HyperBERT: Mixing Hypergraph-Aware Layers with Language Models for Node Classification on Text-Attributed Hypergraphs

Adrián Bazaga, Pietro Liò, Gos Micklem

University of Cambridge, Cambridge, United Kingdom

$\{ar989, pl219, gm263\}$@cam.ac.uk

Abstract

Hypergraphs are characterized by complex topological structure, representing higher-order interactions among multiple entities through hyperedges. Lately, hypergraph-based deep learning methods to learn informative data representations for the problem of node classification on text-attributed hypergraphs have garnered increasing research attention. However, existing methods struggle to simultaneously capture the full extent of hypergraph structural information and the rich linguistic attributes inherent in the nodes attributes, which largely hampers their effectiveness and generalizability. To overcome these challenges, we explore ways to further augment a pretrained BERT model with specialized hypergraph-aware layers for the task of node classification. Such layers introduce higher-order structural inductive bias into the language model, thus improving the model's capacity to harness both higher-order context information from the hypergraph structure and semantic information present in text. In this paper, we propose a new architecture, HyperBERT, a mixed text-hypergraph model which simultaneously models hypergraph relational structure while maintaining the high-quality text encoding capabilities of a pre-trained BERT. Notably, HyperBERT presents results that achieve a new state-of-the-art on five challenging text-attributed hypergraph node classification benchmarks.

1 Introduction

In recent years, the study of hypergraphs has gained significant attention in the field of network analysis (Bai et al., 2021; Gao et al., 2022; Hou et al., 2022). Unlike traditional graphs, where edges represent pairwise relationships, hypergraphs are a generalization of graphs where an edge can join any number of nodes, allowing for more complex interactions by connecting multiple nodes through hyperedges. This characteristic makes hypergraphs an ideal framework for representing complex systems such as social networks, biological networks, and collaboration networks, where relationships often involve sets of entities.

Machine learning on hypergraphs (Wu et al., 2023b; Bai et al., 2021), has been shown to be an effective tool for studying complex graph-based data structures. In particular, when compared with a standard graph, a hypergraph is more capable and flexible in modeling high-order relations since one hyperedge in a hypergraph can connect more than two nodes. As a result, hypergraph learning has recently gained increasing attention and has been applied to many research fields, such as computer vision (Feng et al., 2019; Hou et al., 2023) and recommendation systems (Wu et al., 2022, 2023a).

In many real-world complex systems, hypergraph data is accompanied by node attribute information (Nakajima and Uno, 2024; Failla et al., 2023; Smaniotto and Pelillo, 2021; Yang et al., 2021). Some examples of node attributes include the age, ethnicity, and gender of individuals in social networks (Contisciani et al., 2020), affiliations of authors in collaboration networks (Pan et al., 2012), and article abstract texts in co-citation graphs (Hu et al., 2021). Previous studies have demonstrated that node attribute data potentially enhances the learning of community structure in networks (Chunaev, 2020; Bassolas et al., 2022).

A particular instance of attributed hypergraphs are Text-Attributed Hypergraphs (TAHGs) (Nakajima and Uno, 2024; Failla et al., 2023; Smaniotto and Pelillo, 2021; Yang et al., 2021), which have been widely used for modeling a variety of real-world applications, such as co-authoring in collaboration networks (Newman, 2001; Patania et al., 2017), information retrieval (Cohan et al., 2020; Yang et al., 2021), product recommendation (Zhu et al., 2021), and many others.

Given the prevalence of TAHGs (Nakajima and Uno, 2024; Failla et al., 2023; Smaniotto and Pelillo, 2021; Yang et al., 2021), we have explored how to effectively handle these graphs, with a fo-

9181

cus on node classification tasks. Such tasks aims at predicting the category or label of a node based on its textual attributes and the structure of the hypergraph. This problem is pivotal in various applications such as predicting protein functions in biological networks (Lugo-Martinez et al., 2020), identifying roles in social networks (Newman, 2001; Patania et al., 2017), and classifying documents in citation networks (Hu et al., 2021). Inherently, TAHGs provide both node attribute and graph structural information. Thus, it is critical to effectively capture both while modeling their interrelated correlation for effective representation learning.

Node classification in text-attributed hypergraphs presents unique challenges that are not encountered in traditional graph-based approaches (Li et al., 2023; Wu et al., 2021; Failla et al., 2023). In this context, there are two key hurdles. First, since hyperedges can connect any set of nodes, capturing such complex interaction patterns becomes increasingly difficult using conventional graph-based methods. Second, effectively exploiting the textual information to learn informative node representations is complex. Recent advances in the field of NLP have led to the emergence of language models (LMs) with strong capabilities for semantic understanding of text (Devlin et al., 2019; Raffel et al., 2020; Brown et al., 2020; Radford et al., 2019, 2018). However, the integration of hypergraph structural information directly into language models is still an open and under-researched topic.

To address the above limitations, we propose a novel architecture, HyperBERT, which mixes hypergraph-aware layers into language models to simultaneously exploit hypergraph topology and textual representations for the node classification task on text-attributed hypergraphs. In particular, HyperBERT extends the original BERT architecture with hypergraph-specific inductive biases. To do so, we design specially-crafted hypergraph-aware layers into the BERT encoder, effectively mixing hypergraph structure with text semantic embeddings for computing node representations. To accomplish the text-hypergraph alignment in the feature space, we propose a novel self-supervised loss for pretraining HyperBERT and effectively aligning semantic and hypergraph feature spaces. HyperBERT achieves state-of-the-art performance across five widely used hypergraph benchmarks. Our code is available at https://github.com/AdrianBZG/HyperBERT.

2 Related Work

Our work is related to two lines of research: text-attributed graph learning-based methods and language model-based approaches on text-attributed graphs (Zhao et al., 2023; Yang et al., 2021; Kuang et al., 2023). In this section, we review briefly the relevant literature for such categories.

Given the recent success of graph representation learning, numerous research works have been proposed for various tasks including node classification (Bhagat et al., 2011) or link prediction (Zhang and Chen, 2018). Graph Neural Networks (GNNs) are recognized as the de facto standard technique for modeling graph data. These methods (e.g. GCN; Kipf and Welling (2017), GAT; Veličković et al. (2018), GraphSAGE; Hamilton et al. (2017) or GIN; Xu et al. (2019)) learn effective mechanisms such that information between the nodes is aggregated for expressive graph representations. In the case of TAGs, a cascade architecture (Hamilton et al., 2017) is typically adopted. In such setting node features are encoded independently using text modeling techniques (e.g. Bag-of-Words (Zhang et al., 2010), skip-gram (Mikolov et al., 2013), or more recently language models such as BERT (Devlin et al., 2019)), and subsequently aggregated via the message passing mechanism to produce the final node representations. However, standard GNN-based approaches are only applicable to general graphs with binary relations.

To alleviate this issue recent research has focused on developing effective methodologies for attributed hypergraph learning in order to capture n-ary or high-order interactions between nodes (Wu and Ng, 2022; Yang et al., 2019; Wu et al., 2022; Wang et al., 2020). For instance, Zhang et al. (2018) proposes an inductive multi-hypergraph learning algorithm, which learns an optimal hypergraph embedding with good performance on the task of multi-view 3D object classification. More recently, several hypergraph-based methods have been proposed for the task of node classification (Yadati et al., 2019; Bai et al., 2021). In particular, (Feng et al., 2019) propose the Hypergraph Neural Network (HGNN) architecture, which encodes high-order node correlations using a hypergraph structure and generalizes the convolution operations to the hypergraph learning process based on hypergraph Laplacian and truncated Chebyshev polynomials. Bai et al. (2021) propose to supplement the family of graph neural networks with two end-to-

end trainable operations, i.e. hypergraph convolution and hypergraph attention (HCHA). Yadati et al. (2019) propose HyperGCN, model to enhance the hypergraph Laplacian with additional weighted pair-wise edges. Kim et al. (2024) propose HypeBoy, a hypergraph self-supervised learning method for node classification. However, HypeBoy primarily focuses on hypergraphs without specifically addressing the complexities introduced by nodes with textual attributes.

Given the success of LMs in solving semantic representation learning tasks, recent works (Zhang et al., 2021b) have proposed to utilise LMs for TAGs, as they can transform the nodes' textual attributes into semantically rich node embeddings. However, node embeddings provided by LMs often only contain information from the textual attributes, overlooking the topological structure present in the TAGs. To overcome this limitation, multiple works integrate LM-based embeddings with graph learning methods (Jin et al., 2023a,b), with the goal of generating node embeddings that are tailored for specific TAG tasks.

For instance, Yang et al. (2021) propose Graphormer, which fuses LM-based text encoding and graph aggregation into an iterative workflow. GIANT (Chien et al., 2021) introduces a novel neighborhood prediction task to fine-tune a XR-Transformer (Zhang et al., 2021a), and feeds node embeddings as obtained by the LM into GNNs for downstream tasks. TAPE (He et al., 2023) presents a method that uses LMs to generate prediction text and explanation text, which serve as augmented text data to subsequently feed into GNNs. LM-GNN (Ioannidis et al., 2022) introduces graph-aware pre-training to adapt the LM on the given graph before fine-tuning the entire model, demonstrating significant performance results. SimTeG (Duan et al., 2023) finds that first training the LMs on the downstream task and then freezing the LMs and training the GNNs can result in improved performance. K-BERT (Liu et al., 2020) leverages a knowledge layer to inject relevant triplets from a knowledge graph into the input sentence and transform it into a knowledge-rich sentence tree. KG-BERT (Yao et al., 2019) treats triples in knowledge graphs as textual sequences. They propose to model entity and relation descriptions of a triple as input, and compute the scoring function of the triple with the KG-BERT language model. Despite these advances, previous works do not take into account the problem where both text attributes and

higher-order interactions exist between the entities, as is the case with text-attributed hypergraph node classification.

3 Preliminaries

In this paper, we focus on learning representations for nodes in Text-Attributed Hypergraphs (TAHG), where we take node classification as the downstream task. Therefore, before describing the details of our proposed method, we start with presenting a few basic concepts, including the formal definition of TAHGs and how language models can be used for node classification in TAHGs. Then, we introduce the formulation of the problem our method is tackling.
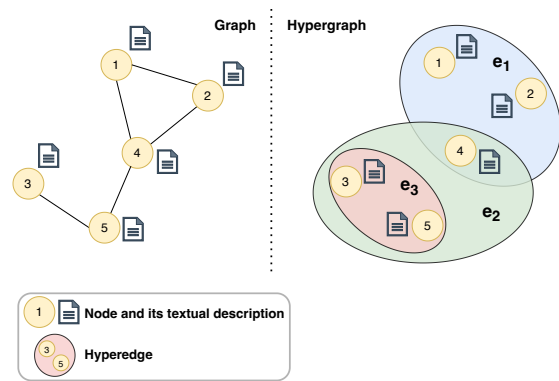

Figure 1: An illustration of a standard text-attributed graph (left) where nodes are connected between each other using binary relations, and a text-attributed hypergraph (right) where nodes are related with high-order connections (hyperedges). In both cases, each node is attributed with a textual description, such as paper abstract in the case for co-citation hypergraphs.

3.1 Text-Attributed Hypergraphs (TAHG)

Formally, a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{S}^V)$ is defined by a node set $\mathcal{V}$ and a hyperedge set $\mathcal{E}$. Each hyperedge $e_j \in \mathcal{E}$ is a non-empty set of nodes. Each node $v_i \in \mathcal{V}$ is associated with a sequential text feature, $s_i \in \mathcal{S}^V$, and the corresponding label y_i. Each label has a real label text c from the set of all label texts C (e.g. 'Artificial Intelligence' or 'Supervised Learning' for the case of paper categories in a co-citation hypergraph). For the sake of clarity, Figure 1 illustrates the difference between standard text-attributed graphs and text-attributed hypergraphs.

3.2 Pretrained Language Models (PLM) for Node Classification

In the context of TAGs, LMs can be employed to encode the text attributes associated with each node and learn a representation that captures the semantic meaning of the text. Let $s_i \in \mathcal{S}^V$ denote the text attributes of node i, and LM be a pre-trained network, such as BERT (Devlin et al., 2019). Then, the text attributes of node i can be encoded by applying BERT to s_i. The output of the LM is denoted as $\mathbf{z_i} \in \mathbb{R}^d$, where d is the dimensionality of the output feature vector. Then, to perform node classification, the output is typically fed into a fully connected layer with a softmax function for class prediction. The goal is to learn a function that maps the encoded text attributes to the corresponding node labels.

3.3 Problem Formulation

Given a text-attributed hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{S}^V)$, with each node associated with a textual description $\mathcal{S} = \{s_1, ..., s_{|\mathcal{S}|}\}$, where $|\mathcal{S}|$ is the number of tokens in the textual description $\mathcal{S}$, and a partial set of node labels $C \subset \mathcal{V}$, the goal is to predict the labels of the remaining nodes $P = \mathcal{V} \setminus C$.

4 Methodology

In this section we present our proposed model, HyperBERT, as illustrated in Figure 2. First, we begin with the details of the HyperBERT layer design. Then, we detail our proposed pre-training methodology, followed by details of the different components of our knowledge alignment loss function. Finally, we describe how fine-tuning is performed for a downstream node classification task.

4.1 HyperBERT Layer

The HyperBERT layer explicitly mixes semantic information obtained from each BERT transformer block with structural information obtained with a Hypergraph Neural Network (HGNN) block.

Semantic Representation. The semantic representations of node textual attributes are encoded by a BERT (Devlin et al., 2019) encoder. Specifically, BERT consists of alternating layers of multi-head self-attention (MHA) and Fully-connected Forward Network (FFN) blocks, as in the original Transformer (Vaswani et al., 2017) architecture. Before every block, Layer Normalisation (LN) is applied, and after every block, a residual connection is added. More formally, in the ℓth encoder layer, the hidden states are represented as $\mathcal{X}_{\mathcal{S}}^\ell = \{x_1^\ell, \ldots, x_N^\ell\}$, where N is the maximum length of the inputs. First, a MHA block maps $\mathcal{X}$ into a query matrix $\mathbf{Q} \in \mathbb{R}^{n \times d_k}$, key matrix $\mathbf{K} \in \mathbb{R}^{n \times d_k}$ and value matrix $\mathbf{V} \in \mathbb{R}^{n \times d_v}$, where m is the number of query vectors, and n the number of key/value vectors. Then, an attention vector is calculated as follows

$$\mathbf{Attn(Q, K, V)} = \mathbf{softmax(A)V},$$
$$\mathbf{A} = \frac{\mathbf{QK^T}}{\sqrt{\mathbf{d_k}}} \qquad (1)$$

In practice, the MHA block calculates the self-attention over h heads, where each head i is independently parametrized by $\mathbf{W_i^Q} \in \mathbb{R}^{d_m \times d_k}$, $\mathbf{W_i^K} \in \mathbb{R}^{d_m \times d_k}$ and $\mathbf{W_i^V} \in \mathbb{R}^{d_m \times d_v}$, mapping the input embeddings $\mathcal{X}$ into queries and key-value pairs. Then, the attention for each head is calculated and concatenated, as follows

$$\mathbf{Head_i} = \mathbf{Attn(QW_i^Q, KW_i^K, VW_i^V)}$$
$$\mathbf{MHA}(\mathcal{X}_{\mathcal{S}}^\ell) = \mathbf{Concat(Head_1, \ldots, Head_h)W^U}$$
$$\bar{\mathcal{X}}_{\mathcal{S}}^\ell = \mathbf{MHA}(\mathcal{X}_{\mathcal{S}}^\ell)$$
$$(2)$$

where $\mathbf{W^U} \in \mathbb{R}^{d_m^h \times d_m}$ is a trainable parameter matrix. Next, to acquire the semantic hidden states of the input, a FFN block is applied, as follows

$$\mathbf{FFN}(\bar{\mathcal{X}}_{\mathcal{S}}^\ell) = \mathbf{max(0, \bar{\mathcal{X}}_{\mathcal{S}}^\ell W_1 + b_1)W_2 + b_2}$$
$$(3)$$

where $\mathbf{W_1} \in \mathbb{R}^{d_m \times d_{ff}}$ and $\mathbf{W_2} \in \mathbb{R}^{d_{ff} \times d_m}$ are linear weight matrices. Finally, layer normalisation and residual connection are applied as follows

$$\tilde{\mathcal{X}}_{\mathcal{S}}^\ell = \mathbf{LayerNorm}(\bar{\mathcal{X}}_{\mathcal{S}}^\ell + \mathbf{FFN}(\bar{\mathcal{X}}_{\mathcal{S}}^\ell)) \qquad (4)$$

Therefore, after the L encoder layers, we obtain the nodes' textual description semantic representations as the pooled feature tensor of the [CLS] token, denoted as $\tilde{\mathcal{X}}_{\mathcal{S}}^\ell$.

Hypergraph Structural Representation. In each HyperBERT layer, structural representations are produced through a Hypergraph Neural Network (HGNN) (Feng et al., 2019) over the node-centered context hypergraph $\mathcal{G}_i$, defined as the sub-hypergraph containing only the hyperedges which
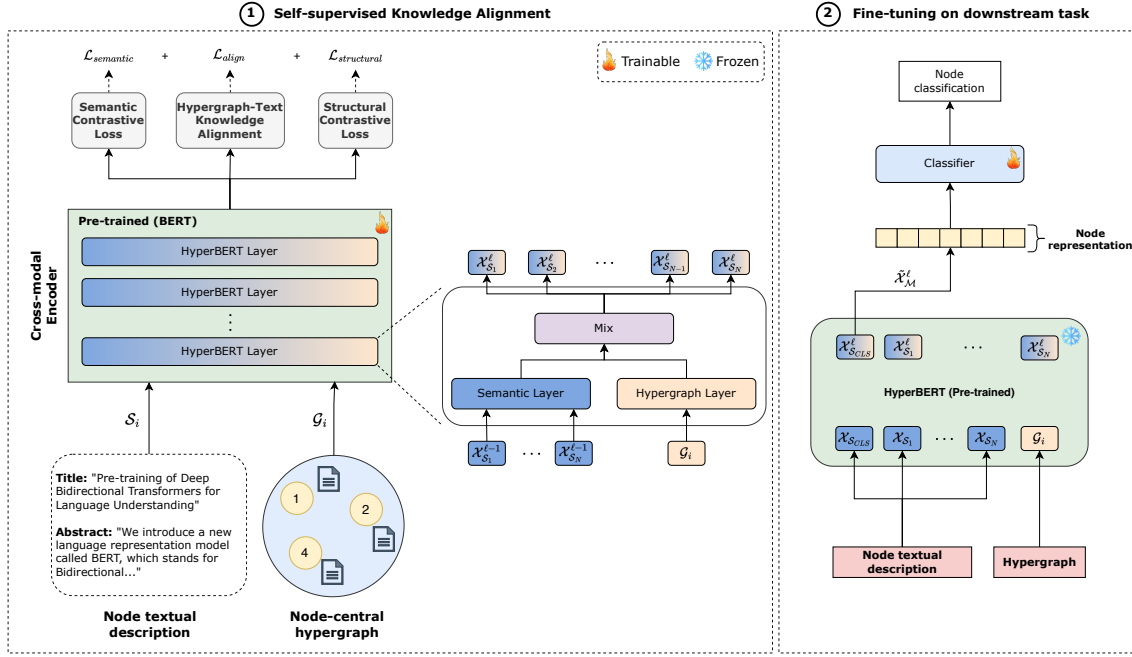
Figure 2: High-level overview of our proposed HyperBERT model. The model mixes hypergraph-aware layers into BERT to simultaneously exploit hypergraph topology and text semantics for node classification tasks on TAHGs. To accomplish the text-hypergraph alignment in the feature space, HyperBERT employs a novel self-supervised loss for pretraining that effectively aligns semantic and hypergraph feature spaces. After pretraining the model, it can be fine-tuned on a variety of downstream hypergraph node-level tasks such as node classification.

node i belongs to, and its incidence matrix H_i. Formally, we formulate the HGNN block to obtain hypergraph node embeddings as follows

$$\tilde{\mathcal{X}}_{\mathcal{G}}^{\ell} = \sigma(\mathbf{D}^{-1}\mathbf{H}\mathbf{W}\mathbf{B}^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{X}^{(\ell-1)}\mathbf{\Psi}), \quad (5)$$

where $\mathbf{D}$ and $\mathbf{B}$ are the degree matrices of the vertex and hyperedge in the hypergraph, respectively. $\sigma(\cdot)$ is a non-linear activation function such as ReLU (Maas, 2013). $\mathbf{\Psi} \in \mathbb{R}^{d_\ell \times d_{(\ell-1)}}$ is a learnable weight matrix between the ℓth and $(\ell-1)^{th}$ layer. Consequently, after L hypergraph layers, we obtain the nodes' structural representations as $\tilde{\mathcal{X}}_{\mathcal{G}}^{\ell}$.

Text-Hypergraph Joint Representation. After computing representations from both semantic and hypergraph structural spaces, the lth HyperBERT layer computes a mixed text-hypergraph representation as follows

$$\tilde{\mathcal{X}}_{\mathcal{M}}^{\ell} = \tilde{\mathcal{X}}_{\mathcal{S}}^{\ell} + \tilde{\mathcal{X}}_{\mathcal{G}}^{\ell}, \quad (6)$$

where $\tilde{\mathcal{X}}_{\mathcal{M}}^{\ell}$ contains a mixture of semantic and hypergraph structural information to enable information flow between both modalities. Specifically, to combine structural and semantic embeddings, we add the hypergraph embedding to the embedding of each of the tokens in $\tilde{\mathcal{X}}_{\mathcal{S}}^{\ell}$. Therefore, $\tilde{\mathcal{X}}_{\mathcal{M}}^{\ell}$

effectively incorporates both the semantic information from text attributes and the structural context from the hypergraph. This is crucial for capturing the complex interplay between textual content and graph topology.

4.2 Hypergraph-Aware Pretraining Task

In order to improve the learning capability of HyperBERT without using any human-annotated labels, we propose a novel hypergraph-aware knowledge alignment pretraining algorithm based on a contrastive learning loss. The proposed hypergraph-aware pretraining task exploits inherent hypergraph knowledge from the TAHG and is applied to both the semantic and structural representations.

Semantic Contrastive Loss. Based on the semantic representation of node i, $\tilde{\mathcal{X}}_{\mathcal{S}_i}$, the other nodes belonging to its same set of hyperedges, $\mathcal{N}(i)$, and the node i excluded mini-batch instances, $\mathcal{B}(i)$, the semantic contrastive loss objective can be defined as follows:

$$\mathcal{L}_{\text{semantic}} = \frac{-1}{|\mathcal{N}(i)|} \sum_{p \in \mathcal{N}(i)} \log \frac{e^{(\tilde{\mathcal{X}}_{\mathcal{S}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{S}_p}/\tau)}}{\sum\limits_{j \in \mathcal{B}(i)} e^{(\tilde{\mathcal{X}}_{\mathcal{S}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{S}_j}/\tau)}},$$

$$(7)$$

where τ denotes the temperature and $e(\cdot)$ represents the exponential function. Here for node i, we consider its semantic representation $\tilde{\mathcal{X}}_{\mathcal{S}_i}$ as the query instance. The positive instances are the representations of node i's hyperedge members $\{\tilde{\mathcal{X}}_{\mathcal{S}_p} \mid p \in \mathcal{N}(i)\}$. Meanwhile, the negative instances are the representations of other text nodes excluding i within the same mini-batch $\{\tilde{\mathcal{X}}_{\mathcal{S}_j} \mid j \in \mathcal{B}(i)\}$.

Structural Contrastive Loss. Similar to the semantic representation, we also apply our contrastive learning algorithm to the hypergraph structural feature space representation of node i, denoted as $\tilde{\mathcal{X}}_{\mathcal{G}_i}$. Formally, the structural contrastive loss is formulated as follows:

$$\mathcal{L}_{structural} = \frac{-1}{|\mathcal{N}(i)|} \sum_{p \in \mathcal{N}(i)} \log \frac{e^{(\tilde{\mathcal{X}}_{\mathcal{G}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{G}_p}/\tau)}}{\sum\limits_{j \in \mathcal{B}(i)} e^{(\tilde{\mathcal{X}}_{\mathcal{G}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{G}_j}/\tau)}},$$

$$(8)$$

where $\tilde{\mathcal{X}}_{\mathcal{G}_i}$ is the query instance. The positive instances are $\{\tilde{\mathcal{X}}_{\mathcal{G}_p} \mid p \in \mathcal{N}(i)\}$ and the negative instances are $\{\tilde{\mathcal{X}}_{\mathcal{G}_j} \mid j \in \mathcal{B}(i)\}$.

Therefore, our semantic and structural contrastive learning losses ensure nodes belonging to the same hyperedges to share similar representations, which inherently elicits informative hypergraph knowledge during the learning process.

Hypergraph-Text Knowledge Alignment. In this work, our goal is to learn expressive representations that simultaneously encode informative textual semantics within each text node, as well as structural information among hyperedges. However, individually performing contrastive learning on either the semantic or structural spaces is not enough due to the lack of information exchange between both modalities. To better align the knowledge captured by the semantic and hypergraph structural layers, we propose a hypergraph-text knowledge alignment algorithm for TAHGs.

In particular, for each node i, based on its semantic and structural representations, $\tilde{\mathcal{X}}_{\mathcal{S}_i}$ and $\tilde{\mathcal{X}}_{\mathcal{G}_i}$, respectively, we formulate the objective of hypergraph-text knowledge alignment loss, $\mathcal{L}_{align}$, as follows:

$$\mathcal{L}_{align} = \frac{-1}{|\mathcal{N}(i)|} \sum_{p \in \mathcal{N}(i)} \left(\mathcal{L}_{align_1} + \mathcal{L}_{align_2} \right)/2,$$

$$\mathcal{L}_{align_1} = \log \frac{e^{(\tilde{\mathcal{X}}_{\mathcal{G}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{S}_p}/\tau)}}{\sum\limits_{j \in \mathcal{B}(i)} e^{(\tilde{\mathcal{X}}_{\mathcal{G}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{S}_j}/\tau)}},$$

$$\mathcal{L}_{align_2} = \log \frac{e^{(\tilde{\mathcal{X}}_{\mathcal{S}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{G}_p}/\tau)}}{\sum\limits_{j \in \mathcal{B}(i)} e^{(\tilde{\mathcal{X}}_{\mathcal{S}_i} \cdot \tilde{\mathcal{X}}_{\mathcal{G}_j}/\tau)}}$$

$$(9)$$

Given the above formulation, in the first component of the knowledge alignment loss function, $\mathcal{L}_{align_1}$, we treat the structural representation of node i, $\tilde{\mathcal{X}}_{\mathcal{G}_i}$, as the query, then construct the positive and negative instances based on the semantic representations. Specifically, the positive instances include both the representation of node i as well as the representations of i's same hyperedge nodes (i.e., $\{\tilde{\mathcal{X}}_{\mathcal{S}_p} \mid p \in \widetilde{\mathcal{N}}(i)\}$), and the negative instances are the representations of other instances within the same mini-batch $\{\tilde{\mathcal{X}}_{\mathcal{S}_j} \mid j \in \mathcal{B}(i)\}$. In the second component of the knowledge alignment loss, $\mathcal{L}_{align_2}$, we consider the semantic representation $\tilde{\mathcal{X}}_{\mathcal{S}_i}$ as the query and construct its corresponding positive and negative instances in the parallel way as for the first loss component. Consequently, the proposed $\mathcal{L}_{align}$ alignment loss encourages the representations of the same node learned across the two separate feature spaces, semantic and structural, to be pulled together in the embedding space.

Learning Objective. In order to learn our hypergraph-aware language model, we jointly optimize the proposed semantic, structural and hypergraph-text knowledge alignment losses. Specifically, the overall training loss objective is formulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{semantic} + \lambda_2 \mathcal{L}_{structural} + \lambda_3 \mathcal{L}_{align}, \quad (10)$$

where $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$. In practice, we found that the values of $\lambda_1 = \lambda_2 = \lambda_3 = 1$ led to good experimental results.

4.3 Fine-Tuning in Downstream Tasks

Once the pretraining is finished, we freeze the parameters of HyperBERT and use it to compute representations of each text node. In particular, during fine-tuning HyperBERT computes the text-hypergraph joint representation, $\tilde{\mathcal{X}}_{\mathcal{M}}^{\ell}$, capturing

an aligned representation of the node's textual attributes and its topological structure in the hypergraph. Furthermore, this representation is used to fine-tune separate models on different downstream tasks. Specifically, we train a linear classifier from the node features to class labels for node classification using the cross entropy loss.

5 Experiments

In this section we show our model performance on five hypergraph node classification benchmarks from a variety of domains. Also, we present ablation studies to analyse the importance of the different components of the HyperBERT model.

5.1 Datasets and Evaluation Metrics

For our experiments we consider five datasets, with complete details included in Appendix A. The hypergraph datasets are from diverse domains and sizes, expressing co-citation, co-authorship and movie-actor relations. Following previous works on hypergraph node classification (Wang et al., 2023; Kim et al., 2024), we calculate classification accuracy in all our experiments. We evaluate all models over 10 runs and report the mean and standard deviation of the test set accuracy averaged across them.

5.2 Overall Performance

We first conducted experiments to evaluate model performance on text-attributed hypergraph node classification and present the results in Table 1. As shown in the table, our model HyperBERT outperforms all the baselines on the five hypergraph evaluation datasets, so demonstrating superior capability in text-attributed hypergraph node classification.

On the co-citation datasets, compared with AllSetTransformer (Chien et al., 2022), our model's classification accuracy is 64.5% for the Cora dataset, achieving 16.9% absolute performance improvement (47.6% vs 64.5%). In the case of PubMed, HyperBERT achieves 78.9% classification accuracy, which when compared with AllSet-Transformer (72.4%) attains a 6.5% absolute accuracy increment. When compared with more recent methods that make use of self-supervised techniques for training the model, such as HypeBoy (Kim et al., 2024), our model's classification accuracy is 64.5% in Cora, a +2.2% improvement from 62.3%, and 78.9% in PubMed, a +1.9% increase from 77.0%. This effectively shows the

benefit of our proposed architecture for solving text-attributed hypergraph node classification tasks.

In the co-authorship dataset experiments, when compared with the AllSetTransformer, Hyper-BERT attains an accuracy of 82.3% (+17%) and 69.2% (+11.7%) in the DBLP-A and Cora-CA datasets, respectively. In comparison with Hype-Boy, HyperBERT obtains +1.7% and +2.9% performance improvements in the DBLP-A and Cora-CA datasets, respectively. Furthermore, our model reaches a performance in the IMDB movies dataset of 49.7%, an absolute improvement of 7.4% and 2.1% when compared with AllSetTransformer and HypeBoy, respectively.

5.3 Ablation Study

To investigate the contribution of each module in HyperBERT, we conduct several ablation studies to isolate and measure the impact of individual components on overall performance.

Effect of Architectural Components. In order to validate the architectural design of HyperBERT, we explore the effect of its different components towards node classification accuracy in the Pubmed dataset, and provide the results in Table 2. On the one hand, we investigate suppressing particular components of the loss function, such as the semantic loss, structural loss, and alignment loss. To do so, we set their respective λ factors to 0, effectively cancelling their contribution during the pretraining stage. We find that these loss components lead to significant performance decreases when removed. Specifically, accuracy decreases to 66.4% (-12.5%), 68.8% (-10.1%) and 63.1% (-15.8%) when removing the semantic, structural and alignment losses, respectively. As expected, the biggest performance impact is given when eliminating the alignment loss, as it leads to completely disparate feature spaces for text and hypergraph node representations. We also study the impact of neglecting the high-order topology by replacing the HGNN encoder with a GNN, decreasing performance to 71.2% (-7.7%). In addition, we show the effect of not pretraining the model and instead training it from scratch on the downstream task, leading to a performance decrease of 71.5% (-7.4%). Therefore, the results of these experiments justify our architectural choices.

Impact of Hypergraph Encoder. The choice of hypergraph encoder plays a crucial role in the

Table 1: Performance of HyperBERT on a variety of hypergraph node classification benchmarks. Cell values indicate mean and standard deviation accuracy calculated over 10 runs. Underscoring depicts the best performance for each dataset across previous methods. Bold indicates the best overall method for each dataset.

Method	Co-citation		Co-authorship		Movies
	Cora	Pubmed	DBLP-A	Cora-CA	IMDB
HGNN (Feng et al., 2019)	50.0 (7.2)	72.9 (5.0)	67.1 (6.0)	50.2 (5.7)	42.2 (2.9)
HyperGCN (Yadati et al., 2019)	33.1 (10.2)	63.5 (14.4)	68.2 (14.4)	50.2 (5.7)	37.9 (4.5)
HNHN (Dong et al., 2020)	50.0 (7.9)	72.1 (5.4)	62.6 (4.8)	48.3 (6.2)	42.3 (3.4)
UniGCN (Huang and Yang, 2021)	49.1 (8.4)	74.4 (3.9)	65.1 (4.7)	51.3 (6.3)	41.6 (3.5)
UniGIN (Huang and Yang, 2021)	47.8 (7.7)	69.8 (5.6)	63.4 (5.1)	48.3 (6.1)	41.4 (2.7)
UniGCNII (Huang and Yang, 2021)	48.5 (7.4)	74.1 (3.9)	65.8 (3.9)	54.8 (7.5)	42.5 (3.9)
AllSetTransformer (Chien et al., 2022)	47.6 (4.2)	72.4 (4.5)	65.3 (3.9)	57.5 (5.7)	42.3 (2.4)
HyperGCL (Wei et al., 2022)	60.3 (7.4)	76.8 (3.7)	79.7 (3.8)	62.0 (5.1)	43.9 (3.6)
H-GD (Zheng et al., 2022)	50.6 (8.2)	74.5 (3.5)	75.1 (3.6)	58.8 (6.2)	43.0 (3.3)
ED-HNN (Wang et al., 2023)	47.6 (7.7)	72.7 (4.7)	65.8 (4.8)	54.8 (5.4)	41.4 (3.0)
HypeBoy (Kim et al., 2024)	<u>62.3</u> (7.7)	<u>77.0</u> (3.4)	<u>80.6</u> (2.3)	<u>66.3</u> (4.6)	<u>47.6</u> (2.5)
HyperBERT (Ours)	**64.5** (4.5)	**78.9** (3.1)	**82.3** (1.7)	**69.2** (4.1)	**49.7** (1.8)

Method	Accuracy (%)
HyperBERT w/o semantic loss	66.4 ± 1.3
HyperBERT w/o structural loss	68.8 ± 1.6
HyperBERT w/o alignment loss	63.1 ± 2.3
HyperBERT with GNN as structural layer	71.2 ± 3.5
HyperBERT w/o pretraining	71.5 ± 1.5
HyperBERT	**78.9 ± 3.1**

Table 2: Effect of the different architectural components in node classification accuracy (and ± 95% confidence interval) of HyperBERT on the Pubmed benchmark.

Encoder	Accuracy (%)
HyperGCN (Yadati et al., 2019)	76.3 ± 2.5
HNHN (Dong et al., 2020)	76.9 ± 1.8
UniGCN (Huang and Yang, 2021)	77.6 ± 2.3
UniGIN (Huang and Yang, 2021)	77.5 ± 2.1
UniGCNII (Huang and Yang, 2021)	77.9 ± 3.4
ED-HNN (Wang et al., 2023)	78.2 ± 2.6
HGNN (Feng et al., 2019)	**78.9 ± 3.1**

Table 3: Accuracy of node classification on the PubMed test set using different hypergraph architectures as hypergraph encoder layers.

computation of the hypergraph structural representations in HyperBERT. For this reason, Table 3 shows the node classification results in the PubMed dataset when using several different hypergraph architectures as hypergraph encoder layers while keeping BERT as the text encoder. As can be observed, the highest performance is achieved when using HGNN as the hypergraph encoder, with 78.9% (± 3.1) accuracy, whereas the lowest is attained when using HyperGCN, leading to an accuracy of 76.3% (± 2.5). These results highlight the significant impact that the choice of hypergraph encoder can have on model performance, suggesting that careful selection of the encoder architecture is essential for optimizing node classification accuracy.

Impact of Text Encoder. The choice of text encoder is crucial in determining the overall performance of HyperBERT, particularly when the hypergraph encoder is fixed to HGNN. Table 4 presents the node classification results on the PubMed dataset using different text encoders while keeping HGNN as the hypergraph encoder. The highest performance is achieved with BERT, showing an

accuracy of 78.9% (± 3.1), while ELECTRA and RoBERTa demonstrate slightly lower accuracies of 78.4% (± 2.9) and 78.2% (± 3.0), respectively. These results indicate that while all three text encoders provide competitive performance, BERT remains the most effective choice in this setup.

Encoder	Accuracy (%)
RoBERTa (Liu et al., 2019)	78.2 ± 3.0
ELECTRA (Clark et al., 2020)	78.4 ± 2.9
BERT (Devlin et al., 2019)	**78.9 ± 3.1**

Table 4: Accuracy of node classification on the PubMed test set using different text encoders while keeping HGNN as the hypergraph encoder.

6 Conclusion

In this work, we propose a new model architecture to improve the capability of hypergraph structural encoding of BERT effectively while retaining its semantic encoding abilities. In order to achieve this, we designed HyperBERT, a hypergraph-aware language model based on BERT that augments with

structural context for the challenging task of text-attributed hypergraph node classification. Notably, extensive experiments on five benchmark hypergraph datasets demonstrate the effectiveness of HyperBERT for text-attributed hypergraph node classification tasks, proving that simultaneously modelling hypergraph structure and semantic context is crucial to achieve state-of-the-art results.

Limitations

In this work we evaluate HyperBERT primarily on node classification tasks. However, there are other relevant downstream tasks, such as hyperedge prediction. Future work would benefit from investigating the applicability and capacity of HyperBERT to broader tasks.

References

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

Aleix Bassolas, Anton Holmgren, Antoine Marot, Martin Rosvall, and Vincenzo Nicosia. 2022. Mapping nonlocal relationships between metadata and network structure with metadata-dependent encoding of random walks. *Science Advances*, 8(43):eabn7558.

Smriti Bhagat, Graham Cormode, and S. Muthukrishnan. 2011. Node classification in social networks. *ArXiv*, abs/1101.3291.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olgica Milenkovic, and Inderjit S. Dhillon. 2021. Node feature extraction by self-supervised multi-scale neighborhood prediction. *ArXiv*, abs/2111.00064.

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

Petr Chunaev. 2020. Community detection in node-attributed social networks: A survey. *Computer Science Review*, 37:100286.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Martina Contisciani, Eleanor A. Power, and Caterina De Bacco. 2020. Community detection with node attributes in multilayer networks. *Scientific Reports*, 10(1).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Yihe Dong, Will Sawin, and Yoshua Bengio. 2020. Hnhn: Hypergraph networks with hyperedge neurons. *ICML Graph Representation Learning and Beyond Workshop*.

Keyu Duan, Qian Liu, Tat-Seng Chua, Shuicheng Yan, Wei Tsang Ooi, Qizhe Xie, and Junxian He. 2023. Simteg: A frustratingly simple approach improves textual graph learning.

Andrea Failla, Salvatore Citraro, and Giulio Rossetti. 2023. Attributed stream hypergraphs: temporal modeling of node-attributed high-order interactions. *Applied Network Science*, 8(1).

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3558–3565.

Matthias Fey and Jan Eric Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. ArXiv:1903.02428 [cs, stat].

Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. 2022. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2548–2566.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS*.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2023. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning.

Jian Hou, Marcello Pelillo, and Huaqiang Yuan. 2022. Hypergraph matching via game-theoretic hypergraph clustering. *Pattern Recognition*, 125:108526.

Jian Hou, Huaqiang Yuan, and Marcello Pelillo. 2023. Game-theoretic hypergraph matching with density enhancement. *Pattern Recognition*, 133:109035.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2021. Open graph benchmark: Datasets for machine learning on graphs.

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*.

Vassilis N. Ioannidis, Xiang Song, Da Zheng, Houyu Zhang, Jun Ma, Yi Xu, Belinda Zeng, Trishul Chilimbi, and George Karypis. 2022. Efficient and effective training of language and graph neural network models.

Bowen Jin, Yu Zhang, Yu Meng, and Jiawei Han. 2023a. Edgeformers: Graph-empowered transformers for representation learning on textual-edge networks. In *International Conference on Learning Representations*.

Bowen Jin, Yu Zhang, Qi Zhu, and Jiawei Han. 2023b. Heterformer: Transformer-based deep node representation learning on heterogeneous text-rich networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1020–1031.

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*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Haoyu Kuang, Jiarong Xu, Haozhe Zhang, Zuyu Zhao, Qi Zhang, Xuanjing Huang, and Zhongyu Wei. 2023. Unleashing the power of language models in text-attributed graph. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8429–8441, Singapore. Association for Computational Linguistics.

Yiran Li, Renchi Yang, and Jieming Shi. 2023. Efficient and effective attributed hypergraph clustering via k-nearest neighbor augmentation. *Proc. ACM Manag. Data*, 1(2).

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2901–2908.

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.

Jose Lugo-Martinez, Daniel Zeiberg, Thomas Gaudelet, Noël Malod-Dognin, Natasa Przulj, and Predrag Radivojac. 2020. Classification in biological networks with hypergraphlet kernels. *Bioinformatics*, 37(7):1000–1007.

Andrew L. Maas. 2013. Rectifier nonlinearities improve neural network acoustic models.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Kazuki Nakajima and Takeaki Uno. 2024. Inferring community structure in attributed hypergraphs using stochastic block models.

M. E. J. Newman. 2001. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409.

Raj Kumar Pan, Kimmo Kaski, and Santo Fortunato. 2012. World citation and collaboration networks: uncovering the role of geography in science. *Scientific Reports*, 2(1).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. ArXiv:1912.01703 [cs, stat].

Alice Patania, Giovanni Petri, and Francesco Vaccarino. 2017. The shape of collaborations. *EPJ Data Science*, 6(1).

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Sebastiano Smaniotto and Marcello Pelillo. 2021. Two metrics for attributed hypergraphs. *Pattern Recognition Letters*, 149:143–149.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. ArXiv:1706.03762 [cs].

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item recommendation with sequential hypergraphs. In *SIGIR*.

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

Tianxin Wei, Yuning You, Tianlong Chen, Yang Shen, Jingrui He, and Zhangyang Wang. 2022. Augmentations in hypergraph contrastive learning: Fabricated and generative. *arXiv preprint arXiv:2210.03801*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Hanrui Wu, Jinyi Long, Nuosi Li, Dahai Yu, and Michael K. Ng. 2022. Adversarial auto-encoder domain adaptation for cold-start recommendation with positive and negative hypergraphs. *ACM Trans. Inf. Syst.*, 41(2).

Hanrui Wu and Michael K. Ng. 2022. Hypergraph convolution on nodes-hyperedges network for semi-supervised node classification. *ACM Trans. Knowl. Discov. Data*, 16(4).

Hanrui Wu, Chung Wang Wong, Jia Zhang, Yuguang Yan, Dahai Yu, Jinyi Long, and Michael K. Ng. 2023a. Cold-start next-item recommendation by user-item matching and auto-encoders. *IEEE Transactions on Services Computing*, 16(4):2477–2489.

Hanrui Wu, Yuguang Yan, and Michael Kwok-Po Ng. 2023b. Hypergraph collaborative network on vertices and hyperedges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3245–3258.

Longcan Wu, Daling Wang, Kaisong Song, Shi Feng, Yifei Zhang, and Ge Yu. 2021. Dual-view hypergraph neural networks for attributed graph learning. *Knowledge-Based Systems*, 227:107185.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *International Conference on Learning Representations*.

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*. Curran Associates Inc., Red Hook, NY, USA.

Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudre-Mauroux. 2019. Revisiting user mobility and social relationships in lbsns: A hypergraph embedding approach. In *The World Wide Web Conference*, WWW '19, page 2147–2157, New York, NY, USA. Association for Computing Machinery.

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. In *Advances in Neural Information Processing Systems*, volume 34, pages 28798–28810. Curran Associates, Inc.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion.

Jiong Zhang, Wei-Cheng Chang, Hsiang-Fu Yu, and Inderjit S Dhillon. 2021a. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. In *Advances in Neural Information Processing Systems*.

Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 5171–5181, Red Hook, NY, USA. Curran Associates Inc.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2021b. Greaselm: Graph reasoning enhanced language models. In *International Conference on Learning Representations*.

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(1–4):43–52.

Zizhao Zhang, Haojie Lin, Xibin Zhao, Rongrong Ji, and Yue Gao. 2018. Inductive multi-hypergraph learning and its application on view-based 3d object classification. *IEEE Transactions on Image Processing*, 27(12):5957–5968.

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 *The Eleventh International Conference on Learning Representations*.

Yizhen Zheng, Shirui Pan, Vincent Lee, Yu Zheng, and Philip S. Yu. 2022. Rethinking and scaling up graph contrastive learning: An extremely efficient approach with group discrimination. In *Advances in Neural Information Processing Systems*.

Jason Zhu, Yanling Cui, Yuming Liu, Hao Sun, Xue Li, Markus Pelger, Tianqi Yang, Liangjie Zhang, Ruofei Zhang, and Huasha Zhao. 2021. Textgnn: Improving text encoder via graph neural network in sponsored search. In *Proceedings of the Web Conference 2021*, WWW '21, page 2848–2857, New York, NY, USA. Association for Computing Machinery.

Appendix

A Datasets

For our experiments we use five publicly available benchmark hypergraph datasets. The hypergraph datasets are from diverse domains, expressing co-citation, co-authorship and movie-actor relations. Table 5 provides statistics on the datasets.

Dataset	Nodes	Hyperedges	# Classes
Cora	1,434	1,579	7
Pubmed	3,840	7,963	3
DBLP-A	2,591	2,690	6
Cora-CA	2,388	1,072	7
IMDB	3,939	2,015	3

Table 5: Statistics of the text-attributed hypergraph datasets used in our experiments.

We utilize two co-citation datasets: Cora and Pubmed. In these datasets, each node represents a publication and hyperedges represent sets of publications co-cited by particular publications. For example, if a publication has cited other nodes (publications) v_i, v_j and v_k, these nodes are grouped as a hyperedge $\{v_i, v_j, v_k\} \in \mathcal{E}$. Node classes indicate categories of the publications. Moreover, for co-authorship datasets we utilize Cora-CA and DBLP-A. In Cora-CA and DBLP-A, each node represents a publication, and a set of publications that are written by a particular author is grouped as a hyperedge. Classes indicate categories of the publication. We use IMDB for a movie-actor dataset. In this dataset, each node indicates a movie, and the movies that a particular actor has acted on are grouped as a hyperedge. Node attributes describe the plot of the movie and classes indicate the genre of the movie.

B Implementation Details

We implemented HyperBERT with PyTorch (Paszke et al., 2019) and the HuggingFace library (Wolf et al., 2020). For the hypergraph neural network components, we use PyTorch Geometric (Fey and Lenssen, 2019). For the knowledge alignment stage, we set batch size as 32, and training steps to 20,000 on a single NVIDIA A100 GPU. We utilize the Adam optimizer with a fixed weight decay rate of 10^{-6} and learning rate of 10^{-3}. The number of layers for HyperBERT is set to 6, number of heads is 8 and dropout rate is 0.5. The dimensionality is set to 512. The temperature parameter τ contained in our loss, is set to 0.2. For training on downstream tasks, we use 200 epochs with an output MLP for node classification containing 2 layers and dimensionality of 512. We evaluate validation accuracy of the model every 10 epochs and keep the model checkpoint that yields the best validation accuracy, then we use it for predicting on the test dataset. Results are on the test set unless stated otherwise.