loss function, using eq. (20)) instead of eq. (21).

Metrics. In accordance with prior research [11], [12], [23], [24], [43], we evaluate the performances of baselines and our method by accuracy (ACC: $\%$), which is defined as the ratio between the number of correct predictions to the total number of predictions.

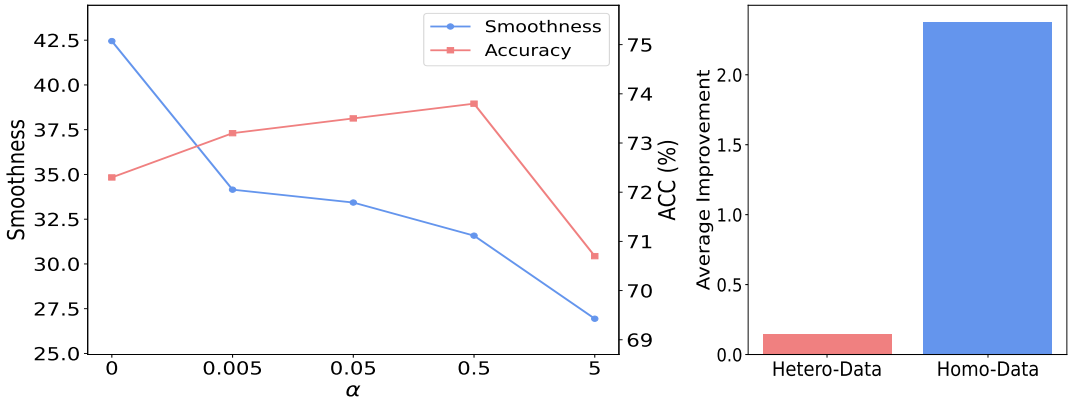
Implementation details. Following a previous work [11], we randomly partitioned the dataset into training, validation, and test sets using a 50%/25%/25% split. All experiments were performed on an RTX 3090 using PyTorch. We identified the optimal value of α in eq. (21) through grid search. Our code is available at: https://github.com/tbh-98/A-Novel-Smoothness-Prior-for-Hypergraph-Machine-Learning/tree/main/Hypergraph-MLP.

Comparison with baselines on clean data. The results for real-world hypergraph node classification benchmarks, conducted without considering structural perturbations, are summarized in table VI and table VII. In table VI, we present the ACC comparison of the proposed Hypergraph-MLP against all baseline methods. Our results demonstrate that



(a) Cora. (b) DBLP.

Fig. 9: Comparision with baselines on the perturbed Cora and DBLP. Mean **ACC** (%) ± standard deviation from 20 runs.



(a) Different hyperparameters. (b) Different data.

Fig. 10: Model behaviors.

Hypergraph-MLP not only outperforms the baseline methods across four datasets (Cora, Citeseer, 20News, and House) but also achieves the highest average mean ACC over all of the datasets. Furthermore, Hypergraph-MLP consistently outperforms the standard MLP across all the datasets. These findings underscore the efficacy of the smoothness-based loss function in effectively enabling the MLP-based model to leverage hypergraph structural information, all while not explicitly using the structure itself as input for message passing. As illustrated in table VII, Hypergraph-MLP demonstrates the fastest inference speed on all the seven chosen datasets compared to existing message-passing-based hypergraph neural networks. Notably, across the seven datasets, Hypergraph-MLP achieves an average mean inference time that is only 49% of the time required by the fastest message-passing-based model (HyperGCN) and a mere 5% of the time needed by the most time-consuming message-passing-based model (AllDeepSets). This attribute renders Hypergraph-MLP well-suited for applications with stringent latency constraints, such as autonomous driving and recommender systems, in contrast to previous hypergraph neural networks.

Comparison with baselines on perturbed data. We summarise the results on perturbed Cora and DBLP in fig. 9. We perturb hypergraph structures by randomly replacing original hyperedges with fake ones during inference. Moreover, we define the fake hyperedge ratio as $\frac{n_f}{n_o} * 100\%$, where n_f represents the number of fake hyperedges in the perturbed structure, and n_o is the number of hyperedges in the original structure. The results are consistent across perturbed Cora and DBLP: compared with message-passing-based hypergraph neural networks, the Hypergraph-MLP consistently demonstrates better robustness against structural perturbations. Eliminating the reliance on hypergraph structure during inference makes Hypergraph-MLP unaffected by structural perturbations introduced exclusively at inference. This characteristic positions Hypergraph-MLP as a robust choice for applications vulnerable to structure evasion attacks.

Analysis of model behaviours.[3] We study the impact of the hyperparameter α in eq. (21) and the homophily level of the datasets on the efficacy of Hypergraph-MLP. Firstly, we demonstrate how the smoothness of the generated node features and the classification accuracy of Hypergraph-MLP vary with different values of α, as illustrated infig. 10a. These results reveal that a modest α value facilitates Hypergraph-MLP in producing node features with an optimal level of smoothness, which, in turn, improve the classification accuracy. Subsequently, fig. 10b presents the average enhancement in classification accuracy that Hypergraph-MLP achieves over the vanilla MLP across both homophilic and heterophilic datasets. For this analysis, we set datasets with a homophily score below 0.6 as heterophilic, whereas those with scores above 0.6 as homophilic. The results in fig. 10b indicate a more pronounced improvement in classification accuracy with Hypergraph-MLP for homophilic datasets compared to heterophilic ones. This trend suggests that the superior performance of our model on MLP-based architectures primarily stems from our smoothness-based loss function, which effectively encourages the generation of similar features for connected nodes.

VIII. Conclusion

We propose a Gaussian smoothness prior for hypergraph machine learning. This prior models the data generation distribution of node features and hyperedge features on a hypergraph as a multivariate Gaussian distribution, whose covariance matrix is determined by the hypergraph structure itself. Leveraging this prior, we propose a Gaussian-smoothness-based estimator designed to estimate both the structure negative log-likelihood and the feature negative log-posterior. We apply this estimator to two key tasks: structure inference and node classification. Theron, we develop two novel frameworks: HGSI for inferring the latent higher-order interactions embedded within the data, and Hypergraph-MLP for classifying nodes on pre-established hypergraphs. Our extensive experiments, covering both hypergraph structure inference and node classification tasks, confirm the efficiency and effectiveness of our smoothness-based approaches. We believe both the proposed smoothness prior and the smoothness-based learning approaches can benefit the hypergraph research community through at least three distinct perspectives: 1) offering a fresh perspective in understanding the interactions between data and hypergraph structures; 2) providing a potent and effective method for inferring an appropriate hypergraph structure to capture the higher-order relationships among data; and 3) delivering an efficient and robust learning framework for learning node embeddings with pre-defined hypergraphs.

[3]In fig. 10a, we quantify the smoothness-level based on eq. (11). In fig. 10b, we define the average improvement as $\sum_{i=1}^{|D|} \frac{ACC^{H}_{D_i} - ACC^{v}_{D_i}}{|D|}$, where $|D|$ denotes the number of datasets, $ACC^{H}_{D_i}$ denotes the accuracy of the Hypergraph-MLP on dataset D_i, and $ACC^{V}_{D_i}$ denotes the accuracy of the vanilla MLP on dataset D_i.

References

[1] B. Tang, S. Chen, and X. Dong, "Hypergraph-mlp: learning on hypergraphs without message passing," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024.

[2] Y. Han, B. Zhou, J. Pei, and Y. Jia, "Understanding importance of collaborations in co-authorship networks: A supportiveness analysis approach," in *Proceedings of the 2009 SIAM International Conference on Data Mining*. SIAM, 2009, pp. 1112–1123.

[3] B. Jhun, "Effective epidemic containment strategy in hypergraphs," *Physical Review Research*, vol. 3, no. 3, p. 033282, 2021.

[4] R. Viñas, C. K. Joshi, D. Georgiev, P. Lin, B. Dumitrascu, E. R. Gamazon, and P. Liò, "Hypergraph factorization for multi-tissue gene expression imputation," *Nature machine intelligence*, vol. 5, no. 7, pp. 739–753, 2023.

[5] C. Bick, E. Gross, H. A. Harrington, and M. T. Schaub, "What are higher-order networks?" *SIAM Review*, vol. 65, no. 3, pp. 686–731, 2023.

[6] A. Antelmi, G. Cordasco, M. Polato, V. Scarano, C. Spagnuolo, and D. Yang, "A survey on hypergraph representation learning," *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–38, 2023.

[7] Y. Gao, Z. Zhang, H. Lin, X. Zhao, S. Du, and C. Zou, "Hypergraph learning: Methods and practices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 5, pp. 2548–2566, 2022.

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[9] A. Goyal and Y. Bengio, "Inductive biases for deep learning of higher-level cognition," *CoRR*, vol. abs/2011.15091, 2020.

[10] J. N. Kaur, E. Kiciman, and A. Sharma, "Modeling the data-generating process is necessary for out-of-distribution generalization," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. [Online]. Available: https://openreview.net/pdf?id=uyqks-LILZX

[11] E. Chien, C. Pan, J. Peng, and O. Milenkovic, "You are allset: A multiset function framework for hypergraph neural networks," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=hpBTIv2uy_E

[12] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 3558–3565.

[13] X. Zhu, J. Lafferty, and Z. Ghahramani, *Semi-supervised learning: From Gaussian fields to Gaussian processes*. School of Computer Science, Carnegie Mellon University, 2003.

[14] C. Godsil and G. F. Royle, *Algebraic graph theory*. Springer Science & Business Media, 2001, vol. 207.

[15] M. A. Bahmanian and M. Sajna, "Connection and separation in hypergraphs," *Theory and Applications of Graphs*, vol. 2, no. 2, p. 5, 2015.

[16] X. Ouvrard, "Hypergraphs: an introduction and review," *arXiv preprint arXiv:2002.05014*, 2020.

[17] V. Kalofolias and N. Perraudin, "Large scale graph learning from smooth signals," *arXiv preprint arXiv:1710.05654*, 2017.

[18] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, 2016.

[19] S. P. Chepuri, S. Liu, G. Leus, and A. O. Hero, "Learning sparse graphs under smoothness prior," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 6508–6512.

[20] X. Pu, T. Cao, X. Zhang, X. Dong, and S. Chen, "Learning to learn graph topologies," *Advances in Neural Information Processing Systems*, vol. 34, pp. 4249–4262, 2021.

[21] J. Jia and A. R. Benson, "A unifying generative model for graph learning algorithms: Label propagation, graph convolutions, and combinations," *SIAM Journal on Mathematics of Data Science*, vol. 4, no. 1, pp. 100–125, 2022. [Online]. Available: https://doi.org/10.1137/21M1395351

[22] M. Abramowitz, I. A. Stegun, and R. H. Romer, "Handbook of mathematical functions with formulas, graphs, and mathematical tables," 1988.

[23] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "Hypergcn: A new method for training graph convolutional networks on hypergraphs," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/1efa39bcaec6f3900149160693694536-Paper.pdf

[24] S. Bai, F. Zhang, and P. H. Torr, "Hypergraph convolution and hypergraph attention," *Pattern Recognition*, vol. 110, p. 107637, 2021.

[25] S. Zhang, Y. Liu, Y. Sun, and N. Shah, "Graph-less neural networks: Teaching old MLPs new tricks via distillation," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=4p6_5HBWPCw

[26] Y. Hu, H. You, Z. Wang, Z. Wang, E. Zhou, and Y. Gao, "Graph-mlp: Node classification without message passing in graph," *arXiv preprint arXiv:2106.04051*, 2021.

[27] L. Sun, Y. Dou, C. Yang, K. Zhang, J. Wang, S. Y. Philip, L. He, and B. Li, "Adversarial attack and defense on graph data: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[28] C. Hu, R. Yu, B. Zeng, Y. Zhan, Y. Fu, Q. Zhang, R. Liu, and H. Shi, "Hyperattack: Multi-gradient-guided white-box adversarial structure attack of hypergraph neural networks," *arXiv preprint arXiv:2302.12407*, 2023.

[29] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graphs," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 32, no. 32, 2010.

[30] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 44–63, 2019.

[31] Y. Chen, L. Wu, and M. Zaki, "Iterative deep graph learning for graph neural networks: Better and robust node embeddings," *Advances in neural information processing systems*, vol. 33, pp. 19 314–19 326, 2020.

[32] C. H. Nguyen and H. Mamitsuka, "Learning on hypergraphs with sparsity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2710–2722, 2021.

[33] M. Hein, S. Setzer, L. Jost, and S. S. Rangapuram, "The total variation on hypergraphs-learning on hypergraphs revisited," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[34] R. Qu, H. Feng, C. Xu, and B. Hu, "Analysis of hypergraph signals via high-order total variation," *Symmetry*, vol. 14, no. 3, p. 543, 2022.

[35] N. Veldt, A. R. Benson, and J. Kleinberg, "Combinatorial characterizations and impossibilities for higher-order homophily," *Science Advances*, vol. 9, no. 1, p. eabq3200, 2023.

[36] B. Tang, S. Chen, and X. 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.

[37] Y. Gao, Z. Zhang, H. Lin, X. Zhao, S. Du, and C. Zou, "Hypergraph learning: Methods and practices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 5, pp. 2548–2566, 2020.

[38] C. Xu, M. Li, Z. Ni, Y. Zhang, and S. 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.

[39] C. Xu, Y. Wei, B. Tang, S. Yin, Y. Zhang, and S. Chen, "Dynamic-group-aware networks for multi-agent trajectory prediction with relational reasoning," *arXiv preprint arXiv:2206.13114*, 2022.

[40] J.-G. Young, G. Petri, and T. P. Peixoto, "Hypergraph reconstruction from network data," *Communications Physics*, vol. 4, pp. 1–11, 2021.

[41] H. Serviansky, N. Segol, J. Shlomi, K. Cranmer, E. Gross, H. Maron, and Y. Lipman, "Set2graph: Learning graphs from sets," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 080–22 091, 2020.

[42] D. W. Zhang, G. J. Burghouts, and C. G. Snoek, "Pruning edges and gradients to learn hypergraphs from larger sets," in *Learning on Graphs Conference*. PMLR, 2022, pp. 53–1.

[43] J. Huang and J. Yang, "Unignn: a unified framework for graph and hypergraph neural networks," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021.

[44] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, 2020.

[45] J. J. Whang, Y. Hou, D. F. Gleich, and I. S. Dhillon, "Non-exhaustive, overlapping clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2644–2659, 2019.

[46] M. T. Do, S.-e. Yoon, B. Hooi, and K. Shin, "Structural patterns and generative models of real-world hypergraphs," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

[47] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[48] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On visual similarity based 3d model retrieval," in *Computer graphics forum*, vol. 22, no. 3. Wiley Online Library, 2003, pp. 223–232.

[49] P. S. Chodrow, N. Veldt, and A. R. Benson, "Hypergraph clustering: from blockmodels to modularity," *arXiv preprint arXiv:2101.09611*, 2021.