Mitigating Degree Bias in Hypergraphs via Attribute-as-Structure Approach

Ryusei Nishide, Makoto Miwa
Toyota Technological Institute, Nagoya, Japan
{sd25428, makoto-miwa}@toyota-ti.ac.jp

Abstract

Entity representation learning on hypergraphs is hindered by degree bias, where nodes with sparse connections suffer from limited structural information for aggregation. Prevailing "attribute-as-feature" approaches, which treat rich textual attributes (e.g., titles, abstracts, keywords) merely as node features, fail to address this structurally rooted problem as they do not create new aggregation pathways. To overcome this limitation, we propose a novel "attribute-as-structure" approach specifically designed for heterogeneous hypergraphs. Our approach integrates attributes directly into the hypergraph topology as distinct node types, creating new structural pathways to enrich sparsely connected entities while preserving semantic distinctiveness within complex many-to-many hyperedge interactions. We introduce an entity-attribute aware learning framework featuring two key innovations: (1) a specialized heterogeneous hypergraph encoder with dual attention mechanisms—self-attention for entity-entity relationships and cross-type attention for entity-attribute relevance, and (2) Attribute-Attentive Contrastive Learning (AACL), a novel objective that dynamically weighs attribute importance while explicitly aligning entity representations with their structural attributes. Experiments on multiple hypergraph datasets demonstrate consistent improvements in node classification performance, with particularly significant gains for structurally sparse nodes, demonstrating the effectiveness of our approach for degree bias mitigation.

1 Introduction

Entity representation learning on hypergraphs has become fundamental in various applications including knowledge graphs, recommendation systems, and information retrieval (Xia et al., 2022; Antelmi et al., 2024). Many state-of-the-art methods (Lee and Shin, 2023; Kim et al., 2024; Dong et al., 2020) for this task learn node representations by leveraging hypergraph topology, by aggregating information from connected nodes and hyperedges. However, a fundamental limitation shared by these structure-aware approaches is their performance on nodes with sparse connections. This practical issue is a manifestation of degree bias (Subramonian et al., 2024), where low-degree nodes are uninformed due to the limited structural information for aggregation, hindering model performance.

To compensate for this structural weakness, leveraging entity attribute information becomes a natural approach. However, its most common utilization, the "attribute-as-feature" approach (Bazaga et al., 2024), faces a fundamental limitation rooted in the mechanism of information aggregation. In graph-based models, information aggregation occurs through pathways defined by the topology. While rich features enhance a node's initial self-information, they do not create new pathways for information aggregation. As a result, this approach is inherently limited in its ability to enrich the structural context for low-degree nodes and thus cannot fully address a structurally rooted problem like degree bias.

However, developing an effective "attribute-as-structure" approach for hypergraphs is non-trivial. There are two major challenges to be addressed: **(1) Semantic Preservation Challenge**: How can we preserve the distinct semantic roles of entities and attributes when they are simultaneously involved in higher-order hyperedge interactions? Unlike pairwise relationships in standard graphs, hyperedges involve complex many-to-many relationships that make it difficult to maintain semantic distinctiveness. **(2) Learning Objective Challenge**: What learning objectives can effectively capture both structural patterns and entity-attribute relationships while avoiding feature conflicts that may arise when rich attribute information is integrated with sparse structural signals?

In this paper, we propose a novel entity-attribute

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 1784–1801

aware learning framework for heterogeneous hypergraphs that realizes an "attribute-as-structure" approach to address these challenges. Our framework introduces specialized mechanisms to handle the unique complexities of hypergraph structures while maintaining the semantic integrity of both entities and attributes.

Our main contributions are:

- **Heterogeneous Node Type Construction**: We construct hypergraphs where entity attributes become distinct node types, creating additional aggregation pathways for sparse entities rather than treating attributes as mere features.
- **Type-Aware Attention Architecture**: We design dual attention mechanisms—self-attention for entity-entity relationships and cross-type attention for entity-attribute relevance—preserving semantic distinctiveness in many-to-many hyperedge interactions.
- **Attribute-Attentive Contrastive Learning (AACL)**: We propose a novel AACL objective that dynamically weighs attribute importance while aligning entity representations with structural attributes.
- **Degree Bias Mitigation**: We demonstrate consistent improvements across multiple datasets (up to +1.64% overall, +2.10% for sparse nodes), with particularly significant gains for structurally sparse nodes, demonstrating the effectiveness of our approach for degree bias mitigation.

2 Related Work

2.1 Hypergraph Representation Learning

Researchers have developed various methods for hypergraph representation learning (Antelmi et al., 2024). Early approaches like HGNN (Feng et al., 2019) and HyperGCN (Yadati et al., 2019) introduced spectral and convolutional operations, which were later generalized by methods like UniGNN (Huang and Yang, 2021) and HNHN (Dong et al., 2020). More recently, self-supervised learning has become prominent, with methods exploring contrastive (TriCL (Lee and Shin, 2023)), generative (HypeBoy (Kim et al., 2024)), and augmentation-based (HyperGCL (Wei et al., 2022)) objectives. However, these methods typically treat all nodes uniformly, without distinguishing between entities and their attributes.

2.2 Scope of This Work and Relation to Other Approaches

Our work is situated within the prevalent aggregation-based approach for hypergraph representation learning and aims to address its inherent limitations regarding attribute information and structural sparsity.

Other emerging approaches are beyond our scope. For instance, Hypergraph Transformers (Liu et al., 2024) compute all-pairs self-attention, bypassing explicit neighborhood aggregation. While promising, this approach moves away from explicit topological structure utilization rather than addressing local structural sparsity through targeted aggregation pathways. Our focus remains on improving prevalent aggregation-based frameworks.

2.3 Handling Attributes in Aggregation-Based Frameworks

Within aggregation-based frameworks, three main strategies have emerged for incorporating attributes. The most common **attribute-as-feature** approach treats attributes as node features processed alongside structural information (He et al., 2024; Tang et al., 2024; Bazaga et al., 2024). While conceptually simple, this approach cannot create new aggregation pathways and thus fails to address the structural root of degree bias.

A second strategy uses **Large Language Models (LLMs)** to guide hypergraph construction (Chu et al., 2024), outsourcing semantic interpretation to external models. While effective, this approach faces computational intensity and limited interpretability.

A third approach explores **attribute-as-structure** in standard graphs, where attributes become separate structural components (Wang et al., 2025; Tan et al., 2024). For example, CONN (Tan et al., 2024) selectively diffuses messages from both nodes and attribute categories. Related work addressing degree bias includes DegFairGT (Hoang et al., 2025), which learns structural augmentation during training, and GraphPatcher (Ju et al., 2023), which applies test-time virtual node generation. These methods fundamentally differ from ours in three aspects: (1) they operate on pairwise graphs rather than hypergraphs with many-to-many relationships, (2) they employ probabilistic connection estimation or test-time patching, whereas our approach deterministically converts metadata into distinct

node types based on an explicit schema, and (3) they augment existing entity structure, whereas we introduce a new schema that integrates attributes as first-class structural components.

Our work extends the attribute-as-structure approach to heterogeneous hypergraphs. Instead of treating attributes as features or relying on an external LLM to design the structure, our "attribute-as-structure" approach integrates attributes directly into the topology as first-class nodes. We posit that this self-contained approach offers a more direct and efficient way to solve the structural sparsity issues discussed in the introduction, without departing from the well-understood aggregation-based learning framework.

3 Methodology

Our "attribute-as-structure" approach addresses the root cause of degree bias by integrating attributes directly into the hypergraph topology as distinct node types. This creates new structural pathways for sparse entities, and is realized through two key innovations: **(1)** a heterogeneous hypergraph encoder with specialized attention to handle different node and edge types, and **(2)** an Attribute-Attentive Contrastive Learning (AACL) loss to dynamically weigh attribute importance. Together, these components effectively integrate structural patterns with attribute information.

The remainder of this section first provides an overview of the framework in Section 3.1, then details the encoder (Section 3.2) and the learning objectives including our AACL loss (Section 3.3).

3.1 Overall Framework

Fig. 1 illustrates the complete framework of our proposed method. The framework consists of three main stages:

1. **Heterogeneous hypergraph construction**: We construct a heterogeneous hypergraph where entity nodes (e.g., papers) and attribute nodes (e.g., titles, abstracts, keywords) are explicitly distinguished as different node types. Heterogeneous hyperedges capture both the relationships between entities and their associated attributes, as well as higher-order relationships among entities themselves.

2. **Heterogeneous hypergraph encoding**: Our specialized encoder processes the hypergraph through multiple steps: (i) initial representation, where entity nodes are initialized with structural features and attribute nodes with pre-trained domain-specific encoders, both enriched with learnable type embeddings; (ii) message passing via specialized attention mechanisms that distinguish between entity-entity relationships and entity-attribute relationships; and (iii) type-specific transformations to maintain the distinct representational spaces.

3. **Contrastive learning**: The model is trained using a dual-objective function that combines structural pattern learning and entity-attribute relationship learning. The structural component leverages the TriCL framework to capture hypergraph topology, while our novel AACL loss explicitly optimizes the relationships between entities and their associated attributes.

Our framework operates on a heterogeneous hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T}, \mathcal{R})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of hyperedges, $\mathcal{T}$ is the set of node types (including entity types and attribute types), and $\mathcal{R}$ is the set of relation types. For each node $v \in \mathcal{V}$, we denote its type as $\tau(v) \in \mathcal{T}$, and for each hyperedge $e \in \mathcal{E}$, we denote its relation type as $\phi(e) \in \mathcal{R}$.

3.2 Heterogeneous Hypergraph Encoder

Our encoder processes the heterogeneous hypergraph via three main steps: type-specific initial representation of nodes, bidirectional message passing between nodes and hyperedges, and a final type-specific output transformation.

3.2.1 Initial Representation through Type-specific Embeddings

To capture the distinct characteristics of different node types in our heterogeneous hypergraph, we first transform the input features of each node type to a common embedding dimension d using type-specific feature processors. We then generate three specialized types of embeddings:

Node type embeddings $\mathbf{e}_t \in \mathbb{R}^d$: Learnable vectors that encode the semantic properties specific to each node type (e.g., entity, title, abstract).

Entity-attribute role embeddings $\mathbf{e}_k \in \mathbb{R}^d$: Vectors that distinguish between the functional roles of entities versus attributes in the hypergraph.

Edge type embeddings $\mathbf{e}_r \in \mathbb{R}^d$: Vectors representing the different relationship types (e.g., entity-to-entity connections versus entity-attribute connections).

For the initial node features $\mathbf{X}_t \in \mathbb{R}^{n_t \times d_t}$ (which for attribute nodes come from pre-trained encoders like BERT), we compute the initial node represen-

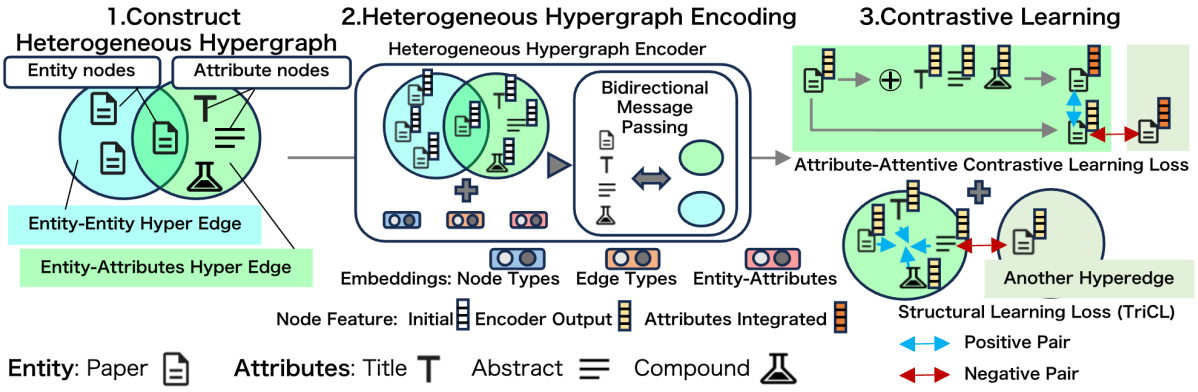

Figure 1: Overall framework of our proposed entity-attribute aware representation learning method for heterogeneous hypergraphs. The framework processes heterogeneous hypergraphs through three key stages: heterogeneous hypergraph construction, heterogeneous hypergraph encoding with specialized attention mechanisms, and entity-attribute aware learning with our proposed AACL loss.

tations by applying a non-linear transformation and explicitly combining these learned embeddings:

$$\mathbf{H}_t^{(0)} = \text{ELU}(\text{LN}(\mathbf{W}_t\mathbf{X}_t + \mathbf{b}_t)) + \mathbf{e}_t + \mathbf{e}_k \quad (1)$$

Here, $\mathbf{H}_t^{(0)} \in \mathbb{R}^{n_t \times d}$ represents the initial representation of nodes of type t, where n_t is the number of nodes of that type. The learnable weight matrix $\mathbf{W}_t \in \mathbb{R}^{d \times d_t}$ projects features to dimension d, and we apply ELU activation (Clevert et al., 2016) after Layer Normalization (Ba et al., 2016) for training stability. Finally, we add both the node type embedding $\mathbf{e}_t$ and the entity-attribute role embedding $\mathbf{e}_k$ to incorporate type-specific information. The embedding vectors are added rather than concatenated to maintain consistent dimension throughout the model.

3.2.2 Entity-Hyperedge-Attribute Message Passing

We implement a bidirectional message passing mechanism with specialized attention mechanisms that are specifically designed to differentiate between entity-entity relationships and entity-attribute relationships captured by heterogeneous hyperedges. Our approach introduces two specialized types of attention mechanisms: (i) a self-attention mechanism that focuses specifically on relationships between different entity types, capturing their structural connections through heterogeneous hyperedges, and (ii) a cross-type attention mechanism that specifically assesses the relevance of attribute nodes to entities, enabling context-aware integration of attribute information. These attention mechanisms are implemented in a type-aware manner by processing each edge type $r \in \mathcal{R}$ separately, where each r corresponds to a specific type of heterogeneous hyperedge. For each edge type, the message passing process consists of two distinct steps: first from nodes to hyperedges, and then from hyperedges back to nodes. In each step, we use multi-head attention (Vaswani et al., 2017) with H_{node} and H_{edge} attention heads respectively, allowing the model to capture different aspects of the relationships.

1) **Node → Hyperedge Message Propagation**

In this step, we compute messages from nodes to their connected hyperedges. For each node v connected to hyperedge e with relation type r (representing a specific type of heterogeneous hyperedge), we incorporate edge type information into the node representation $\mathbf{H}_{v,r}^{(l)} = \mathbf{H}_v^{(l)} + \mathbf{e}_r$ and then compute attention-weighted messages:

$$\begin{aligned} \mathbf{Q}_1^h &= \mathbf{W}_{Q,1}^h \mathbf{H}_{v,r}^{(l)}, \\ \mathbf{K}_1^h &= \mathbf{W}_{K,1}^h \mathbf{H}_{v,r}^{(l)}, \\ \mathbf{V}_1^h &= \mathbf{W}_{V,1}^h \mathbf{H}_{v,r}^{(l)}, \end{aligned} \quad (2)$$

where $\mathbf{W}_{Q,1}^h, \mathbf{W}_{K,1}^h, \mathbf{W}_{V,1}^h \in \mathbb{R}^{d \times d}$ are learnable weight matrices for query, key, and value projections in the h-th attention head, respectively. The message from nodes to hyperedge e with relation type r is then computed as:

$$\mathbf{M}_{e,r}^{(l)} = \sum_{h=1}^{H_{edge}} \text{softmax}\left(\frac{\mathbf{Q}_1^h (\mathbf{K}_1^h)^\top}{\sqrt{d_k}}\right) \mathbf{V}_1^h, \quad (3)$$

where d_k is the dimension of the key vectors, and the attention weights determine how much each node's features contribute to the hyperedge representation.

2) **Hyperedge → Node Message Propagation**

After obtaining the hyperedge representations, we propagate information back to the nodes as:

$$\begin{aligned}\mathbf{Q}_{2,r}^h &= \mathbf{W}_{Q,2,r}^h \mathbf{H}_v^{(l)},\\ \mathbf{K}_{2,r}^h &= \mathbf{W}_{K,2,r}^h \mathbf{M}_{e,r}^{(l)},\\ \mathbf{V}_{2,r}^h &= \mathbf{W}_{V,2,r}^h \mathbf{M}_{e,r}^{(l)},\end{aligned} \tag{4}$$

where $\mathbf{W}_{Q,2,r}^h, \mathbf{W}_{K,2,r}^h, \mathbf{W}_{V,2,r}^h \in \mathbb{R}^{d\times d}$ are learnable weight matrices for query, key, and value projections specific to relation type r in the h-th attention head. The updated node representation for the next layer is then computed as:

$$\mathbf{H}^{(l+1)} = \mathbf{H}^{(l)} + \sum_{r\in\mathcal{R}} \sum_{h=1}^{H_{node}} \text{softmax}\left(\frac{\mathbf{Q}_{2,r}^h (\mathbf{K}_{2,r}^h)^\top}{\sqrt{d_k}}\right)\mathbf{V}_{2,r}^h, \tag{5}$$

where $\mathbf{Q}_{2,r}^h$, $\mathbf{K}_{2,r}^h$, and $\mathbf{V}_{2,r}^h$ are computed using transformation matrices that depend on the relation type r. This approach allows for the integration of type-specific information into the node representations.

For deeper networks, we stack multiple layers of this message passing mechanism, where each layer's output serves as the input to the next layer. The multi-layer architecture enables the model to capture increasingly complex patterns in the heterogeneous hypergraph. Our message passing approach specifically addresses the challenge of maintaining distinct representations for entity and attribute nodes throughout the information exchange process. These specialized attention mechanisms enable our model to effectively integrate both structural patterns and semantic attribute information while preserving the distinct functional characteristics of different node types.

3.2.3 Output Transformation

After L layers of message passing, we obtain the final hidden representations $\mathbf{H}^{(L)} \in \mathbb{R}^{n\times d_{hidden}}$ for all nodes, where n is the total number of nodes and d_{hidden} is the hidden dimension. To further enhance the type-specific characteristics, we apply different multi-layer perceptrons (MLPs) for each node type to obtain the final node representations:

$$\mathbf{Z}_v = \text{MLP}_{\tau(v)}(\mathbf{H}_v^{(L)}), \tag{6}$$

where v represents a node, $\tau(v) \in \mathcal{T}$ denotes its type, and $\text{MLP}_{\tau(v)}$ is the type-specific transformation network for that node type. This means that entities and different types of attributes each have their own dedicated transformation networks.

Each $\text{MLP}_{\tau(v)}$ consists of a two-layer neural network:

$$\begin{aligned}\text{MLP}_{\tau(v)}(\mathbf{h}) = \mathbf{W}_2^{\tau(v)}\text{Dropout}(\\ \text{ELU}(\text{LN}(\mathbf{W}_1^{\tau(v)}\mathbf{h} + \mathbf{b}_1^{\tau(v)}))) + \mathbf{b}_2^{\tau(v)}\end{aligned} \tag{7}$$

Here, $\mathbf{W}_1^{\tau(v)} \in \mathbb{R}^{d_{hidden}\times d_{hidden}}$, $\mathbf{W}_2^{\tau(v)} \in \mathbb{R}^{d_{out}\times d_{hidden}}$ are type-specific learnable weights, with corresponding biases $\mathbf{b}_1^{\tau(v)} \in \mathbb{R}^{d_{hidden}}$ and $\mathbf{b}_2^{\tau(v)} \in \mathbb{R}^{d_{out}}$. We maintain consistent d_{out} across all node types for downstream compatibility, applying Layer Normalization (Ba et al., 2016), ELU activation, and Dropout (Srivastava et al., 2014) to enhance model training. The type-specific MLPs are crucial for maintaining the distinct representational spaces of entity and attribute nodes, ensuring that our model preserves the fundamental functional differences between these node types even after multiple rounds of message passing.

Throughout our entire model architecture, including both the message passing encoder stack and these output transformations, we employ residual connections (He et al., 2016) to facilitate gradient flow and ensure stable training.

3.3 Learning Objectives

Our training approach employs a multi-objective optimization strategy that captures both structural patterns within the hypergraph and the relationships between entities and their attributes. The overall loss function combines two complementary components:

$$\mathcal{L} = \mathcal{L}_{\text{TriCL}} + \lambda\mathcal{L}_{\text{AACL}}, \tag{8}$$

where $\mathcal{L}_{\text{TriCL}}$ focuses on structural pattern learning, $\mathcal{L}_{\text{AACL}}$ targets entity-attribute relationships, and λ is a weighting hyperparameter that balances these objectives.

3.3.1 Structural Learning via TriCL

The TriCL framework (Lee and Shin, 2023) captures hypergraph structural patterns through symmetric contrastive learning between two augmented views of the same hypergraph. We adopt this effective framework as our base structural learning component, utilizing its two key augmentation strategies: (1) hypergraph connectivity masking (controlled by *drop-incidence-rate*, which removes node-hyperedge connections) and (2) node feature masking (controlled by *drop-feature-rate*, which

masks dimensions in feature vectors). These create diversified views while preserving the hypergraph's underlying structure. TriCL employs three complementary contrastive objectives:

$$\mathcal{L}_{\text{TriCL}} = \mathcal{L}_{\text{node}} + \mathcal{L}_{\text{edge}} + \mathcal{L}_{\text{member}}, \tag{9}$$

where $\mathcal{L}_{\text{node}}$ encourages similar representations for nodes appearing in similar hyperedges across views, $\mathcal{L}_{\text{edge}}$ promotes similar representations for hyperedges containing similar sets of nodes across views, and $\mathcal{L}_{\text{member}}$ preserves hyperedge membership information in the learned representations. By contrasting these representations between augmented views, TriCL effectively captures the structural topology of the hypergraph.

3.3.2 Attribute-Attentive Contrastive Learning Loss

While the TriCL component captures structural patterns within the hypergraph topology, existing hypergraph representation learning methods, including TriCL, often underutilize the rich attribute information associated with nodes. This limitation is particularly important in scientific literature networks where papers (entities) have various attributes (titles, abstracts, compound names, etc.) that provide crucial semantic context beyond structural connections. To address this limitation, we propose the Attribute-Attentive Contrastive Learning (AACL) loss that explicitly relates entity nodes to their attribute information. This loss function is designed to maximize the alignment between entity representations and their corresponding attribute-integrated representations, while maintaining clear distinctions between different entities.

Our AACL loss functions complementarily with the TriCL loss, which serves as the base component of our overall learning framework. While the TriCL loss focuses on capturing structural relationships within the hypergraph, our proposed AACL loss specifically targets the effective integration of attribute information into entity representations. A key feature is the dynamic learning of relative importance across different attribute types through an attention mechanism, enabling context-aware attribute integration.

Formally, the AACL loss is defined as a contrastive learning objective that maximizes similarity between entity representations and their attribute-integrated representations:

$$\mathcal{L}_{\text{AACL}} = -\sum_i \log \frac{\exp(\text{sim}(\hat{\mathbf{z}}_i, \hat{\mathbf{z}}_i^{\text{attr}})/\tau)}{\sum_j \exp(\text{sim}(\hat{\mathbf{z}}_i, \hat{\mathbf{z}}_j^{\text{attr}})/\tau)}, \tag{10}$$

where $\hat{\mathbf{z}}_i$ is the normalized representation vector of entity i, $\hat{\mathbf{z}}_i^{\text{attr}}$ is the normalized attribute-integrated representation of entity i, $\text{sim}(\cdot,\cdot)$ is the cosine similarity function, and τ is a temperature parameter.

The attribute-integrated representation is computed as:

$$\mathbf{z}_i^{\text{attr}} = \mathbf{Z}_i + \sum_{a \in \mathcal{A}(i)} \alpha_{i,a} \mathbf{Z}_a, \tag{11}$$

where $\mathcal{A}(i)$ is the set of attributes associated with entity i, $\alpha_{i,a}$ is the attention score of attribute a for entity i, and $\mathbf{Z}_a$ is the encoded representation of attribute a output by the heterogeneous hypergraph encoder.

The attention scores $\alpha_{i,a}$ are computed using an attention mechanism:

$$\alpha_{i,a} = \frac{\exp(f(\mathbf{Z}_i, \mathbf{Z}_a))}{\sum_{a' \in \mathcal{A}(i)} \exp(f(\mathbf{Z}_i, \mathbf{Z}_{a'}))}, \tag{12}$$

where f is implemented as a two-layer neural network:

$$f(\mathbf{Z}_i, \mathbf{Z}_a) = W_2 \text{ReLU}(W_1[\mathbf{Z}_i; \mathbf{Z}_a] + b_1) + b_2 \tag{13}$$

For an efficient implementation, we utilize sparse matrix operations. The specifics of this implementation and an analysis of its computational overhead are provided in Appendix E.

Implementation Details We integrate AACL into TriCL's dual-view framework (Lee and Shin, 2023), computing the loss for each augmented view and averaging across views (see Appendix D for complete implementation details).

4 Experiments

To evaluate our proposed method, we conducted experiments on multiple hypergraph datasets. Our evaluation consists of four main experiments: (1) entity-attribute integration evaluation using our proposed AACL loss to demonstrate the effectiveness of explicit entity-attribute modeling, (2) connectivity-based performance analysis to understand when attribute information provides the most benefit by analyzing performance across different hyperedge membership counts, (3) comparison with text-aware hypergraph methods to validate our approach against existing methods that incorporate textual information. Additionally, we provide comprehensive evaluation on standard benchmark datasets in Appendix I to demonstrate the effectiveness of our heterogeneous hypergraph encoder

Method	PubMed	Cora-C	Cora-A
HypeBoy	83.11 ± 0.6	81.63 ± 1.3	78.46 ± 1.0
TriCL	83.95 ± 0.4	81.34 ± 1.2	81.12 ± 0.9
Ours (Base)	84.86 ± 0.6	81.23 ± 1.0	81.12 ± 1.0
Ours (w/ Attrs.)	85.32 ± 0.6	82.00 ± 1.0	82.10 ± 0.8
Ours (Full)	**85.59 ± 0.6**	**82.33 ± 0.9**	**82.38 ± 0.8**
vs. TriCL	+1.64	+0.99	+1.26

Table 1: Node classification accuracy [%] comparison on datasets with attribute (Attr) information.

even without attribute modeling. To quantify the gains from structural learning itself, we also report results from a non-structural baseline using only pre-trained embeddings in Appendix J.

4.1 Experimental Setup

To evaluate our approach, we compare three model variants: **Ours (Base)**, our encoder on an entity-only hypergraph; **Ours (w/ Attributes)**, which adds attribute nodes; and **Ours (Full)**, which further incorporates our AACL loss. We conduct our primary evaluation on the PubMed, Cora-C, and Cora-A datasets, chosen for their rich textual attributes. Following the standard linear evaluation protocol (Veličković et al., 2019), we train the encoder unsupervisedly and then evaluate the frozen embeddings with a linear classifier. We compare our method against state-of-the-art supervised, self-supervised, and attribute-as-feature baselines. Comprehensive details on datasets (Appendix A), baselines (Appendix B), attribute encoders (Appendix F), and hyperparameters (Appendix C) are provided in the appendices.

4.2 Results and Analysis

4.2.1 Integration of Attribute Nodes with the AACL loss

Table 1 presents results from primary experiments on the PubMed, Cora-C, and Cora-A datasets, where we explicitly model entity-attribute relationships and incorporate our proposed AACL loss. It compares our three model variants against state-of-the-art baselines including TriCL and HypeBoy. The best performance is highlighted in **boldface** and the second-best is underlined.

The results demonstrate a clear progression in performance with each added component of our approach. Results show consistent improvements: +1.64% on PubMed, +0.99% on Cora-C, and +1.26% on Cora-A over TriCL, progressing from the base model to attribute modeling and finally to the AACL loss. These improvements are statistically significant, exceeding three standard deviations (3σ) from the baseline variance. Importantly, the "Ours (w/ Attrs.)" variant—which adds attribute nodes but uses only the TriCL structural loss without our AACL loss—already shows substantial gains (+0.46% on PubMed, +0.77% on Cora-C, +0.98% on Cora-A) over the base model. This demonstrates that the topological change itself is beneficial, independent of the semantic alignment provided by AACL. On PubMed, where domain-specific attributes are available, gains are most pronounced, highlighting heterogeneous attribute integration value.

The effectiveness of our AACL loss can be attributed to its ability to dynamically assess the importance of different attribute types through attention mechanisms. For instance, in scientific papers, abstracts often contain more comprehensive information than titles, while certain domain-specific attributes (like compound names in PubMed) may carry critical semantic information. Our loss function enables the model to appropriately weight these different attribute types based on their relevance to each entity, rather than treating all attributes equally.

Our ablation design deliberately pairs the attribute-augmented structure with the purely structural TriCL loss ("Ours (w/ Attrs.)") to isolate the topological contribution. Since TriCL provides no explicit semantic supervision for entity-attribute alignment, gains from this variant can be attributed solely to the structural change—demonstrating that defining "where" information comes from (via the heterogeneous encoder) provides benefits independent of "how" it is aligned (via AACL).

For a detailed analysis of the impact of different pre-trained attribute encoders on model performance, please refer to Appendix F. We also performed parameter sensitivity analysis in Appendix H.

4.2.2 Performance Analysis by Hyperedge Membership Count

To further understand when attribute modeling is most beneficial, we analyzed our model's performance across different node connectivity levels. Table 2 shows the performance evaluation results categorized by hyperedge membership count on PubMed. We classified nodes into five groups (1-2, 3-5, 6-10, 11-20, $\geq$21) and analyzed performance

Method	PubMed				
	1-2	3-5	6-10	11-20	≥21
# of nodes	264	750	969	815	274
Ours (Base)	80.21	83.09	83.52	87.34	92.22
Ours (w/ Attributes)	82.09	83.72	84.31	86.97	91.43
Ours (Full)	**82.31**	**83.84**	**84.68**	**87.39**	**91.72**
Improvement	+2.10	+0.75	+1.16	+0.05	-0.50

Table 2: Node classification accuracy [%] analysis based on hyperedge membership counts for PubMed dataset.

Method	Cora-C	Cora-A
Attribute-as-Feature		
HyperBERT	80.64 ± 0.8	80.23 ± 1.1
Ours (Base + Features)	81.45 ± 0.7	81.52 ± 0.9
Attribute-as-Structure		
Ours (Full)	$\mathbf{82.33 \pm 0.9}$	$\mathbf{82.38 \pm 0.8}$

Table 3: Node classification accuracy [%]: attribute-as-structure vs. attribute-as-feature approaches.

across variants.

Our results strongly support that attribute information is most beneficial for nodes with limited structural information. The largest improvement (+2.10%) occurs in the lowest connectivity group (1-2), where structural information is most limited. Notably, as connectivity increases, the benefit of attribute information generally diminishes, culminating in a slight performance decrease (-0.50%) for the highest connectivity group (≥21). This performance degradation for highly connected nodes suggests that attribute information may introduce noise when rich structural signals are already available, creating a form of feature redundancy or even feature conflict.

These results provide strong support for our central hypothesis: the attribute-as-structure approach is most effective when structural information is scarce. For nodes with 1-2 hyperedge connections, traditional aggregation-based methods suffer from insufficient neighborhood information, leading to under-informed representations. Our approach compensates by creating attribute-mediated aggregation pathways, effectively expanding the information neighborhood for sparse entities. Notably, even 1-to-1 attributes (e.g., titles, abstracts) contribute by enabling sparse entities to aggregate rich semantic information from their own attribute nodes, even without creating connections between different entities through shared attributes. Conversely, for highly connected nodes (≥21 connections), abundant structural signals may render additional attribute pathways redundant or even harmful, as evidenced by the slight performance decline.

We observed similar patterns in the Cora-C and Cora-A datasets, with the consistent trend that attribute information is most valuable for structurally sparse regions. For space constraints, we present the detailed results and analysis for these datasets in Appendix G. The consistent improvement for less-connected nodes across all three datasets provides strong support for one of our central claims: explicit modeling of entity-attribute relationships through our proposed method is particularly valuable when structural information alone is insufficient for accurate representation learning.

We also evaluated our base model on 10 benchmark datasets in the no attribute hypergraph setting (without attribute nodes) and found it achieves competitive or superior performance compared to existing methods (detailed results in Appendix I), demonstrating the effectiveness of our heterogeneous hypergraph encoder architecture even without explicit attribute modeling.

4.2.3 Comparison with Attribute-as-Feature Methods

To validate our attribute-as-structure approach, we compare against attribute-as-feature methods that treat textual content as node features rather than distinct structural components. We evaluate on Cora-C and Cora-A datasets using: (1) HyperBERT (Bazaga et al., 2024), a state-of-the-art method combining BERT with hypergraph message passing, and (2) our base model with concatenated title-abstract features (Ours (Base + Features)). Implementation details are provided in Appendix K.

Table 3 suggests the effectiveness of our attribute-as-structure approach. Our method consistently outperforms both attribute-as-feature baselines, with particularly notable improvements over HyperBERT (+1.69% on Cora-C, +2.15% on Cora-A). The moderate gains over our feature-concatenated baseline (+0.88% and +0.86%) confirm that structural integration of attributes provides additional benefits beyond simple feature enrichment, supporting our core hypothesis that creating new aggregation pathways addresses degree bias more effectively than feature augmentation alone.

Computational Efficiency. Our attribute-as-structure approach demonstrates a favorable performance-cost trade-off, achieving sub-linear

computational scaling despite the increase in node count. Specifically, despite a 3× increase in node count from adding attribute nodes, inference time increases by only ∼1.5× (e.g., 30ms to 48ms on PubMed), as shown in Table 6 in Appendix E. This sub-linear scaling is achieved through efficient sparse matrix operations and our specialized attention mechanisms.

5 Conclusion

In this paper, we addressed the fundamental problem of degree bias in hypergraph representation learning. We proposed a "attribute-as-structure" approach that creates new aggregation pathways, demonstrating that this topology-based solution effectively mitigates the structural root of degree bias. Our experiments show consistent improvements (up to +1.64% overall, +2.10% for sparse nodes), supporting our core hypothesis that information aggregation pathways are key to solving structurally-rooted problems in hypergraph representation learning.

Our work provides a principled approach for integrating auxiliary information into graph learning by transforming it into structural pathways, with broad applicability to knowledge graphs, recommendation systems, and information retrieval. Our dual attention mechanism also offers a template for preserving semantic distinctiveness in multi-type networks. Furthermore, our findings suggest that topology, rather than node features, is crucial for determining information flow, indicating that future research in this area should prioritize structural innovations. Future work could extend this approach to multi-modal data and explore adaptive structure construction. To ensure reproducibility, our code and experimental data will be made publicly available upon acceptance.

Limitations

The primary limitation of our work is the computational overhead introduced by adding attribute nodes, which may affect scalability for very large graphs. Our current approach focuses on textual attributes; extending the framework to other modalities, such as images or numerical data, would require further investigation and specialized encoders. Additionally, the optimal balance between structural and attribute information appears to be dataset-dependent, suggesting that finding this balance for new applications may require careful parameter tuning.

Ethical Considerations

In this work, we utilized AI-based tools to assist in code implementation and to improve the clarity and readability of the manuscript. The core research ideas, experimental design, and analysis of the results were conducted entirely by the author. The use of AI was limited to a supporting role, and the intellectual contributions of this paper are solely the author's own.

Acknowledgments

This research was supported in part by KIOXIA Corporation.

References

Alessia Antelmi, Gennaro Cordasco, Mirko Polato, Vittorio Scarano, Carmine Spagnuolo, and Dingqi Yang. 2024. A survey on hypergraph representation learning. *ACM Comput. Surv.*, 56(1):24:1–24:38.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

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*, pages 9181–9193, Miami, Florida, USA. Association for Computational Linguistics.

Ying Chen, Jinbo Bi, and James Z. Wang. 2003. A visual parts-based object recognition system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhixuan Chu, Yan Wang, Qing Cui, Longfei Li, Wenqing Chen, Zhan Qin, and Kui Ren. 2024. Llm-guided multi-view hypergraph learning for human-centric explainable recommendation. *Preprint*, arXiv:2401.08217.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Yihe Dong, Will Sawin, and Yoshua Bengio. 2020. Hnhn: Hypergraph networks with hyperedge neurons.

Dheeru Dua and Casey Graff. 2017. UCI machine learning repository.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. 2018. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 264–272.

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*, DL '98, page 89–98. Association for Computing Machinery.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Trans. Comput. Healthcare*, 3(1).

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

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 *The Twelfth International Conference on Learning Representations*.

Van Thuy Hoang, Hyeon-Ju Jeon, and O-Joun Lee. 2025. Mitigating degree bias in graph representation learning with learnable structural augmentation and structural self-attention. *IEEE Transactions on Network Science and Engineering*, 12(5):3656–3670.

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

Mingxuan Ju, Tong Zhao, Wenhao Yu, Neil Shah, and Yanfang Ye. 2023. Graphpatcher: Mitigating degree bias for graph neural networks via test-time augmentation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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

Ken Lang. 1995. Newsweeder: Learning to filter netnews. pages 331–339.

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.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*.

Zexi Liu, Bohan Tang, Ziyuan Ye, Xiaowen Dong, Siheng Chen, and Yanfeng Wang. 2024. Hypergraph transformer for semi-supervised classification.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Mag.*, 29(3):93–106.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. 2015. Multi-view Convolutional Neural Networks for 3D Shape Recognition . In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 945–953, Los Alamitos, CA, USA. IEEE Computer Society.

Arjun Subramonian, Jian Kang, and Yizhou Sun. 2024. Theoretical and empirical insights into the origins of degree bias in graph neural networks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Qiaoyu Tan, Xin Zhang, Xiao Huang, Hao Chen, Jundong Li, and Xia Hu. 2024. Collaborative graph neural networks for attributed network embedding.

Yanran Tang, Ruihong Qiu, Yilun Liu, Xue Li, and Zi Huang. 2024. Casegnn: Graph neural networks for legal case retrieval with text-attributed graphs.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International*

Conference on Neural Information Processing Systems, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. In *International Conference on Learning Representations*.

Jinlu Wang, Jipeng Guo, Yanfeng Sun, Junbin Gao, Shaofan Wang, Yachao Yang, and Baocai Yin. 2025. Dgnn: Decoupled graph neural networks with structural consistency between attribute and graph embedding representations. *IEEE Transactions on Big Data*, 11(4):1813–1827.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference.

Tianxin Wei, Yuning You, Tianlong Chen, Yang Shen, Jingrui He, and Zhangyang Wang. 2022. Augmentations in hypergraph contrastive learning: Fabricated and generative. In *Advances in Neural Information Processing Systems*.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920.

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*, KDD '22, page 2100–2109, New York, NY, USA. Association for Computing Machinery.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. Hypergcn: a new method of training graph convolutional networks on hypergraphs. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Jaewon Yang and Jure Leskovec. 2012. Community-affiliation graph model for overlapping network community detection. In *2012 IEEE 12th International Conference on Data Mining*, pages 1170–1175.

A Dataset Details

For a fair comparison, we use the same 10 benchmark datasets as used in TriCL (Lee and Shin, 2023) from the existing hypergraph neural networks literature; these datasets are categorized into: (1) co-citation datasets (Cora (McCallum et al., 2000), Citeseer (Giles et al., 1998), and PubMed (Sen et al., 2008)), (2) co-authorship datasets (Cora-A and DBLP (Yang and Leskovec, 2012)), (3) computer vision and graphics datasets (NTU2012 (Chen et al., 2003) and ModelNet40 (Wu et al., 2015)), and (4) datasets from the UCI Categorical Machine Learning Repository (Dua and Graff, 2017) (Zoo, 20Newsgroups (Lang, 1995), and Mushroom). Some basic statistics of the datasets are provided in Table 4.

Dataset	#Nodes	#Hyperedges	#Features	#Classes
Cora-C	1,434	1,579	1,433	7
Citeseer	1,458	1,079	3,703	6
PubMed	3,840	7,963	500	3
Cora-A	2,388	1,072	1,433	7
DBLP	41,302	22,363	1,425	6
NTU2012	2,012	2,012	100	67
ModelNet40	12,311	12,311	100	40
Zoo	101	43	16	7
20Newsgroups	16,242	100	100	4
Mushroom	8,124	298	22	2

Table 4: Dataset statistics.

The co-citation datasets are composed of a set of papers and their citation links. To represent a co-citation relationship as a hypergraph, papers become nodes and citation links become hyperedges. To be specific, the nodes $v_1, \ldots, v_k$ compose a hyperedge e when the papers corresponding to $v_1, \ldots, v_k$ are referred by the document e. The co-authorship datasets are composed of a set of papers with their authors. In hypergraphs that model the co-authorship datasets, nodes and hyperedges represent papers and authors, respectively. Precisely, the nodes $v_1, \ldots, v_k$ compose a hyperedge e when the papers corresponding to $v_1, \ldots, v_k$ are written by the author e. Features of each node are represented by bag-of-words features from its abstract. Nodes are labeled with their categories. The hypergraphs preprocessed from all the co-citation and co-authorship datasets are publicly available with the official implementation of HyperGCN (Yadati et al., 2019).

For visual datasets, the hypergraph construction follows the setting described in Feng et al. (Feng et al., 2019), and the node features are extracted by Group-View Convolutional Neural Network (GVCNN) (Feng et al., 2018) and Multi-View Convolutional Neural Network (MVCNN) (Su et al., 2015).

In the 20Newsgroups dataset, the TF-IDF representations of news messages are used as the node

Dataset	Title	Abstract	Other Attributes
PubMed	3,840	3,840	605 (compound names)
Cora-C	1,434	1,434	-
Cora-A	2,388	2,388	-

Table 5: Attribute information statistics for the main experimental datasets.

features. In the Mushroom dataset, the node features indicate categorical descriptions of 23 species of mushrooms. In the Zoo dataset, the node features are a mix of categorical and numerical measurements describing different animals.

For the main datasets used in our entity-attribute distinction experiments (PubMed, Cora-C, and Cora-A), we additionally collected detailed attribute information as shown in Table 5. For each dataset, we extracted title and abstract information from all papers. For PubMed, we also collected compound names mentioned in the papers, which serve as domain-specific attributes. These rich attribute information sources allow our model to leverage heterogeneous data types for improved entity representation learning.

We remove nodes that are not included in any hyperedge (i.e. isolated nodes) from the hypergraphs, because such nodes cause trivial structures in hypergraphs and their predictions would only depend on the features of that node. For all the datasets, we randomly select 10%, 10%, and 80% of nodes disjointly for the training, validation, and test sets, respectively.

B Baseline Methods

We compared our method with three categories of approaches: (1) supervised hypergraph learning methods that require labeled data, including MLP (a simple baseline), HGNN (Feng et al., 2019), HyperGCN (Yadati et al., 2019), UniGCN (Huang and Yang, 2021), and AllSet; (2) state-of-the-art self-supervised hypergraph contrastive learning methods that do not require labels, including HypeBoy (Kim et al., 2024) and TriCL (Lee and Shin, 2023); and (3) attribute-as-feature hypergraph methods that incorporate textual information HyperBERT (Bazaga et al., 2024), which combines pre-trained language models with hypergraph message passing.

C Hyperparameter Search Details

A key finding was that the optimal loss balancing coefficient λ was highly dataset-dependent, ranging from 0.5 for Cora-A to 1.8 for PubMed, highlighting the varying importance of attribute information.

C.1 Search Space

We conducted a comprehensive hyperparameter search for each model and dataset using Bayesian optimization with Weights & Biases (Wandb). This approach explored key architectural parameters (e.g., number of layers, hidden dimensions) and optimization parameters (e.g., learning rates, weight decay). Based on preliminary experiments, we used a fixed configuration for attention mechanisms with 8 attention heads for both node-to-hyperedge ($H_{edge} = 8$) and hyperedge-to-node ($H_{node} = 8$) message passing. The dimension of key vectors d_k was set to d_{hidden}/H_{edge} or d_{hidden}/H_{node} accordingly. Optimal configurations found through the search typically employed 1-3 layers with hidden dimensions between 256 and 512.

The explored search space for the main parameters was as follows:

- epochs: {400, 450, 500, ..., 1150, 1200}
- learning-rate: {5e-5, 1e-4, 5e-4, 1e-3, 5e-3}
- weight-decay: {1e-5, 1e-4, 1e-3}
- optimizer: {AdamW (Loshchilov and Hutter, 2019), RAdam (Liu et al., 2020)}
- L: {1, 2, 3}
- num-feature-layers: {1, 2, 3}
- d_{hidden} (hidden dimension): {128, 256, 512}
- d_{out} (output dimension): {128, 256, 512}
- dropout-rate: {0.1, 0.2, 0.3, 0.4, 0.5}
- drop-feature-rate: {0.1, 0.2, 0.3, 0.4, 0.5}
- drop-incidence-rate: {0.1, 0.2, 0.3, 0.4, 0.5}

For our attribute-aware models, we additionally tuned the loss balancing coefficient λ, which balances the structural TriCL loss and our proposed AACL loss. For attention mechanisms, we used a fixed configuration with 8 attention heads for both node-to-hyperedge ($H_{edge} = 8$) and hyperedge-to-node ($H_{node} = 8$) message passing processes

across all datasets and models, based on preliminary experiments. The dimension of key vectors d_k was set to d_{hidden}/H_{edge} for node-to-hyperedge attention and d_{hidden}/H_{node} for hyperedge-to-node attention, following standard practice in multi-head attention mechanisms.

D Implementation Details

Our approach builds upon the TriCL framework (Lee and Shin, 2023), which captures hypergraph structural patterns through symmetric contrastive learning between two augmented views. In TriCL, two distinct views of the input hypergraph are created by independently applying random connectivity masking with probability *drop-incidence-rate* and feature masking with probability *drop-feature-rate*. These augmentations create two different perspectives of the same underlying structure.

We extend this view-based contrastive learning approach to incorporate entity-attribute relationships. Since our model already computes embeddings across these two augmented views for the TriCL objectives, we efficiently leverage this same dual-view architecture to compute our AACL loss. Specifically, for each augmented view, we calculate the AACL as described in Section 3.3.2, and then average these losses across views. This integration allows us to simultaneously capture both structural patterns and entity-attribute relationships with minimal computational overhead compared to using TriCL alone, which comes only from the attention-based weighting and contrastive loss calculation for entity-attribute pairs.

E Detailed Computational Overhead Analysis

Efficient AACL Implementation Details. To efficiently compute the attribute-integrated representations required for the AACL loss, we utilize sparse matrix operations. The computation is formally expressed as:

$$\mathbf{z}_i^{\text{attr}} = \mathbf{Z}_i + \sum_{t \in \text{AttrTypes}} \alpha_{i,t}(S_t \mathbf{Z}_t), \qquad (14)$$

where S_t is a sparse matrix representing entity-attribute relationships for attribute type t, and $\mathbf{Z}_t$ represents the encoded representations of all attributes of type t output by our heterogeneous hypergraph encoder. This sparse implementation is crucial for achieving the sub-linear scaling discussed in this section.

Our "attribute-as-structure" approach introduces computational overhead primarily in the message passing and attention computation phases. The computational increase remains linear in the number of added attribute nodes and remains tractable due to sparse hyperedge connectivity. In our experiments, where we add two attribute nodes per entity (title and abstract), this translates to processing exactly three times more nodes during the encoding phase compared to entity-only methods, as empirically demonstrated in the following analysis.

When adding A attribute nodes per entity to N entity nodes, the main computational increases occur in: (1) node embedding computation, which processes $N + A \times N$ total nodes instead of N nodes, and (2) attention weight calculation within hyperedges that now include both entity and attribute nodes.

Since our method aggregates information through the existing hypergraph structure without introducing attribute-to-attribute interactions, the computational increase remains linear in the number of added attribute nodes. The attention mechanism complexity grows proportionally to the expanded node set within each hyperedge, but remains tractable due to sparse hyperedge connectivity.

To evaluate the practical viability of our attribute-as-structure approach, we conducted comprehensive computational efficiency experiments measuring both training and inference times across our three model variants. All efficiency experiments used identical hyperparameters with 50 training epochs. For fair comparison, we used only title and abstract attributes (excluding compound names in PubMed) to ensure consistent 3x node scaling across datasets.

Table 6 presents both the structural changes and corresponding computational overhead when incorporating attribute nodes. Training times are measured over 50 epochs with identical hyperparameters across all model variants. Across all datasets, we observe a consistent 3x increase in node count due to the addition of title and abstract attribute nodes, while hyperedge count increases by an average of 91% due to entity-attribute connections. Despite this notable structural expansion, our approach demonstrates sub-linear scaling in computational cost.

Our results validate the theoretical analysis: de-

Dataset	Hypergraph Size				Training (sec)			Inference (ms)	
	Base		w/ Attrs		Base	w/Attrs	Full	Base	w/Attrs
	Nodes	Edges	Nodes	Edges					
Cora-C	1,434	1,579	4,302	3,013	12	17	21	23.85	33.39
Cora-A	2,388	1,072	7,164	3,460	15	30	33	24.98	39.82
PubMed	3,840	7,963	11,520	11,803	46	91	96	30.06	47.80

Table 6: Computational efficiency comparison: hypergraph size and runtime analysis.

spite 3x node growth, training time increases only 80-101% and inference time 53%, demonstrating sub-linear scaling through efficient sparse hypergraph operations and specialized attention mechanisms. The additional cost of our AACL loss is minimal, adding only 13% training overhead (Full vs w/Attrs) with no inference penalty, supporting our design choice of leveraging the existing dual-view architecture from TriCL. Considering our average accuracy improvement of +1.64% with approximately 2x training time increase, our approach demonstrates a favorable performance-cost trade-off. Larger datasets (PubMed) show higher absolute overhead but maintain consistent scaling patterns, suggesting scalability to applications.

F Attribute Encoder Comparison

To extract meaningful features from textual attribute information, we employed several pre-trained language models as attribute node encoders, using them in a feature extraction mode with weights frozen (i.e., not fine-tuned). Specifically, we experimented with domain-specific encoders such as BiomedBERT (Gu et al., 2021) (optimized for biomedical text) for the PubMed dataset, as well as general-purpose encoders including ModernBERT (Warner et al., 2025) and All-MPNet-Base-v2 (Reimers and Gurevych, 2019) for all datasets. These pre-trained encoders allowed us to capture semantic relationships in the textual attributes before integrating them into our heterogeneous hypergraph model.

In this appendix, we provide a detailed analysis of the impact of different pre-trained encoders on the quality of attribute node representations. Table 7 shows the performance comparison on the validation set when using different encoders for attribute feature extraction. This validation performance guided our selection of the most appropriate encoder for each dataset in our final model.

For the PubMed dataset, the domain-specific BiomedBERT encoder yielded the best results with 85.59% accuracy, showing a 0.21% improvement over the general-purpose All-MPNet-Base-v2 encoder (85.38%). This highlights the importance of domain-specific knowledge in capturing accurate semantic relationships for specialized domains. The BiomedBERT model, being pre-trained specifically on biomedical literature, can better understand domain-specific terminology and contextual relationships between medical terms and compound names.

Attribute Encoder	PubMed	Cora-C	Cora-A
ModernBERT	85.06	81.50	81.47
All-MPNet-Base-v2	85.30	**82.48**	**82.21**
BiomedBERT	**85.42**	–	–

Table 7: Node classification accuracy [%] comparison of different attribute encoders across validation sets.

For more general datasets like Cora-C and Cora-A, the All-MPNet-Base-v2 encoder performed significantly better than ModernBERT, with nearly 1% improvement in both datasets (82.50% vs. 81.56% for Cora-C and 82.38% vs. 81.44% for Cora-A). This suggests that sentence-level encodings provided by All-MPNet-Base-v2 are particularly beneficial for capturing the semantics of academic paper titles and abstracts in these datasets. The model is trained to produce sentence embeddings that capture the overall meaning of academic text more effectively (Reimers and Gurevych, 2019).

Based on these results, we selected BiomedBERT as our attribute encoder for the PubMed dataset and All-MPNet-Base-v2 for the Cora-C and Cora-A datasets in our final model. These results demonstrate that the choice of attribute encoder significantly impacts the overall performance, and that domain-specific encoders can provide substantial benefits when available. Our experiments suggest that selecting the appropriate encoder based on the domain characteristics is an important consideration for optimizing model performance.

G Detailed Performance Analysis by Hyperedge Membership Count

Table 8 presents the detailed results of our performance analysis based on hyperedge membership counts for Cora-C and Cora-A datasets, complementing the PubMed results presented in the main text.

G.1 Analysis of Cora-C Results

The results for Cora-C demonstrate a similar pattern to what we observed in PubMed. Substantial improvements are observed for both low connectivity nodes (+1.2% for the 1-2 group) and moderately connected nodes (+2.6% for the 6-10 group), but we again see a performance decline (-1.3%) in the highest connectivity group. This reinforces our observation that there exists an optimal balance between structural and attribute information, with excessive information potentially confusing the model.

G.2 Analysis of Cora-A Results

In Cora-A, we observe a dramatic improvement (+24.2%) in the high connectivity group (11-20); however, as this group contains only 7 nodes, this large improvement should be interpreted with caution. The more reliable improvement in Cora-A is seen in the low connectivity groups (+1.8% for the 1-2 group) and moderately connected nodes (+2.2% for the 6-10 group), consistent with the patterns observed in PubMed.

G.3 Cross-Dataset Comparison

The distribution of nodes across connectivity groups varies considerably between datasets. While PubMed has a relatively balanced distribution (as shown in the main text), Cora-C and especially Cora-A are heavily skewed toward lower connectivity groups. Cora-A has over 97% of its nodes in the 1-2 and 3-5 groups, while Cora-C has approximately 80% of its nodes in these lower connectivity groups. This variation in structural characteristics makes the consistent pattern of improvement for less-connected nodes particularly notable, demonstrating the robust effectiveness of our approach across diverse network topologies.

H Parameter Sensitivity Analysis

To achieve optimal performance with our Attribute-Attentive Contrastive Learning (AACL) loss, we conducted comprehensive parameter sensitivity analysis for the regularization parameter λ that balances structural learning (TriCL) and entity-attribute relationship learning (AACL).

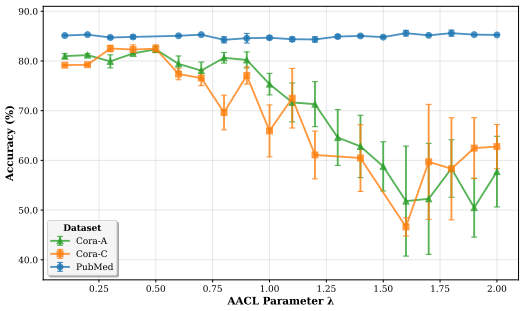

Figure 2: Effect of AACL parameter λ on node classification accuracy across datasets. Optimal values demonstrate dataset-dependent patterns, with citation networks requiring moderate regularization while biomedical literature benefits from stronger regularization.

Figure 2 reveals distinct dataset-dependent optimization patterns across our three primary experimental datasets. Citation networks (Cora-C and Cora-A) require moderate regularization with optimal values of $\lambda = 0.3$ and $\lambda = 0.5$ respectively, while the biomedical literature network (PubMed) benefits from stronger regularization with $\lambda = 1.8$.

Dataset-Specific Optimization Patterns. This pattern reflects the underlying structural differences between dataset types. PubMed's richer semantic attributes, including domain-specific compound names alongside titles and abstracts, benefit from stronger attribute-based regularization to fully leverage the diverse attribute information. In contrast, citation networks require more balanced structural-semantic integration, as excessive attribute emphasis may overshadow important structural patterns in these more structurally-driven networks.

Statistical Significance. The statistical significance of these patterns provides strong evidence for the importance of dataset-specific parameter tuning. For Cora-A, we observed a strong negative correlation between λ values and performance deviation from optimal ($r = -0.923$, $p < 0.001$), while Cora-C showed similar patterns ($r = -0.857$, $p < 0.001$). These correlations confirm that the optimal balance between structural and attribute information is highly dataset-dependent and requires careful tuning for optimal performance.

Practical Implications. These findings suggest that practitioners should conduct parameter sensitivity analysis when applying our method to

Method	Cora-C					Cora-A				
	1-2	3-5	6-10	11-20	≥21	1-2	3-5	6-10	11-20	≥21
# nodes	459	475	160	39	15	802	1050	52	7	1
Base	80.7	82.6	77.9	85.1	77.3	79.4	82.9	81.1	22.1	-
w/ Attr	81.7	83.2	78.9	84.6	**77.4**	80.7	83.3	**84.3**	**51.0**	-
Full	**81.9**	**83.6**	**80.5**	**85.7**	76.0	**81.2**	**83.5**	83.3	46.3	-
Improvement	+1.2	+1.0	+2.6	+0.6	-1.3	+1.8	+0.6	+2.2	+24.2	-

Table 8: Node classification accuracy [%] analysis based on hyperedge membership counts for Cora-C and Cora-A datasets.

new datasets, particularly focusing on the trade-off between structural pattern preservation and attribute information integration. Datasets with richer, more diverse attribute information may benefit from higher λ values, while structurally-rich datasets may require more conservative attribute integration.

I Node Classification in No Attribute Setting

To establish a comprehensive performance baseline and validate the effectiveness of our heterogeneous hypergraph encoder architecture, we evaluated our base model in the no attribute hypergraph setting without attribute nodes. In this setting, similar to most existing hypergraph learning methods, the hypergraph consists only of entity nodes.

Table 9 shows the performance comparison between our base model (Ours (Base)) and existing methods across 10 benchmark datasets in this no attribute setting.

Our base model, which incorporates our heterogeneous hypergraph encoder with TriCL as the learning objective, achieves performance equal to or better than existing contrastive learning approaches (HypeBoy, TriCL) across most datasets. Specifically, on the PubMed dataset, our model shows a 0.9% improvement compared to TriCL (84.86% vs. 83.95%), and on Zoo, a 1.7% improvement (81.85% vs. 80.12%).

Notably, when compared with supervised hypergraph learning methods (e.g., HGNN, HyperGCN), our model maintains competitive performance. While supervised methods like UniGCN show strong performance on specific datasets such as ModelNet40 (97.91%) and 20News (80.37%), the improvements our heterogeneous hypergraph encoder brings to the self-supervised learning approach help close the gap with these supervised methods.

It is worth noting that this is the same base model (Ours (Base)) that we used as the foundation in our primary experiments with attribute nodes, as shown in Table 1 in the main paper. These results demonstrate that even without explicitly modeling attribute nodes, our heterogeneous hypergraph encoder architecture provides more effective representation learning than existing methods with similar learning objectives.

We attribute this performance gain to our encoder's ability to better capture the structural patterns within hypergraphs through its specialized attention mechanisms. The consistent improvement across diverse datasets indicates the robust generalization capability of our approach in the no attribute hypergraph setting, which is further enhanced when attribute information is explicitly incorporated as demonstrated in our primary experiments.

This baseline evaluation serves two important purposes: (1) it validates that our architectural improvements are beneficial even in traditional settings, and (2) it provides a solid foundation upon which our attribute-as-structure innovations can build.

J Pre-trained Embedding Baseline Results

To evaluate the effectiveness of hypergraph representation learning, we conducted baseline experiments using pre-trained embeddings directly for node classification without any graph structure learning. This provides a lower bound for comparison with our proposed hypergraph-based methods.

J.1 Experimental Setup

We evaluated three feature combinations using pre-trained embeddings:

- **Title-only**: Title embeddings only (768-dim)

Method	Cora-C	Citeseer	Pubmed	Cora-A	DBLP	Zoo	20News	Mushroom	NTU2012	ModelNet40
Supervised Methods										
MLP	51.92 ± 1.6	55.14 ± 2.8	74.20 ± 0.8	55.43 ± 1.5	79.91 ± 0.3	75.43 ± 10.7	79.42 ± 0.4	95.76 ± 0.6	73.77 ± 2.4	95.61 ± 0.2
HGNN	79.79 ± 1.5	68.53 ± 1.1	83.43 ± 0.4	77.77 ± 0.8	89.47 ± 0.2	73.09 ± 8.9	80.32 ± 0.1	99.41 ± 0.4	73.22 ± 2.1	92.27 ± 0.2
HyperGCN	74.97 ± 2.3	67.35 ± 1.5	78.59 ± 0.4	71.50 ± 1.9	88.55 ± 0.6	70.62 ± 7.5	79.97 ± 0.3	99.36 ± 0.4	53.47 ± 0.3	79.46 ± 3.4
UniGCN	77.26 ± 1.7	67.88 ± 1.1	83.73 ± 0.5	78.41 ± 0.6	90.30 ± 0.1	78.52 ± 6.4	**80.37 ± 0.3**	99.83 ± 0.1	73.63 ± 1.6	97.91 ± 0.2
AllSet	75.94 ± 0.7	67.05 ± 1.6	83.72 ± 0.5	75.79 ± 1.7	89.32 ± 0.1	73.58 ± 7.7	80.01 ± 0.3	**99.84 ± 0.1**	71.66 ± 1.3	96.95 ± 0.4
Self-supervised Methods										
HypeBoy	**81.63 ± 1.3**	71.74 ± 1.2	83.11 ± 0.6	78.46 ± 1.0	90.58 ± 0.1	77.84 ± 10.7	77.28 ± 0.3	99.07 ± 0.3	73.77 ± 2.4	94.10 ± 0.2
TriCL	81.34 ± 1.2	72.23 ± 1.0	83.95 ± 0.4	**81.12 ± 0.9**	91.07 ± 0.1	80.12 ± 11.1	80.13 ± 0.2	99.83 ± 0.1	**75.24 ± 2.6**	97.06 ± 0.1
Ours (Base)	81.23 ± 1.0	**72.36 ± 1.3**	**84.86 ± 0.6**	**81.12 ± 0.9**	**91.18 ± 0.1**	**81.85 ± 6.2**	80.00 ± 0.2	99.59 ± 0.2	74.25 ± 2.5	**98.19 ± 0.1**

Table 9: Node classification accuracy [%] comparison in the no attribute hypergraph setting without attribute information. The best and second-best performances are highlighted in **boldface** and underlined, respectively.

Method	PubMed	Cora-C	Cora-A
Title-only	78.57 ± 0.8	67.59 ± 1.5	67.00 ± 1.0
Abstract-only	78.68 ± 1.1	72.06 ± 1.6	74.66 ± 0.9
Title+Abstract	**80.55 ± 0.8**	**74.56 ± 1.3**	**76.69 ± 1.1**
Comparison with Our Full Method			
Ours (Full)	85.59 ± 0.6	82.33 ± 0.9	82.38 ± 0.8
Improvement	+5.04	+7.77	+5.69

Table 10: Pre-trained embedding baseline results: Node classification accuracy [%] without hypergraph structure learning.

- **Abstract-only**: Abstract embeddings only (768-dim)
- **Title+Abstract**: Concatenated title and abstract embeddings (1536-dim)

For each dataset, we used the same pre-trained encoders as our main experiments: Biomed-BERT for PubMed and All-MPNet-Base-v2 for Cora datasets. Classification was performed using a simple linear classifier with the same train/validation/test splits (10%/10%/80%) and 5-fold evaluation protocol.

J.2 Results and Analysis

Table 10 presents the baseline results using pre-trained embeddings directly for node classification. Across all datasets, Title+Abstract concatenation achieves the highest performance, with Abstract-only generally outperforming Title-only, indicating that abstracts contain richer semantic information for classification tasks.

Consistent Feature Combination Effects: Across all datasets, Title+Abstract concatenation achieves the highest performance, with Abstract-only generally outperforming Title-only. This indicates that abstracts contain richer semantic information for classification tasks.

Dataset-Specific Performance: PubMed achieves the highest baseline performance (80.55%), followed by Cora-A (76.69%) and Cora-C (74.56%). This correlates with dataset size (PubMed: 3,840 papers, Cora-A: 2,388 papers, Cora-C: 1,434 papers) and suggests that larger datasets provide better generalization for text-based classification.

Overfitting Issues: All baseline experiments showed significant overfitting, with training accuracies reaching 90-100% while test performance remained 15-25% lower. This is particularly pronounced in smaller datasets (Cora-C, Cora-A) and highlights the limitation of purely text-based approaches without structural regularization.

Hypergraph Learning Benefits: Our full method achieves substantial improvements over the best pre-trained baselines: +5.04% on PubMed, +7.77% on Cora-C, and +5.69% on Cora-A. These improvements demonstrate the significant value of explicitly modeling hypergraph structures and entity-attribute relationships, particularly for smaller datasets where the relative improvements are larger.

Implications for Structural Learning: The consistent and substantial improvements across all datasets provide strong evidence that hypergraph representation learning offers significant advantages over direct text embedding approaches, even when using the same pre-trained encoders for textual content.

K HyperBERT Comparison Implementation Details

For HyperBERT experiments, we concatenated title and abstract information in the format "Title: [title], Abstract: [abstract]" as input text for each paper node. We focus on Cora-C and Cora-A as these provide standardized textual information suitable for HyperBERT's processing approach, while

PubMed's domain-specific compound names require specialized handling beyond HyperBERT's current capabilities.

For our baseline comparison (Ours (Base + Features)), we concatenated pre-encoded title and abstract embeddings (using All-MPNet-Base-v2) as 1,536-dimensional node features, representing a strong attribute-as-feature baseline using the same textual content as our attribute-as-structure approach.