HiTeC: Hierarchical Contrastive Learning on Text-Attributed Hypergraph with Semantic-Aware Augmentation

Anonymous ACL submission

Abstract

Contrastive learning (CL) has become a dominant paradigm for self-supervised hypergraph learning, enabling effective training without costly labels. However, node entities in real-world hypergraphs are often associated with rich textual information, which has been largely ignored in prior works. Directly applying existing CL-based methods to such text-attributed hypergraphs (TAHGs) leads to three key limitations: (1) The common use of graph-agnostic text encoders fails to capture the correlations between textual semantics and hypergraph topology, resulting in less expressive representations. (2) Their reliance on random data augmentations introduces noise and weakens the contrastive signals. (3) The primary focus on node- and hyperedge-level contrastive signals limits the ability to capture long-range dependencies, which is essential for effective representation learning. To address these challenges, we introduce HiTeC, a two-stage hierarchical contrastive learning framework for effective self-supervised learning on TAHGs. In the first stage, we pre-train the text encoder with a structure-aware contrastive objective to overcome the graph-agnostic nature of conventional methods. In the second stage, we begin by introducing semantic-aware augmentations, including structure-contextualized text augmentation and semantic-aware hyperedge dropping, to facilitate informative view generation. Subsequently, we propose a multi-scale contrastive loss with an s-walk-based subgraph-level objective to capture long-range dependencies. Extensive experiments on six real-world datasets validate the effectiveness of our proposed method.

1 Introduction

Hypergraphs extend traditional graphs by allowing each hyperedge to connect multiple nodes, offering a natural way to model multi-way relationships. Such higher-order structures are prevalent in complex real-world systems, e.g., cortical co-activation in brains (Yu et al., 2011), user-item interactions in e-commerce platforms (Xia et al., 2022), and multi-author collaborations in academic networks (Kim et al., 2023). In these scenarios, nodes are often associated with rich textual attributes (e.g., product reviews in user–item networks and paper abstracts in co-authorship networks), which provide essential semantic information for downstream applications. While Hypergraph Neural Networks (HNNs) have emerged as a powerful paradigm for hypergraph learning (Chien et al., 2022b; Kim et al., 2024b), effectively capturing high-order structural dependencies, they primarily rely on numerical features and task-specific labels. This reliance limits their effectiveness in real-world applications, where labeled data is scarce, and node attributes are predominantly textual. This motivates the development of self-supervised learning (SSL) methods tailored to text-attributed hypergraphs (TAHGs) (Bazaga et al., 2024; Nakajima and Uno, 2025).

Recent advances in hypergraph self-supervised learning (HSSL) primarily focus on the contrastive paradigm, which maximizes agreement between two augmented views derived from the original hypergraph. HyperGCL (Wei et al., 2022) pioneers this direction with a learnable augmentation strategy. TriCL (Lee and Shin, 2023) introduces tri-directional contrastive objectives over node-, hyperedge-, and membership-level relations in hypergraphs. SE-HSSL (Li et al., 2024) further redesigns a hierarchical membership-level objective, achieving state-of-the-art (SOTA) performance. Beyond contrastive methods, generative approaches have also gained initial attention. HypeBoy (Kim et al., 2024a) proposes a hyperedge-filling pretext task to learn expressive hypergraph representations.

Despite recent progress, current HSSL approaches still face significant limitations on hypergraphs with textual attributes: (1) They typically use shallow encoders (e.g., Bag-of-Words (Zhang et al., 2010) and Skip-Gram (Mikolov et al., 2013))

or pre-trained language models (e.g., BERT (Devlin et al., 2019)) to convert raw text into numerical features while ignoring hypergraph structure. This graph-agnostic design overlooks the correlation between text and higher-order topology, leading to suboptimal representations (Chien et al., 2022a). (2) Current hypergraph contrastive learning (HCL) methods often apply random augmentations to features and structure, e.g., random masking of text or hyperlinks. These perturbations may distort textual meaning and disrupt the alignment between structure and semantic content, introducing noise into the contrastive objective (Fang et al., 2024). (3) Existing HCL methods mainly emphasize node- and hyperedge-level signals, but struggle to capture long-range dependencies and richer topological contexts that are crucial for expressive representation learning (Zhang et al., 2024).

HyperBERT (Bazaga et al., 2024) takes an early attempt at SSL on TAHGs by jointly training text and hypergraph encoders through contrastive learning. However, its contrastive objective mainly relies on 1-hop hyperedge relations, limiting its ability to capture multi-scale dependencies that have proven beneficial for HSSL (Lee and Shin, 2023; Li et al., 2024). Meanwhile, recent text-attributed graph studies (Chien et al., 2022a; Fang et al., 2024; Zhang et al., 2024) explore joint text–structure modeling, but their extension to TAHGs remains challenging due to insufficient modeling of higher-order relations inherent to hypergraphs.

To address these issues, we propose HiTeC, a hierarchical contrastive learning framework for effective SSL on TAHGs. HiTeC adopts a two-stage framework that first injects higher-order topology into text representations and then performs semantic-aware hypergraph contrastive learning. In the first stage, we pre-train a text encoder with a structure-aware contrastive objective to align textual semantics with higher-order topology, producing topology-enhanced text representations. In the second stage, we pre-train a hypergraph encoder with semantic-aware augmentations and hierarchical contrastive objectives. Specifically, we introduce structure-contextualized text augmentation and semantic-aware hyperedge dropping to alleviate perturbation noise. Based on the augmented views, we further formulate node-, hyperedge-, and subgraph-level contrastive objectives, where an s-walk-based subgraph signal captures long-range dependencies and richer structural contexts. Our main contributions are as follows:

- We propose HiTeC, an effective two-stage contrastive learning framework for TAHGs that addresses key limitations of existing HSSL methods in modeling rich textual attributes and complex hypergraph structure.
- We introduce a structure-aware contrastive objective for text encoder pretraining to integrate textual content with hypergraph structure.
- We enhance the hypergraph encoder with two semantic-aware augmentation strategies and hierarchical contrastive objectives, where structure-contextualized text augmentation and semantic-aware hyperedge dropping are used to reduce perturbation noise, and an s-walk-based subgraph-level contrastive signal captures long-range structural dependencies.
- Extensive experiments on six real-world datasets demonstrate the effectiveness and superiority of HiTeC over strong baselines across multiple downstream tasks.

2 Preliminary

Notations. We define a text-attributed hypergraph (TAHG) as $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \mathcal{T}\}$, where $\mathcal{V} = \{v_i\}_{i=1}^{|\mathcal{V}|}$ denotes the set of nodes, and $\mathcal{E} = \{e_j\}_{j=1}^{|\mathcal{E}|}$ denotes the set of hyperedges, where each hyperedge $e \in \mathcal{E}$ is a non-empty subset of nodes, and $|e|$ is the size of e. Each node v_i is associated with a textual sequence $t_i \in \mathcal{T}$, where $\mathcal{T}$ denotes the set of all text attributes. Each hyperedge e_j is associated with a positive scalar weight w_j, and all weights formulate a diagonal matrix $\mathbf{W} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$. The hypergraph structure can be modeled as an incidence matrix $\mathbf{H} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{E}|}$, where $h_{ij} = 1$ if node v_i belongs to hyperedge e_j, and $h_{ij} = 0$ otherwise. We define the node degree matrix $\mathbf{D}_v \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, where each entry is $d(v_i) = \sum_{e_j\in\mathcal{E}} w_j \cdot h_{ij}$. Similarly, the hyperedge degree matrix is defined as $\mathbf{D}_e \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$, where each element $\delta(e_j) = \sum_{v_i\in e_j} h_{ij}$ represents the number of nodes connected by the hyperedge e_j.

Hypergraph neural networks. HNNs have emerged as effective tools for representation learning on hypergraphs (Feng et al., 2019a; Bai et al., 2021; Chien et al., 2022b). Most HNNs adopt a two-stage message passing mechanism that alternates between node-to-hyperedge and hyperedge-to-node aggregation. At each layer l, the embedding of a hyperedge is updated based on its incident

nodes, while a node embedding is updated via its associated hyperedges:

$$\begin{aligned} z_{e_j}^{(l)} &= f_{\mathcal{V}\to\mathcal{E}}^{(l)}\left(z_{e_j}^{(l-1)}, \{z_{v_k}^{(l-1)} \mid v_k \in e_j\}\right), \\ z_{v_i}^{(l)} &= f_{\mathcal{E}\to\mathcal{V}}^{(l)}\left(z_{v_i}^{(l-1)}, \{z_{e_k}^{(l)} \mid v_i \in e_k\}\right), \end{aligned} \tag{1}$$

where $z_{e_j}^{(l)}$ and $z_{v_i}^{(l)}$ denote the embeddings of hyperedge e_j and node v_i at layer l, respectively. The functions $f_{\mathcal{V}\to\mathcal{E}}^{(l)}$ and $f_{\mathcal{E}\to\mathcal{V}}^{(l)}$ are permutation-invariant aggregation functions that propagate messages between nodes and hyperedges. This iterative process allows the model to capture complex higher-order relations in the hypergraph structure.

Problem statement. Given a TAHG $\mathcal{H} = \{\mathcal{V}, \mathcal{E}, \mathcal{T}\}$, the objective is to learn an optimal function $\Psi : \mathcal{H} \to (\mathbf{Z}_{\mathcal{V}}, \mathbf{Z}_{\mathcal{E}})$ in an unsupervised manner, mapping the hypergraph to low-dimensional node and hyperedge embeddings. These representations may then be applied to downstream tasks such as node classification and hyperedge prediction.

3 Methodology

In this section, we present HiTeC, a novel and effective SSL framework tailored for TAHGs. As illustrated in Figure 1, HiTeC adopts a two-stage paradigm. In the first stage, a text encoder f_θ is pre-trained with a structure-aware contrastive objective to capture topology-enhanced text representations. In the second stage, with f_θ frozen, we pre-train a hypergraph encoder g_ϕ to fully exploit the rich semantic context of nodes through semantic-aware augmentations and hierarchical contrastive objectives at the node, hyperedge, and subgraph levels.

3.1 Structure-aware Text Encoder Pretraining

Text encoder. Our text encoder f_θ is built upon a pre-trained language model, followed by a two-layer MLP projection head that maps the [CLS] representation to a d-dimensional embedding. Given a node v_i with textual attribute t_i, the resulting text embedding $x_i \in \mathbb{R}^d$ is computed as:

$$x_i = f_\theta(t_i). \tag{2}$$

Structure-aware contrastive objective. In real-world hypergraphs, nodes are often semantically closer to their local neighbors than to distant ones. To encode this structural prior, we treat the 1-hop neighbors $N_{v_i}^{(1)}$ of v_i, i.e., nodes sharing at least one hyperedge with v_i, as positives, while the remaining nodes $\mathcal{V} \setminus N_{v_i}^{(1)}$ are treated as negatives. Positive and negative representations are constructed by mean pooling the corresponding text embeddings:

$$\begin{aligned} x_i^+ &= \frac{1}{|N_{v_i}^{(1)}|} \sum_{v_j \in N_{v_i}^{(1)}} x_j, \\ x_i^- &= \frac{1}{|\mathcal{V} \setminus N_{v_i}^{(1)}|} \sum_{v_j \in \mathcal{V}\setminus N_{v_i}^{(1)}} x_j. \end{aligned} \tag{3}$$

A triplet margin loss (Schroff et al., 2015) is adopted to guide f_θ in distinguishing positive sample x_i^+ from negative sample x_i^-. For each anchor x_i, the loss is defined as:

$$\ell_t(x_i, x_i^+, x_i^-) = \max\left(\cos(x_i, x_i^-) - \cos(x_i, x_i^+) + m, 0\right), \tag{4}$$

where $\cos(\cdot)$ denotes cosine similarity, and m is the margin hyperparameter. The final objective is:

$$\mathcal{L}_t = \frac{1}{|\mathcal{V}|} \sum_{v_i \in \mathcal{V}} \ell_t(x_i, x_i^+, x_i^-). \tag{5}$$

3.2 Semantic-aware Hypergraph Encoder Pretraining

Existing HCL methods typically overlook node textual semantics and rely on random augmentation strategies, resulting in suboptimal performance (Fang et al., 2024; Wu et al., 2024). Moreover, they struggle to model long-range dependencies, limiting broader contextual understanding (Zhang et al., 2024; Wu et al., 2024). To tackle these issues, we propose a semantic-aware contrastive learning strategy to pre-train g_ϕ. As shown in Figure 1, this stage comprises three components: semantic-aware augmentation, hypergraph encoding, and hierarchical contrastive objectives.

Semantic-aware augmentation. To eliminate the impact of noise arising from random perturbation, we introduce two semantic-aware augmentation strategies: structure-contextualized text augmentation and semantic-aware hyperedge dropping.

Structure-contextualized text augmentation. As perturbations applied directly to text embeddings can easily distort semantic meaning, we instead augment raw text with structure-aware contextual information. Specifically, for each node text attribute t_i, we first construct a structure-contextualized text view by concatenating three contextual components with the original text: $\tilde{t}_{1,i} = [\texttt{[DK]} \,||\, \texttt{[GT]} \,||\, \texttt{[NC]} \,||\, t_i]$, where: (i) [DK]

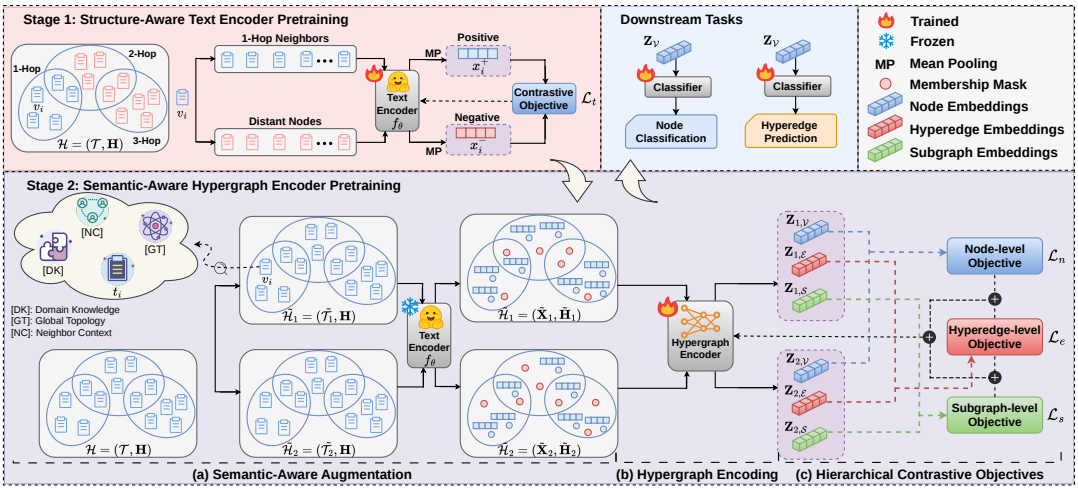


Figure 1: The overall framework of HiTeC.

denotes domain knowledge (e.g., academic or e-commerce) shared across all nodes; (ii) [GT] encodes global topological information for node v_i (e.g., node degree and local sparsity); and (iii) [NC] represents neighbor context, constructed by concatenating the raw texts of sampled neighbors of v_i. The second text view is the original text, i.e., $\tilde{t}_{2,i} = t_i$. This simple yet effective strategy generates semantically consistent but contextually diverse text variants, which are encoded by the shared encoder f_θ into a unified embedding space, denoted as $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$, respectively. The full template for constructing the structure-contextualized text view is provided in Appendix A.

Semantic-aware hyperedge dropping. Effective structure augmentation should consider both textual semantics and topology. To achieve this, we define a semantic cohesiveness score that reflects both the structural grouping of nodes and their semantic similarity, which is further used to guide the drop probability of each node–hyperedge incidence. Connections within more cohesive hyperedges are more likely to be preserved, encouraging the retention of structurally and semantically aligned patterns, while selectively pruning noisy or weakly node–hyperedge connections.

Definition 1 (***semantic cohesiveness score***). *Given a TAHG $\mathcal{H}(\mathcal{V}, \mathcal{E}, \mathbf{X})$, the semantic cohesiveness score of a hyperedge $e \in \mathcal{E}$ is defined as the average pairwise similarity among its member nodes. Let $\mathbf{X}_e \in \mathbb{R}^{|e| \times d}$ denote the normalized feature matrix for nodes in e. The semantic cohesiveness score $s(e)$ is then computed as:*

$$s(e) = \frac{1}{\binom{|e|}{2}} \sum_{i=1}^{|e|} \sum_{j=i+1}^{|e|} \boldsymbol{S}_e(i,j), \quad \boldsymbol{S}_e = \mathbf{X}_e \mathbf{X}_e^\top,$$

where $\boldsymbol{S}_e(i,j)$ denotes the similarity between the i-th and j-th nodes in hyperedge e.

The drop probability of each incidence (v_i, e_j), where $v_i \in e_j$, is then determined by $s(e_j)$:

$$p_{\text{drop}}(v_i, e_j) = 1 - \sigma\left(\frac{s(e_j) - 0.5}{\tau_{\text{drop}}}\right), \quad (6)$$

where $\sigma(\cdot)$ is the sigmoid function and τ_{drop} is a temperature hyperparameter. Higher cohesiveness scores correspond to lower drop probabilities. In implementation, we construct a binary masking matrix $\mathbf{M} \in \{0,1\}^{|\mathcal{V}| \times |\mathcal{E}|}$, where entries $m_{ij} \sim \mathcal{B}(1 - p_{\text{drop}}(v_i, e_j))$ are sampled from a Bernoulli distribution. The perturbed matrix $\tilde{\mathbf{H}}$ is:

$$\tilde{\mathbf{H}} = \mathbf{M} \odot \mathbf{H}, \quad (7)$$

where $\odot$ denotes element-wise multiplication. We apply this semantic-aware hyperedge drop mechanism to both views, yielding two augmented hypergraphs: $\tilde{\mathcal{H}}_1 = (\tilde{\mathbf{X}}_1, \tilde{\mathbf{H}}_1)$ and $\tilde{\mathcal{H}}_2 = (\tilde{\mathbf{X}}_2, \tilde{\mathbf{H}}_2)$. This procedure complements text-level augmentation by introducing structural diversity while retaining semantic consistency, enabling the hypergraph encoder to learn more robust node representations.

Hypergraph encoder. We take the HGNN (Feng et al., 2019b) with the element-wise mean pooling layer as our hypergraph encoder. At the l-th layer, the hyperedge representation $\mathbf{Z}_{\mathcal{E}}^{(l)} \in \mathbb{R}^{|\mathcal{E}| \times d}$ and the node representation $\mathbf{Z}_{\mathcal{V}}^{(l)} \in \mathbb{R}^{|\mathcal{V}| \times d}$ are updated as:

$$\begin{aligned} \mathbf{Z}_{\mathcal{E}}^{(l)} &= \rho\left(\mathbf{D}_e^{-1} \mathbf{H}^\top \mathbf{Z}_{\mathcal{V}}^{(l-1)} \boldsymbol{\Theta}_{\mathcal{E}}^{(l)} + \boldsymbol{b}_e^{(l)}\right), \\ \mathbf{Z}_{\mathcal{V}}^{(l)} &= \rho\left(\mathbf{D}_v^{-1} \mathbf{H} \mathbf{W} \mathbf{Z}_{\mathcal{E}}^{(l)} \boldsymbol{\Theta}_{\mathcal{V}}^{(l)} + \boldsymbol{b}_v^{(l)}\right), \end{aligned} \quad (8)$$

where $\rho(\cdot)$ is the PReLU activation function, $\boldsymbol{\Theta}_{\mathcal{E}}^{(l)}$ and $\boldsymbol{\Theta}_{\mathcal{V}}^{(l)}$ are trainable weight matrices, and $\boldsymbol{b}_e^{(l)}$ and

$\boldsymbol{b}_v^{(l)}$ are bias terms. $\mathbf{W}$ is initialized as an identity matrix, assigning equal weights to all hyperedges.

Hierarchical contrastive objectives. Prior works (Lee and Shin, 2023; Li et al., 2024) have shown that multi-scale contrastive objectives are beneficial for HSSL. To better capture structural semantics at different granularities, we adopt a hierarchical contrastive learning framework across node, hyperedge, and subgraph levels. In particular, we introduce a novel subgraph-level objective based on s-walk sampling to capture long-range dependencies beyond local structures. All objectives are optimized using the widely adopted contrastive loss, InfoNCE (Oord et al., 2018).

Node-level objective. We first model the correlation of nodes across two augmented views. Given a node $v_i \in \mathcal{V}$, let $\mathbf{z}_{1,v_i}$ and $\mathbf{z}_{2,v_i}$ denote its node embeddings derived from two augmented hypergraphs. By fixing $\mathbf{z}_{1,v_i}$ as anchor, we treat$(\mathbf{z}_{1,v_i}, \mathbf{z}_{2,v_i})$ as a positive pair, and all $\mathbf{z}_{2,v_k}$ for $k \neq i$ as negative samples. The loss of anchor $\mathbf{z}_{1,v_i}$ is:

$$\ell_n(\mathbf{z}_{1,v_i}, \mathbf{z}_{2,v_i}) = -\log \frac{e^{\cos(\mathbf{z}_{1,v_i}, \mathbf{z}_{2,v_i})/\tau_n}}{\sum_{k=1}^{|\mathcal{V}|} e^{\cos(\mathbf{z}_{1,v_i}, \mathbf{z}_{2,v_k})/\tau_n}}, \tag{9}$$

where τ_n is the temperature coefficient. In practice, we further symmetrize the contrastive loss by using the second view's embedding as the anchor, yielding the final node-level contrastive objective:

$$\mathcal{L}_n = \frac{1}{2|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \left[\ell_n(\mathbf{z}_{1,v_i}, \mathbf{z}_{2,v_i}) + \ell_n(\mathbf{z}_{2,v_i}, \mathbf{z}_{1,v_i})\right]. \tag{10}$$

Hyperedge-level objective. To consider group-level knowledge, we integrate hyperedge-level contrastive signal during optimization. Given a hyperedge $e_j \in \mathcal{E}$, let $\mathbf{z}_{1,e_j}$ and $\mathbf{z}_{2,e_j}$ be its hyperedge representations from the two augmented views. Similar to the node-level objective, the contrastive loss with $\mathbf{z}_{1,e_j}$ as anchor is formulated as:

$$\ell_e(\mathbf{z}_{1,e_j}, \mathbf{z}_{2,e_j}) = -\log \frac{e^{\cos(\mathbf{z}_{1,e_j}, \mathbf{z}_{2,e_j})/\tau_e}}{\sum_{k=1}^{|\mathcal{E}|} e^{\cos(\mathbf{z}_{1,e_j}, \mathbf{z}_{2,e_k})/\tau_e}}, \tag{11}$$

where τ_e is the temperature coefficient. By symmetrizing the contrastive loss, the final hyperedge-level objective can be written as:

$$\mathcal{L}_e = \frac{1}{2|\mathcal{E}|} \sum_{j=1}^{|\mathcal{E}|} \left[\ell_e(\mathbf{z}_{1,e_j}, \mathbf{z}_{2,e_j}) + \ell_e(\mathbf{z}_{2,e_j}, \mathbf{z}_{1,e_j})\right]. \tag{12}$$

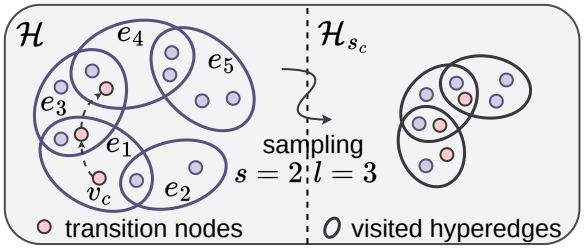


Figure 2: An example of the s-walk-based subgraph sampling process. Given the center node v_c, we perform an s-walk with $s = 2$ and $l = 3$ on $\mathcal{H}$. The resulting hyperedge traversal sequence is $[e_1, e_3, e_4]$, and the sampled subgraph is denoted as $\mathcal{H}_{s_c}$.

Subgraph-level objective. While node- and hyperedge-level objectives effectively capture local structural signals, they remain insufficient for modeling long-range dependencies in hypergraphs. We therefore introduce a subgraph-level objective based on an s-walk sampling strategy. Unlike random walks, an s-walk defines a high-order traversal sequence in hypergraphs, where s controls the overlap strength between consecutive hyperedges, enabling the extraction of cohesive and semantically meaningful substructures unique to hypergraphs (Aksoy et al., 2020; Preti et al., 2024). We next formally define s-walk and describe our subgraph sampling method in detail.

Definition 2 (***s-walk***). *Given $s \in \mathbb{N}^+$, an s-walk of length l is defined as an ordered sequence of hyperedges $[e_1, \ldots, e_l]$, where every consecutive pair of hyperedges shares at least s common nodes, i.e., $|e_j \cap e_{j+1}| \geq s$ for all $j \in [1, l-1]$.*

As illustrated in Figure 2, an s-walk-based subgraph $\mathcal{H}_{s_c} = (\mathcal{V}_{s_c}, \mathcal{E}_{s_c})$ is constructed from a center node v_c via iterative s-walk sampling. We first randomly select a hyperedge e_1 containing v_c. At each step, we identify hyperedges sharing at least s nodes with the current hyperedge e_j, randomly sample one as e_{j+1}, and then randomly choose a node from e_{j+1} as the transition node. The process stops when the traversal length reaches l. The sampled subgraph consists of the visited hyperedges $\mathcal{E}_{s_c} = \{e_j\}_{j=1}^{l}$ and their associated nodes $\mathcal{V}_{s_c} = \bigcup_{j=1}^{l} e_j$. Each sampled subgraph $\mathcal{H}_{s_c}$ is encoded by the shared hypergraph encoder $g_\phi(\cdot)$, and its representation is obtained through mean pooling P_{mean} over the corresponding node embeddings:

$$\mathbf{z}_{s_c} = P_{\text{mean}}(g_\phi(\mathcal{H}_{s_c})). \tag{13}$$

However, constructing and encoding subgraphs for all nodes incurs a high computational cost. To

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	Avg.$\lvert e\rvert$	# Avg.tokens	# Class
Citeseer	1,778	2,118	2	198	6
Cora	2,708	1,579	3	189	7
History	41,551	169,454	9	300	12
Photo	48,362	212,247	11	189	12
Computers	87,229	277,539	9	115	10
Fitness	173,055	1,468,229	11	28	13

Table 1: Statistics of datasets.

mitigate this, we follow prior findings suggesting that high-degree nodes tend to carry richer contextual information (Fang et al., 2025). Specifically, we select a representative node subset $\mathcal{V}_r \subseteq \mathcal{V}$ by ranking node degrees and retaining the top $r\%$. As validated in our experiments, this strategy maintains competitive performance even with a significantly reduced number of subgraphs. For each anchor node $v_m \in \mathcal{V}_r$, we obtain subgraph representations $\mathbf{z}_{1,s_m}$ and $\mathbf{z}_{2,s_m}$ from two augmented views. Similar to the former contrastive signals, the subgraph-level loss with anchor $\mathbf{z}_{1,s_m}$ is defined as:

$$\ell_{\mathrm{s}}(\mathbf{z}_{1,s_m}, \mathbf{z}_{2,s_m}) = -\log \frac{e^{\cos(\mathbf{z}_{1,s_m}, \mathbf{z}_{2,s_m})/\tau_s}}{\sum_{k=1}^{|\mathcal{V}_r|} e^{\cos(\mathbf{z}_{1,s_m}, \mathbf{z}_{2,s_k})/\tau_s}}, \tag{14}$$

where τ_s is a temperature coefficient. And the final symmetric loss across all anchors is:

$$\mathcal{L}_{\mathrm{s}} = \frac{1}{2|\mathcal{V}_r|} \sum_{m=1}^{|\mathcal{V}_r|} \left[\ell_s(\mathbf{z}_{1,s_m}, \mathbf{z}_{2,s_m}) + \ell_{\mathrm{s}}(\mathbf{z}_{2,s_m}, \mathbf{z}_{1,s_m})\right] \tag{15}$$

Overall objective. Combining Eq. (10), (12), and (15), the overall contrastive loss is defined as: contrastive loss is defined as:

$$\mathcal{L} = \mathcal{L}_n + \lambda_e \mathcal{L}_e + \lambda_s \mathcal{L}_s, \tag{16}$$

where λ_e and λ_s are balancing weights. A detailed complexity analysis is provided in Appendix B.

4 Experiments

4.1 Experimental Setup

Datasets. As no public TAHG datasets are available, we construct six text-attributed hypergraph datasets via max clique-based hypergraph reconstruction (Bron and Kerbosch, 1973; Wang and Kleinberg, 2024). They include two co-citation hypergraphs (Citeseer (Giles et al., 1998) and Cora (Sen et al., 2008)) and four co-purchasing networks (History, Photo, Computers, and Fitness (Yan et al., 2023)). Dataset statistics are summarized in Table 1, with details in Appendix C.1.

Baselines. We compare HiTeC with eight representative SSL baselines: (1) three graph-based methods (GraphCL (You et al., 2020), BGRL (Thakoor et al., 2022), and GraphMAE (Hou et al., 2022)); (2) four hypergraph-based methods (TriCL (Lee and Shin, 2023), SE-HSSL (Li et al., 2024), HypeBoy (Kim et al., 2024a), and VilLain (Lee et al., 2024)); and (3) one TAHG-based method, HyperBERT (Bazaga et al., 2024). Since graph- and hypergraph-based methods cannot process raw text directly, we extract node features using different text encoders. Specifically, both graph- and hypergraph-based methods use shallow encoders (SE), i.e., Skip-Gram (Mikolov et al., 2013). Graph-based methods additionally use GIANT (Chien et al., 2022a), a self-supervised encoder for text-attributed graphs (TAGs), while hypergraph-based methods use pretrained language models, including BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Detailed descriptions of the baselines are provided in Appendix C.2.

Evaluation protocol. We evaluate our approach on two hypergraph benchmark tasks: node classification and hyperedge prediction. After pretraining with HiTeC, we adopt the linear evaluation protocol (Lee and Shin, 2023; Li et al., 2024; Kim et al., 2024a), where the learned node embeddings are frozen and used as input features for downstream classifiers. For node classification, we train a logistic regression classifier with a 10%/10%/80% train/validation/test split. For hyperedge prediction, following (Kim et al., 2024a), we sample an equal number of negative hyperedges to match the ground-truth positives and split the data using a 60%/20%/20% ratio, after which a two-layer MLP classifier is trained for prediction. For each dataset, we report the average test accuracy and standard deviation over 20 random data splits, each evaluated with 5 different random initializations. Full evaluation details are provided in Appendix C.3.

Implementation details. For HiTeC, we adopt BERT-Base (Devlin et al., 2019) as the default text encoder and apply LoRA (Hu et al., 2022) for parameter-efficient fine-tuning. We further evaluate other text encoder backbones in Appendix D.2. We tune the s-walk length l and overlap size s in $\{1, 2, 3, 4, 5, 10\}$, the sampling ratio r in $\{10, 20, 30, 40, 50\}$, and λ_e and λ_s in $\{1, 2, 3, 4\}$. Additional implementation details are shown in Appendix C.4, and the code is available at `https://anonymous.4open.science/r/HiTeC-7D8A/`.

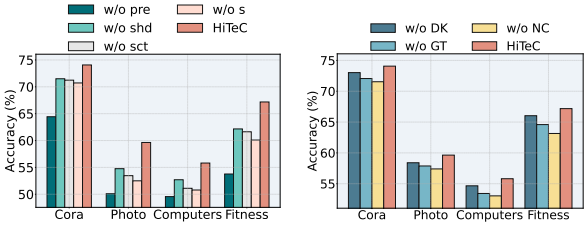
	Method	Citeseer	Cora	History	Photo	Computers	Fitness
Graph SSL	GraphCL + SE	59.63 ± 1.58	56.69 ± 1.99	62.86 ± 0.22	42.61 ± 0.21	34.45 ± 0.38	52.11 ± 0.43
	GraphCL + GIANT	57.58 ± 1.90	55.64 ± 1.50	62.08 ± 0.24	42.74 ± 0.22	34.65 ± 0.23	52.04 ± 0.34
	BGRL + SE	60.81 ± 1.45	59.72 ± 1.32	65.74 ± 0.30	44.60 ± 0.30	39.40 ± 0.46	55.66 ± 0.33
	BGRL + GIANT	58.68 ± 1.92	57.44 ± 1.39	63.99 ± 0.35	43.65 ± 0.25	37.00 ± 0.37	55.05 ± 0.25
	GraphMAE + SE	46.42 ± 2.75	50.33 ± 0.47	52.90 ± 9.01	41.52 ± 0.08	25.61 ± 0.11	40.65 ± 0.37
	GraphMAE + GIANT	50.11 ± 1.74	53.59 ± 1.07	56.20 ± 0.12	41.54 ± 0.06	25.57 ± 0.12	42.04 ± 0.04
HSSL	TriCL + SE	60.78 ± 1.00	58.79 ± 1.30	69.28 ± 0.32	47.11 ± 0.44	43.43 ± 0.66	45.46 ± 0.21
	TriCL + BERT	64.50 ± 1.08	62.67 ± 1.13	75.01 ± 0.22	51.25 ± 0.62	46.60 ± 0.90	48.97 ± 0.63
	TriCL + RoBERTa	64.48 ± 0.95	60.74 ± 1.01	70.62 ± 0.81	48.42 ± 0.67	43.82 ± 1.11	47.49 ± 0.60
	SE-HSSL + SE	57.04 ± 3.31	59.87 ± 1.83	68.59 ± 0.74	43.75 ± 2.01	36.53 ± 2.32	54.67 ± 2.00
	SE-HSSL + BERT	65.83 ± 1.21	64.30 ± 0.89	74.88 ± 0.42	51.80 ± 0.63	41.63 ± 2.21	56.92 ± 1.44
	SE-HSSL + RoBERTa	65.04 ± 1.04	62.76 ± 1.15	72.13 ± 1.07	49.41 ± 2.62	40.86 ± 3.99	54.79 ± 0.64
	HypeBoy + SE	59.98 ± 2.45	29.87 ± 0.62	OOM	OOM	OOM	OOM
	HypeBoy + BERT	61.98 ± 1.70	32.73 ± 1.71	OOM	OOM	OOM	OOM
	HypeBoy + RoBERTa	50.41 ± 2.88	30.84 ± 0.95	OOM	OOM	OOM	OOM
	VilLain (w/o feat.)	42.22 ± 1.72	53.72 ± 1.00	40.99 ± 3.06	33.94 ± 1.39	20.99 ± 0.91	42.04 ± 0.04
	HyperBERT	58.25 ± 3.04	47.03 ± 0.77	62.12 ± 0.19	41.46 ± 0.14	26.79 ± 0.20	OOM
	HiTeC	**68.69 ± 0.97**	**74.07 ± 0.88**	**79.81 ± 0.16**	**59.64 ± 0.16**	**55.81 ± 0.12**	**67.18 ± 0.19**

Table 2: Evaluation results on node classification task(% mean ± std). The **best** and second-best results are highlighted in **bold** and underline, respectively. "w/o feat." denotes methods without node features. OOM indicates that the method runs out of memory on a single 80GB GPU.

4.2 Main Results

In this section, we evaluate our approach on two representative hypergraph tasks: node classification and hyperedge prediction (see Appendix D.1).

Node classification evaluation. Table 2 presents the node classification results, revealing two key observations: (1) Graph-based SSL methods consistently underperform HSSL methods due to the loss of high-order relations caused by clique expansion. Even the TAG encoder GIANT, which jointly encodes text and pairwise structure, remains insufficient for modeling higher-order structures, highlighting the need for dedicated SSL methods for TAHGs. (2) HiTeC achieves the best overall performance across all datasets. Methods that neglect rich node features (e.g., VilLain) achieve limited performance, highlighting the importance of textual semantics in TAHG learning. HCL methods (e.g., TriCL and SE-HSSL) rely on structure-driven contrastive signals without semantic-aware augmentation, resulting in suboptimal performance. While HyperBERT jointly models textual and hypergraph information, it lacks multi-scale structural and semantic contrastive signals. In contrast, HiTeC consistently outperforms all baselines, surpassing the second-best method by 9.77% and 10.26% on Cora and Fitness, respectively. These improvements are mainly attributed to semantic-aware augmentations, which reduce perturbation noise, and the s-walk-based subgraph signals, which capture more informative high-order dependencies. Additional analysis on efficiency is provided in Appendix D.5.



(a) Key component ablation. (b) Text view ablation.

Figure 3: Ablation study. (a) Key component ablation: "w/o pre" removes structure-aware text encoder pre-training; "w/o shd" replaces semantic-aware hyperedge dropping with random drop; "w/o sct" removes the structure-contextualized text view; and "w/o s" removes the subgraph-level objective. (b) Text view ablation: "w/o DK", "w/o GT", and "w/o NC" remove domain knowledge, global topology, and neighbor context from the structure-contextualized text view, respectively.

4.3 Ablation Study

In this section, we conduct ablation studies on: (1) key components of HiTeC, (2) structure-contextualized text view components.

Impact of key components. We evaluate the contribution of key components in HiTeC. As shown in Figure 3(a), removing any module noticeably degrades performance. The most significant drop is observed when removing text encoder pre-training ('w/o pre"), resulting in a decrease of 13.41% on Fitness, highlighting the effectiveness of injecting topological information into the text encoder. The subgraph-level objective also plays a critical role: removing it ('w/o s") causes a decrease of 7.14% on Photo, demonstrating the benefit of modeling

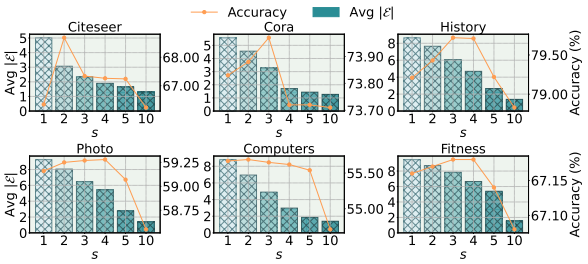


Figure 4: Sensitivity analysis of overlap size s. Avg$|\mathcal{E}|$ indicates the average number of sampled hyperedges.

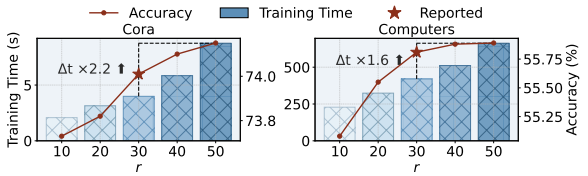


Figure 5: Training time and accuracy under different sampling ratios r. Δt indicates the relative speedup.

long-range dependencies inherent in TAHGs.

Impact of structure-contextualized text view components. We further evaluate the effectiveness of different components in the structure-contextualized text view in Figure 3(b). Overall, the complete view consistently outperforms all alternative variants. Specifically, removing neighbor context ("w/o NC") and global topology ("w/o GT") leads to pronounced performance degradation, underscoring the importance of incorporating meaningful structural signals into the original text.

4.4 Parameter Sensitivity Analysis

We investigate the impact of three key hyperparameters: the overlap size s, the subgraph sampling ratio r, and the walk length l (see Appendix D.3).

Overlap size s. Figure 4 illustrates how model performance varies with different s while fixing $l = 4$. Model accuracy first improves and then degrades as s increases, reflecting a trade-off between structural coherence and exploration range. Moderate overlap constraints (e.g., $s = 3$) encourage cohesive subgraphs while preserving contextual diversity, leading to the best performance in most cases. In contrast, $s = 1$ reduces to a vanilla random walk with weak structural correlation, while overly large overlap constraints (e.g., $s = 10$) greatly restrict candidate hyperedges, reducing contextual diversity and impairing performance.

Subgraph sampling ratio r. We analyze the effect of $r \in \{10, 20, 30, 40, 50\}$ on Cora and Computers with $s = 3$ and $l = 4$. As shown in Figure 5, model performance quickly improves and then plateaus as r increases, while training time grows substantially. On both datasets, $r = 30$ achieves accuracy comparable to $r = 50$ with 2.2× and 1.6× training speedups, respectively. These results suggest that sampling a small set of structurally central nodes is sufficient for subgraph-level learning. We therefore adopt $r = 30$ in the main experiments. Results on additional datasets are provided in Appendix D.4.

5 Related Work

Our work is closely related to self-supervised learning on hypergraphs and representation learning on text-attributed graphs (see Appendix E.1).

5.1 Self-supervised Learning on Hypergraphs

SSL has emerged as a promising paradigm for hypergraph representation learning without expensive labels. Most existing HSSL methods are based on contrastive learning. HyperGCL (Wei et al., 2022) introduces a learnable augmentation strategy, while TriCL (Lee and Shin, 2023) and SE-HSSL (Li et al., 2024) design hierarchical contrastive objectives across multiple granularities. Beyond contrastive methods, generative SSL has also been explored, with HypeBoy (Kim et al., 2024a) proposing a hyperedge-filling task. Other efforts explore alternative settings, such as feature-absent scenarios in VilLain (Lee et al., 2024). While effective for modeling hypergraphs, these methods largely overlook the rich textual semantics in TAHGs, resulting in suboptimal performance. HyperBERT (Bazaga et al., 2024) pioneers SSL on TAHGs by jointly optimizing text and hypergraph encoders through contrastive learning. However, its contrastive signals are limited to 1-hop hyperedge relations, restricting the modeling of long-range dependencies.

6 Conclusion

We propose HiTeC, an effective contrastive learning framework tailored for TAHGs. HiTeC adopts a two-stage paradigm that first injects topological information into text representations and then performs semantic-aware hierarchical contrastive learning on TAHGs. Specifically, two semantic-aware augmentation strategies are introduced to reduce perturbation noise, while an s-walk-based subgraph-level contrast captures long-range dependencies beyond node- and hyperedge-level scopes. Extensive experiments on six real-world datasets and multiple downstream tasks demonstrate the effectiveness and robustness of HiTeC.

Limitations

There are two limitations in our work: (1) The construction of the structure-contextualized text view relies on a handcrafted template. Although the proposed template demonstrates strong empirical effectiveness across multiple datasets, its optimal form may vary across different domains and tasks. Future work may explore more adaptive text-level augmentation. (2) The structure-aware pre-training stage introduces additional training overhead compared with conventional HCL methods. Nevertheless, HiTeC remains more efficient than prior TAHG methods such as HyperBERT due to its decoupled training paradigm (see Appendix D.5).

References

Sinan G Aksoy, Cliff Joslyn, Carlos Ortiz Marrero, Brenda Praggastis, and Emilie Purvine. 2020. Hypernetwork science via high-order hypergraph walks. *EPJ Data Science*, 9(1):16.

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

Adrián Bazaga, Pietro Lio, and Gos Micklem. 2024. HyperBERT: Mixing hypergraph-aware layers with language models for node classification on text-attributed hypergraphs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Coen Bron and Joep Kerbosch. 1973. Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM*.

Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olgica Milenkovic, and Inderjit S Dhillon. 2022a. Node feature extraction by self-supervised multi-scale neighborhood prediction. In *International Conference on Learning Representations(ICLR)*.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Xiangfei Fang, Boying Wang, Chengying Huan, Shaonan Ma, Heng Zhang, and Chen Zhao. 2025. Hyperkan: Hypergraph representation learning with kolmogorov-arnold networks. *arXiv preprint arXiv:2503.12365*.

Yi Fang, Dongzhe Fan, Daochen Zha, and Qiaoyu Tan. 2024. Gaugllm: Improving graph contrastive learning for text-attributed graphs with large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 747–758.

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

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

C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. 1998. Citeseer: an automatic citation indexing system. In *Proceedings of the Third ACM Conference on Digital Libraries*, page 89–98, New York, NY, USA. Association for Computing Machinery.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2024. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *International Conference on Learning Representations(ICLR)*.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. 2022. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 594–604.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Sunwoo Kim, Fanchen Bu, Minyoung Choe, Jaemin Yoo, and Kijung Shin. 2023. How transitive are real-world group interactions?-measurement and reproduction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1132–1143.

Sunwoo Kim, Shinhwan Kang, Fanchen Bu, Soo Yong Lee, Jaemin Yoo, and Kijung Shin. 2024a. Hypeboy: Generative self-supervised representation learning on hypergraphs. In *International Conference on Learning Representations(ICLR)*.

Sunwoo Kim, Soo Yong Lee, Yue Gao, Alessia Antelmi, Mirko Polato, and Kijung Shin. 2024b. A survey on hypergraph neural networks: an in-depth

and step-by-step guide. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6534–6544.

Dongjin Lee and Kijung Shin. 2023. I'm me, we're us, and i'm us: Tri-directional contrastive learning on hypergraphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 8456–8464.

Geon Lee, Soo Yong Lee, and Kijung Shin. 2024. Villain: Self-supervised learning on homogeneous hypergraphs without features via virtual label propagation. In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 594–605.

Fan Li, Xiaoyang Wang, Dawei Cheng, Wenjie Zhang, Ying Zhang, and Xuemin Lin. 2024. Hypergraph self-supervised learning with sampling-efficient signals. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Kazuki Nakajima and Takeaki Uno. 2025. Inference and visualization of community structure in attributed hypergraphs using mixed-membership stochastic block models. *Social Network Analysis and Mining*, 15(1):5.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Giulia Preti, Gianmarco De Francisci Morales, and Francesco Bonchi. 2024. Hyper-distance oracles in hypergraphs. *The VLDB Journal*, 33(5):1333–1356.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective classification in network data articles. *AI Magazine*, 29:93–106.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Veličković, and Michal Valko. 2022. Large-scale representation learning on graphs via bootstrapping. In *International Conference on Learning Representations(ICLR)*.

Yanbang Wang and Jon Kleinberg. 2024. From graphs to hypergraphs: Hypergraph projection and its remediation. *arXiv preprint arXiv:2401.08519*.

Yaoke Wang, Yun Zhu, Wenqiao Zhang, Yueting Zhuang, Liyunfei Liyunfei, and Siliang Tang. 2024. Bridging local details and global context in text-attributed graphs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Tianxin Wei, Yuning You, Tianlong Chen, Yang Shen, Jingrui He, and Zhangyang Wang. 2022. Augmentations in hypergraph contrastive learning: Fabricated and generative. *Advances in neural information processing systems*, 35:1909–1922.

Yucheng Wu, Leye Wang, Xiao Han, and Han-Jia Ye. 2024. Graph contrastive learning with cohesive subgraph awareness. In *Proceedings of the ACM Web Conference 2024*, pages 629–640.

Lianghao Xia, Chao Huang, and Chuxu Zhang. 2022. Self-supervised hypergraph transformer for recommender systems. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2100–2109.

Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, and 1 others. 2023. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. *Advances in Neural Information Processing Systems*, 36:17238–17264.

Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. 2021. Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems*, 34:28798–28810.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823.

Shan Yu, Hongdian Yang, Hiroyuki Nakahara, Gustavo S Santos, Danko Nikolić, and Dietmar Plenz. 2011. Higher-order interactions characterized in cortical activity. *Journal of neuroscience*, 31(48):17514–17526.

Song Kyung Yu, Da Eun Lee, Yunyong Ko, and Sang-Wook Kim. 2025. Hygen: Regularizing negative hyperedge generation for accurate hyperedge prediction. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 1500–1504.

Peiyan Zhang, Chaozhuo Li, Liying Kang, Feiran Huang, Senzhang Wang, Xing Xie, and Sunghun Kim. 2024. High-frequency-aware hierarchical contrastive selective coding for representation learning on text attributed graphs. In *Proceedings of the ACM Web Conference 2024*, pages 4316–4327.

Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1:43–52.

Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023. Learning on large-scale text-attributed graphs via variational inference. In *International Conference on Learning Representations(ICLR)*.

Appendix

A Additional Template Details

As shown in Table 3, the structure-contextualized view follows a fixed template that concatenates contextual components into a single sequence: $\tilde{t}_{1,i} = [\texttt{[DK]} \,||\, \texttt{[GT]} \,||\, \texttt{[NC]} \,||\, t_i]$. All components are deterministically constructed from the hypergraph structure and node attributes, without external knowledge. The number of neighboring nodes in [NC] is limited to 10 to control sequence length and ensure stable training. This design enriches structural and semantic context while preserving semantic consistency.

B Complexity Analysis

We analyze the computational complexity of the forward phase of HiTeC during training.

Text encoder pretraining. Let l_p denote the number of layers in the PLM, $\bar{L}$ the average token length, and d the hidden dimension. The PLM encoding costs $\mathcal{O}\big(l_p|\mathcal{V}|(\bar{L}d^2 + \bar{L}^2 d)\big)$. The triplet margin objective costs $\mathcal{O}(|\mathcal{V}|\, d)$. Therefore, the overall complexity is $\mathcal{O}\big(l_p|\mathcal{V}|(\bar{L}d^2 + \bar{L}^2 d) + |\mathcal{V}|d\big)$.

Hypergraph encoder pretraining. Let l_h be the number of HNN layers and δ_e the average hyperedge size. The hypergraph propagation costs $\mathcal{O}\big(l_h(|\mathcal{E}|\delta_e d + |\mathcal{V}|d^2)\big)$. For contrastive learning with m sampled negatives (a constant), the node-, hyperedge-, and subgraph-level objectives incur costs of $\mathcal{O}(m|\mathcal{V}|d)$, $\mathcal{O}(m|\mathcal{E}|d)$, and $\mathcal{O}(mKd)$, respectively, where K denotes the number of sampled subgraphs. Thus, the total complexity is $\mathcal{O}(l_h(|\mathcal{E}|\delta_e d + |\mathcal{V}|d^2) + m(|\mathcal{V}| + |\mathcal{E}| + K)d)$.

C Additional Experimental Setup

C.1 Datasets

Since no publicly available benchmark datasets are specifically designed for Text-Attributed Hypergraphs (TAHGs), we construct six datasets spanning two domains: academic co-citation networks and E-commerce platforms. Basic statistics are summarized in Table 1.

Co-citation hypergraphs. We construct two academic hypergraphs, Citeseer (Giles et al., 1998) and Cora (Sen et al., 2008), from standard citation networks, where each node represents a scientific publication. Node textual attributes, including titles and abstracts, are adopted from prior works (He et al., 2024; Wang et al., 2024). To capture high-order relations, we define each hyperedge as the set of papers co-citing the same reference.

E-commerce hypergraphs. We further construct four e-commerce hypergraphs from the Amazon product network (Ni et al., 2019): Book-History, Ele-Photo, Ele-Computers, and Sports-Fitness. Each node represents a product, while edges reflect frequent co-purchase or co-browsing behavior. Node textual attributes are adopted from (Yan et al., 2023). Specifically, Book-History uses book titles and descriptions; Ele-Computers and Ele-Photo use high-rated reviews and product summaries; Sports-Fitness only uses product titles. To induce high-order structures, we adopt a max-clique-based reconstruction strategy (Bron and Kerbosch, 1973; Wang and Kleinberg, 2024), where hyperedges are formed from maximal cliques in the original graph.

C.2 Baselines

We compare our method against eight representative self-supervised learning (SSL) baselines, covering graph-based, hypergraph-based, and text-attributed hypergraph (TAHG)-based paradigms.

Graph-based SSL. In this work, we adopt three widely used graph-based SSL frameworks: two contrastive learning methods, GraphCL (You et al., 2020) and BGRL (Thakoor et al., 2022), and one generative method, GraphMAE (Hou et al., 2022). To adapt these methods to hypergraphs, we apply clique expansion following the protocol in (Lee and Shin, 2023). For textual feature extraction, we employ shallow encoders (SE) implemented via Skip-Gram (Mikolov et al., 2013), as well as GIANT (Chien et al., 2022a), a graph-agnostic text encoder designed for raw textual attributes.

Hypergraph-based SSL. We consider four state-of-the-art self-supervised learning methods specifically designed for hypergraphs. These include two contrastive approaches, TriCL (Lee and Shin, 2023) and SE-HSSL (Li et al., 2024); one generative framework, HypeBoy (Kim et al., 2024a); and VilLain (Lee et al., 2024), which is tailored for feature-absent hypergraph settings. For a fair comparison, all methods adopt the same shallow encoder (SE). In addition, we further evaluate these baselines by replacing the SE with two pre-trained language models, BERT-Base (Devlin et al., 2019) and RoBERTa-Base (Liu et al., 2019).

TAHG-based SSL. We further include HyperBERT (Bazaga et al., 2024), to the best of our

Component	Description	Template
[DK]	Domain Knowledge	You are in an Academic / E-commerce / ... hypergraph. Each node represents a Paper / Product / ... , and hyperedges capture Co-citation / Co-purchasing / ... relationships among groups of nodes.
[GT]	Global Topology	Your node degree is value , indicating a Sparse / Dense connectivity pattern. You participate in the following hyperedges: Hyperedge IDs .
t	Raw Text	Your title is Title . Your abstract or description is Abstract / Description .
[NC]	Neighbor Context	Below are brief descriptions of your neighboring nodes. For neighbor 1, the title is: Neighbor 1 Title and the abstract or description is : Neighbor 1 Abstract / Description . For neighbor 2 ...

Table 3: Structure-contextualized text view template.

knowledge, the only existing SSL method specifically designed for text-attributed hypergraphs. HyperBERT jointly models textual semantics and hypergraph structure using a Transformer-based architecture, in which hypergraph-aware layers are introduced to integrate structural signals with text representations. However, this joint modeling paradigm incurs substantial computational and memory overhead, limiting its scalability to large hypergraphs, as validated in our experiments.

C.3 Evaluation Protocol

We evaluate our method on two standard hypergraph learning tasks: node classification and hyperedge prediction. After pretraining with HiTeC, we adopt a linear evaluation protocol (Lee and Shin, 2023; Li et al., 2024; Kim et al., 2024a), where the pretrained node embeddings are frozen and used as input features for downstream classifiers.

Node classification. Following (Lee and Shin, 2023; Li et al., 2024), we randomly split each dataset into 10%/10%/80% for training, validation, and testing. Then, a simple linear classifier is trained on top of the fixed node embeddings using ℓ_2-regularized logistic regression. We report the average test accuracy and standard deviation over 20 random splits, each with 5 different initializations.

Hyperedge prediction. Following (Kim et al., 2024a), we need to generate negative hyperedge samples to evaluate a model on the hyperedge prediction task. In our experiments, we adopt Clique Negative Sampling (CNS) (Yu et al., 2025), a widely used strategy for constructing negative hyperedge samples. Specifically, for a given hyperedge e, we replace a node $u \in e$ with a node $v \notin e$ such that v is connected to all other nodes in e, i.e., $(e \setminus \{u\}) \cup \{v\}$. Then, a two-layer MLP is trained on the fixed node embeddings using the 60%/20%/20% data split. Same as the node classification task, we report the average test accuracy and standard deviation over 20 random splits, each with 5 initializations.

C.4 Implementation Details

All experiments are conducted on a single NVIDIA A800 PCIe GPU with 80GB of memory. For all baselines, we adopt their official implementations with default settings. For HiTeC, we adopt BERT-Base (Devlin et al., 2019) as the default text encoder and apply LoRA (Hu et al., 2022) to the query and value matrices of the self-attention layers for parameter-efficient fine-tuning, while keeping the PLM backbone frozen. Unless otherwise specified, the LoRA configuration is set to $r = 16$, $\alpha = 32$, and dropout $= 0.05$. We additionally evaluate other text encoder backbones, including DistilBERT (Sanh et al., 2019), RoBERTa-Base (Liu et al., 2019), and DeBERTa-Base (He et al., 2020). The text encoder is trained for 5 epochs on small datasets (Citeseer, Cora), 4 epochs on medium datasets (History, Photo), and 3 epochs on large datasets (Computers, Fitness), using a learning rate of 2e-5. The hypergraph encoder is implemented as a single-layer module. We tune the contrastive loss weights λ_e and λ_s over $\{1, 2, 3, 4\}$. The overlap threshold s and walk length l are tuned from $\{1, 2, 3, 4, 5, 10\}$, the subgraph sampling ratio r in $\{10, 20, 30, 40, 50\}$. To ensure a fair comparison, we tune the learning rate from $\{0.01, 0.001, 1e\text{-}4, 5e\text{-}4\}$ and weight decay from $\{0, 1e\text{-}5\}$ for all methods.

	Method	Citeseer	Cora	History	Photo	Computers	Fitness
Graph SSL	GraphCL + SE	69.48 ± 2.07	59.05 ± 2.16	65.65 ± 1.06	67.48 ± 1.18	68.66 ± 1.33	61.22 ± 1.34
	GraphCL + GIANT	72.70 ± 2.06	55.48 ± 1.96	63.02 ± 1.61	68.78 ± 1.16	70.87 ± 0.87	63.02 ± 0.01
	BGRL+SE	70.08 ± 2.38	62.45 ± 2.48	64.47 ± 1.18	64.76 ± 1.23	64.66 ± 1.40	60.05 ± 1.05
	BGRL + GIANT	72.61 ± 1.99	66.22 ± 2.38	65.13 ± 2.09	65.65 ± 1.10	67.67 ± 0.99	60.92 ± 0.34
	GraphMAE + SE	71.71 ± 1.78	64.85 ± 1.66	59.81 ± 0.27	66.33 ± 1.58	64.83 ± 1.72	55.79 ± 0.18
	GraphMAE + GIANT	72.25 ± 1.62	66.30 ± 2.19	64.19 ± 0.23	64.83 ± 0.20	66.10 ± 1.20	59.21 ± 0.88
HSSL	TriCL + SE	68.06 ± 1.75	62.43 ± 2.41	67.18 ± 0.24	69.21 ± 0.19	70.63 ± 0.14	63.87 ± 0.30
	TriCL + BERT	71.36 ± 1.78	64.02 ± 2.33	70.18 ± 0.31	71.71 ± 0.21	73.20 ± 0.17	65.49 ± 0.40
	TriCL + RoBERTa	72.38 ± 1.37	66.49 ± 1.93	68.34 ± 0.58	70.35 ± 0.28	72.20 ± 0.27	63.49 ± 0.17
	SE-HSSL + SE	64.80 ± 2.21	65.57 ± 1.88	63.49 ± 1.98	65.42 ± 0.74	66.44 ± 1.09	62.52 ± 0.18
	SE-HSSL + BERT	72.29 ± 1.74	65.66 ± 2.25	67.93 ± 0.44	71.27 ± 0.18	70.68 ± 0.91	66.00 ± 0.69
	SE-HSSL + RoBERTa	73.09 ± 1.56	65.86 ± 1.94	66.95 ± 0.53	69.65 ± 0.20	71.19 ± 0.16	65.72 ± 0.70
	HypeBoy + SE	71.93 ± 2.04	65.17 ± 1.66	OOM	OOM	OOM	OOM
	HypeBoy + BERT	74.56 ± 2.10	**67.38 ± 1.89**	OOM	OOM	OOM	OOM
	HypeBoy + RoBERTa	73.47 ± 2.10	66.51 ± 1.87	OOM	OOM	OOM	OOM
	VilLain (w/o feat.)	74.25 ± 1.90	63.25 ± 2.57	63.18 ± 0.35	65.02 ± 0.23	65.85 ± 0.28	62.92 ± 0.26
	HyperBERT	71.84 ± 2.03	64.48 ± 2.10	69.52 ± 0.20	69.34 ± 0.18	70.40 ± 0.15	OOM
	HiTeC	**75.22 ± 1.87**	65.91 ± 2.32	**71.83 ± 0.21**	**72.56 ± 0.17**	**73.91 ± 0.14**	**67.32 ± 0.09**

Table 4: Evaluation results on hyperedge prediction task(% mean ± std). The **best** and second-best results are highlighted in **bold** and underline, respectively. "w/o feat." denotes methods without node features. OOM indicates that the method runs out of memory on a single 80GB GPU.

D Additional Experiment Results

D.1 Hyperedge Prediction Evaluation

We further evaluate all models on the hyperedge prediction task, with results presented in Table 4. The key observations are as follows: (1) Graph-based SSL methods remain less effective. Their performance is still hindered by the loss of high-order structure due to clique expansion. Although GIANT achieves slight improvements, probably due to its pseudo-labeling strategy for edge-level supervision, it still underperforms compared to HSSL methods enhanced with pre-trained language models. (2) HiTeC achieves the best performance on 5 out of 6 datasets, demonstrating strong generalizability between tasks. TriCL and SE-HSSL underperform due to their limited capacity to capture long-range dependencies. HypeBoy exhibits competitive results on small-scale hypergraphs, benefiting from its hyperedge-filling strategy, but fails to scale to larger datasets, indicating poor scalability. Overall, HiTeC exceeds the second-best method by 1.65% and 1.15% on History and Fitness, respectively, demonstrating its robustness and scalability across diverse hypergraph scenarios.

D.2 Impact of Text Encoder Backbones

We evaluate different text encoder backbones, including DistilBERT (Sanh et al., 2019), BERT-Base (Devlin et al., 2019), RoBERTa-Base (Liu et al., 2019), and DeBERTa-Base (He et al., 2020), as shown in Table 5. Among them, BERT-Base achieves the best overall performance, followed by DistilBERT. The results indicate that backbone selection can noticeably affect performance across datasets. Nevertheless, our framework remains compatible with diverse text encoders. Based on the overall results, we adopt BERT as the default backbone in the main experiments.

Dataset	DistilBERT	BERT	RoBERTa	DeBERTa
Citeseer	**67.54 ± 1.49**	66.38 ± 1.73	65.09 ± 1.26	64.83 ± 4.65
Cora	**74.55 ± 0.02**	74.07 ± 0.88	72.32 ± 0.91	70.41 ± 2.24
History	78.05 ± 0.18	**79.81 ± 0.16**	74.77 ± 0.24	76.20 ± 0.12
Photo	56.90 ± 0.31	**59.64 ± 0.16**	54.15 ± 0.40	52.54 ± 0.08
Computers	50.64 ± 0.25	**55.81 ± 0.12**	52.36 ± 0.21	51.48 ± 0.56
Fitness	62.97 ± 0.08	**67.18 ± 0.19**	64.98 ± 0.78	65.29 ± 0.14

Table 5: Different text encoder backbones.

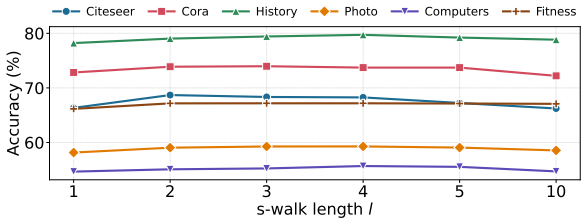


Figure 6: Sensitivity analysis of walk length l.

D.3 Parameter Sensitivity Analysis of s-walk length l

To investigate the effect of walk length on HiTeC, we report the results under different $l \in \{1, 2, 3, 4, 5, 10\}$ while fixing $s = 3$. As shown in Figure 6, model performance generally first improves and then degrades as l increases. This trend suggests that moderate walk lengths help capture richer high-order structural dependencies by exploring broader hypergraph contexts. In contrast,

short walks (e.g., $l = 1$) provide limited structural context, while excessively long walks (e.g., $l = 10$) may introduce noisy or weakly related hyperedges, both leading to suboptimal performance. Overall, $l = 4$ achieves a favorable balance between structural exploration and semantic consistency.

D.4 Parameter Sensitivity Analysis of Subgraph Sampling Ratio r

We further perform a sensitivity analysis by varying the subgraph sampling ratio $r \in \{10, 20, 30, 40, 50\}$ with s fixed as 3 and l fixed as 4. As shown in Figure 7, model performance generally increases rapidly at first and then plateaus, while training time grows sharply with larger r. Notably, on most datasets, setting $r = 30$ achieves comparable accuracy to $r = 50$, while significantly reducing training time. For example, on Cora and Computers, $r = 30$ leads to 2.2× and 1.6× speedups, respectively, with only marginal drops in accuracy. These results indicate that selecting a small number of structurally central nodes based on degree ranking enables efficient subgraph-level representation learning without sacrificing performance. In our main experiments, we report the results with $r = 30$, which achieves a trade-off between efficiency and performance.

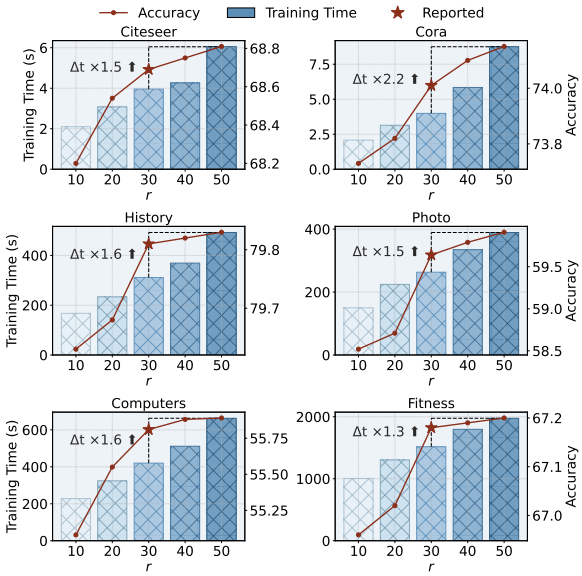


Figure 7: Training time and accuracy under different sampling ratios r. Bar plots report training time (in seconds), while line plots show the corresponding accuracy. We highlight the relative speedup Δt of $r = 30$ over $r = 50$. s and l are fixed to 3 and 4, respectively. All main results are selected with $r = 30$.

Method	Citeseer	Cora	History	Photo	Computers
HyperBERT	4h 30m	5h 22m	29h 42m	33h 40m	56h 04m
HiTeC	2h 22m	2h 56m	18h 31m	19h 55m	28h 44m

Table 6: Comparison of text encoder pretraining time between HyperBERT and HiTeC.

D.5 Efficiency Evaluation

We evaluate training time to assess the efficiency of HiTeC. Specifically, we first compare the text encoder pre-training time between HiTeC and HyperBERT, and then compare the hypergraph encoder pre-training time of HiTeC with two representative HCL methods (TriCL and SE-HSSL).

Text encoder pre-training time evaluation. Table 6 compares the text encoder pre-training time between HyperBERT and HiTeC. For HyperBERT, we follow the official implementation and report the converged training time under its recommended training settings. For HiTeC, we adopt the settings described in the Appendix C.4. From the results, HiTeC consistently achieves substantially lower pre-training time across all datasets. In particular, HiTeC achieves nearly 2× faster pre-training on large datasets such as Computers (56h 04m → 28h 44m), while still achieving superior downstream performance. This efficiency gain mainly comes from the decoupled pre-training paradigm of HiTeC, whereas HyperBERT relies on a joint training scheme for text and hypergraph modeling.

Hypergraph encoder pretraining time evaluation. We further compare the hypergraph encoder pre-training time of HiTeC with two representative hypergraph-based HCL methods. As shown in Figure 8, training time is plotted on a logarithmic scale for clearer visualization, where HiTeC achieves performance comparable to SE-HSSL while being at least 3× faster than TriCL on large-scale datasets. This efficiency gain mainly stems from two factors: (1) The subgraph-level contrastive objective operates on a small subset of sampled subgraphs (see Appendix D.4); and (2) The preprocessing steps are executed once and reused across multiple training runs. Overall, HiTeC strikes a favorable trade-off between training efficiency and accuracy.

E Additional Related Work

E.1 Representation Learning on TAGs

Since hypergraphs can be transformed into graphs via clique expansion, our work is closely related to representation learning on text-attributed graphs

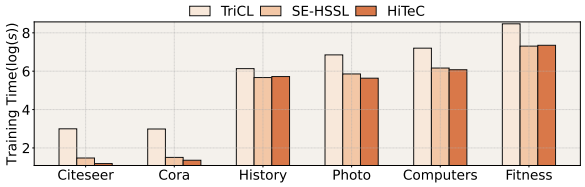


Figure 8: Comparison of hypergraph encoder pretraining time among HiTeC, SE-HSSL, and TriCL. $\log(\cdot)$ denotes the natural logarithm.

(TAGs). Recent TAG methods leverage pre-trained language models to generate numerical features, typically in a cascade (He et al., 2024; Wang et al., 2024) or nested (Yang et al., 2021; Zhao et al., 2023) manner. While most existing TAG methods are supervised, emerging efforts explore self-supervised learning (SSL) to reduce annotation costs. GIANT (Chien et al., 2022a) makes an early attempt by aligning textual and structural features via an extreme multi-label classification objective. More recently, GAugLLM (Fang et al., 2024) proposes an augmentation framework that utilizes large language models (LLMs) to perturb both node features and graph structure, while HASHCODE (Zhang et al., 2024) formulates a multi-objective contrastive learning strategy tailored for TAGs. However, hypergraphs fundamentally differ from ordinary graphs in their high-order structural modeling. Converting hypergraphs into ordinary graphs via clique expansion (Bron and Kerbosch, 1973) inevitably incurs substantial structural information loss (Lee and Shin, 2023; Li et al., 2024), limiting their effectiveness on TAHGs and motivating the need for dedicated HSSL approaches.