Text Enriched Sparse Hyperbolic Graph Convolutional Networks

Nurendra Choudhary, Nikhil Rao, Karthik Subbian, Chandan K. Reddy

Abstract—Heterogeneous networks, which connect informative nodes containing text with different edge types, are routinely used to store and process information in various real-world applications. Graph Neural Networks (GNNs) and their hyperbolic variants provide a promising approach to encode such networks in a low-dimensional latent space through neighborhood aggregation and hierarchical feature extraction, respectively. However, these approaches typically ignore metapath structures and the available semantic information. Furthermore, these approaches are sensitive to the noise present in the training data. To tackle these limitations, in this paper, we propose Text Enriched Sparse Hyperbolic Graph Convolution Network (TESH-GCN) to capture the graph’s metapath structures using semantic signals and further improve prediction in large heterogeneous graphs. In TESH-GCN, we extract semantic node information, which successively acts as a connection signal to extract relevant nodes’ local neighborhood and graph-level metapath features from the sparse adjacency tensor in a reformulated hyperbolic graph convolution layer. These extracted features in conjunction with semantic features from the language model (for robustness) are used for the final downstream task. Experiments on various heterogeneous graph datasets show that our model outperforms the current state-of-the-art approaches by a large margin on the task of link prediction. We also report a reduction in both the training time and model parameters compared to the existing hyperbolic approaches through a reformulated hyperbolic graph convolution. Furthermore, we illustrate the robustness of our model by experimenting with different levels of simulated noise in both the graph structure and text, and also, present a mechanism to explain TESH-GCN’s prediction by analyzing the extracted metapaths.

Index Terms—Hyperbolic space, heterogeneous networks, sparsity, link prediction

1 INTRODUCTION

HETEROGENEOUS networks, which connect informative nodes containing text with different edge types, are routinely used to store and process information in diverse domains such as e-commerce [1], social networks [2], medicine [3], and citation networks [4]. The importance of these domains and the prevalence of graph datasets linking textual information has resulted in the rise of Graph Neural Networks (GNNs) and their variants. These GNN-based methods aim to learn a node representation as a composition of the representations of nodes in their multi-hop neighborhood, either via random walks [5], [6], neural aggregations [7], [8], [9], or Boolean operations [10]. However, basic GNN models only leverage the structural information from a node’s local neighborhood, and thus do not exploit the full extent of the graph structure (i.e., the global context) or the node content. In the context of e-commerce search, based on a consumer’s purchase of “[brand1] shoes”, it is difficult to identify if they would also purchase “[brand2] shoes” or “[brand1] watch” merely on the basis of the products’ nearest graph neighbors, however, global information on purchase behavior could provide additional information in identifying and modeling such purchase patterns. Analysis into such limitations has led to research into several alternatives that capture additional information such as hyperbolic variants [11], [12] to capture the latent hierarchical relations and hybrid models [13], [14] to leverage additional text information from the nodes in the graph. In spite of their preliminary success, these aforementioned techniques fundamentally suffer from several critical limitations such as non-scalability and lack of robustness to noise in real-world graphs when applied in practice. Certain other attempts on aggregating a graph’s structural information [15] utilize graph metrics such as centrality encoding and sibling distance to show improved performance over other approaches. However, there is an exhaustive set of graph metrics and manually incorporating every one of them is impractical. Hence, practitioners need a better approach to automatically detect the most relevant graph features that aid the downstream tasks. For example, metapaths, heterogeneous paths between different nodes that preserve long-distance relations, are traditionally found to be good message passing paths in several graph problems [16]. However, they are only aggregated locally due to computational constraints. The adjacency tensor of a heterogeneous graph can be used to extract both metapath information as well as aggregate local neighborhood features. Efficiently encoding the entire adjacency tensor in training graph neural models can thus help capture all relevant metapath features.

In addition to this, the nodes in the graph datasets also contain auxiliary information in different modalities (generally text) such as product descriptions in e-commerce graphs and article titles in citation networks. Such textual content can be encoded using popular transformer models [17], and consequently serve as an additional source of information. Thus, integrating these transformer models in

---

1 for a homogeneous graph, it will be a matrix
Leveraging hierarchical structures and metapaths help us distinguish between items that are complementary (also buy) or alternatives (also view) of each other.

Integrating semantic content with product features allows us to match different products in the catalogue with the query “[brand1] footwear”.

Product search requires robustness to noise in the hierarchical product graph structure caused by miscategorized items.

Fig. 1: Challenges of graph representation learning in the E-commerce domain.

1) Leveraging metapath structures: Existing GNN frameworks aggregate information only from a local neighborhood of the graph and do not possess the ability to aggregate graph-level metapath structures. However, graph-level information can aid in several graph analysis tasks where node’s local neighborhood information is insufficient, e.g., in Figure 1a, we note that local node-level information is unable to distinguish between the relations of “also buy” and “also view”, whereas, graph-level information allows us to make the differentiation. Indeed, when attempting to combine information from the entire graph, existing methods suffer from over-smoothness [18]. Moreover, the size of modern graph datasets renders aggregating information from the full graph infeasible.

2) Incorporating hierarchical structures: Most of the real-world graphs have inherent hierarchies, which are best represented in a hyperbolic space (rather than the traditional Euclidean space), for e.g., the product hierarchy shown in Figure 1a. However, existing hyperbolic GNNs [11], [12] do not leverage the full graph when aggregating information due to both mathematical and computational challenges.

3) Integrating textual (semantic) content: Previous methods for integrating semantic information of the nodes are relatively ad-hoc in nature. For example, they initialize their node representations with text embeddings for message aggregation in the GNNs [13]. Such methods fix the semantic features and do not allow the framework to learn task-specific embeddings directly from the nodes’ original content, e.g., in Figure 1b, the product tokens “sneakers” and “sandals” are closer to the query token “footwear” in the e-commerce domain which is not the case in a broader semantic context.

4) Robustness to noise: Real-world graphs are susceptible to noise and hence require robust graph representation learning mechanisms, especially in the presence of multiple forms of data (i.e., graph structure and textual content), e.g., in Figure 1c, we observe that the task of product search is susceptible to noise in the product catalogue due to miscategorized items. Previous approaches do not leverage the complementary nature of graphs and text to improve robustness to noise in both of these modalities.

To tackle the above challenges, we introduce Text Enriched Sparse Hyperbolic Graph Convolution Network (TESH-...
2 RELATED WORK

In this section, we describe earlier works related to our proposed model, primarily in the context of graph representation learning and hyperbolic networks.

2.1 Graph Representation Learning

Early research on graph representations relied on learning effective node representations, primarily, through two broad methods, namely, matrix factorization and random walks. In matrix factorization based approaches, the sparse graph adjacency matrix $A$ is factorized into low-dimensional dense matrix $L$ such that the information loss $\|L^T L - A\|$ is minimized. In the random walk based approaches, a node’s neighborhood is collected with random walks through its edges, and the neighborhood is used to predict the node’s representation in a dense network framework. Earlier methods such as LINE [22] and SDNE [23] use first-order (nodes connected by an edge) and second-order (nodes with similar neighborhood) proximity to learn the node representations. These methods form a vector space model for graphs and have shown some preliminary success. However, they are node-specific and do not consider the neighborhood information of a node or the overall graph structure. In more recent works, aggregating information from a node’s neighborhood is explored using the neural network models. Graph neural networks (GNN) [24], typically applied to node classification, aggregate information from a node’s neighborhood to predict the label for the root node. Several approaches based on different neural network architectures for neighborhood aggregation have been developed in recent years and some of the popular ones include GraphSage [7] (LSTM), Graph Convolution Networks (GCN) [8], and Graph Attention Networks (GAT) [9]. Another line of work specifically tailored for heterogeneous graphs [16, 25, 26], utilizes the rich relational information through metapath aggregation. These approaches, while efficient at aggregating neighborhood information, do not consider the node’s semantic attributes or the global graph structure. In the proposed TESH-GCN model, we aim to utilize the node’s semantic signal, in congruence with global adjacency tensor, to capture both the node’s semantic attributes and its position in the overall graph structure.

2.2 Hyperbolic Networks

In recent research [11], graph datasets have been shown to possess an inherent hierarchy between nodes thus demonstrating a non-Euclidean geometry. In [11], the authors provide the gyrovector space model including the hyperbolic variants of the algebraic operations required to design neural networks. The algebraic operations for the Poincaré ball of curvature $c$ are the following: M"{o}bius addition $(\oplus_c)$, exponential map $(\exp^c)$, logarithmic map $(\log^c)$, M"{o}bius scalar multiplication $(\otimes_c)$, and hyperbolic activation $(\sigma_c)$.

\[
x \oplus_c y = \frac{(1 + 2c(x,y) + c|y|^2)x + (1 - c||x||^2)y}{1 + 2c(x,y) + c^2||x||^2||y||^2}
\]

\[
\exp^c(v) = x \oplus_c \left( \tanh \left( \frac{\sqrt{c} \lambda_c ||v||}{2} \right) \right) \frac{v}{\sqrt{c}||v||}
\]

\[
\log^c(y) = \frac{2}{\sqrt{c}\lambda_c} \tanh^{-1} \left( \sqrt{c} \frac{||x \oplus_c y||}{\sqrt{c} - x \oplus_c y} \right) \frac{-x \oplus_c y}{||x \oplus_c y||}
\]


\[ r \odot c = \exp_\mathbb{H}(r \log_\mathbb{H}(c)), \quad \forall r \in \mathbb{R}, x \in \mathbb{H}^n, \]
\[ \sigma_c(x) = \exp_\mathbb{H}(\sigma(\log_\mathbb{H}(x))) \]

where \( \lambda_c^n = \frac{2}{1 - c^2} \) is the metric conformal factor. Based on these approaches, hyperbolic networks such as HGNN [11], HGCN [12], HAN [27], and HypE [28] have shown to outperform their Euclidean counterparts on graph datasets. However, these approaches still focus on the nodes’ local neighborhood and not the overall graph structure. Furthermore, hyperbolic transformations are performed on entire vectors and are thus inefficient on sparse tensors. In our model, we utilize the \( \beta \)-split and \( \beta \)-concatenation operations [29] to optimize the hyperbolic graph convolution for sparse adjacency tensors.

### 3 The Proposed Model

In this section, we first describe the problem setup for link prediction on sparse heterogeneous graphs. We then provide a detailed explanation of the different components of the proposed model and their functionality in the context of link prediction. The overall architecture is depicted in Figure 3. The notations used in this paper are defined in Table 1.

#### 3.1 Problem Setup

Let us consider a heterogeneous graph \( G = (V, E) \) with \( K \) edge types, where \( v \in V \) is the set of its nodes and \( e_k(v_i, v_j) \in E \in \mathbb{B}^{K \times |V| \times |V|} \) is a sparse Boolean adjacency tensor (which indicates if edge type \( e_k \) exists between nodes \( v_i \) and \( v_j \) or not). Each node \( v \) also contains a corresponding text sequence \( s_i \). The sparsity of the adjacency tensor and hierarchy of the graph \( G \) is quantified by the sparsity ratio \( R \) (Definition 1) and hyperbolicity \( \delta \) (Definition 2), respectively. Higher sparsity ratio implies that \( E \) is sparser, whereas lower hyperbolicity implies \( G \) has more hierarchical relations.

\[ R = \frac{|e_k(v_i, v_j) = 0|}{|E|} \]

**Definition 1.** Sparsity ratio \( R \) is defined as the ratio of the number of zero elements to the total number of elements in the adjacency tensor;

**Definition 2.** For a graph \( G \), the hyperbolicity \( \delta \) is calculated as described in [30]. Let \( \{a, b, c, d\} \in G \) be a set of vertices. Let us define \( S_1, S_2 \) and \( S_3 \) as:

\[ S_1 = \text{dist}(a, b) + \text{dist}(d, c) \]
\[ S_2 = \text{dist}(a, c) + \text{dist}(b, d) \]
\[ S_3 = \text{dist}(a, d) + \text{dist}(b, c) \]

Note that we use link prediction as a running example in this paper. Other tasks (node/graph classification) can be easily performed by changing the loss function.
Let $M_1$ and $M_2$ be the two largest values in $(S_1, S_2, S_3)$, then $H(a, b, c, d) = M_1 - M_2$ and $\delta(G)$ is given by:

$$\delta(G) = \frac{1}{2} \max_{(a,b,c,d) \in G} H(a, b, c, d)$$

For the task of link prediction, given input nodes $v_i$ and $v_j$ with corresponding text sequence $s_i$ and $s_j$, respectively and an incomplete training adjacency tensor $E$, our goal is to train TESH-GCN to optimize a predictor $P_\theta$ parameterized by $\theta$ such that:

$$y_k = P_\theta(z = 1|I)P_\theta(y = k|I), \text{ where } I = \{v_i, v_j, s_i, s_j, E\}, \quad \theta = \arg \min_{\hat{\theta}} \left(-\sum_{k=1}^{K} \hat{y}_k \log (y_k) \right)$$

where $z$ is a Boolean indicator that indicates if an edge between the two nodes exists ($z = 1$) or not ($z = 0$) and $y$ is a class predictor for each $k \in K$ edge types. $\hat{y}_k$ is the probability of each class $k \in K$ predicted by TESH-GCN and $y_k$ is the ground truth class label.

### 3.2 Text Enriched Sparse Hyperbolic GCN

In this section, we describe the message aggregation framework of TESH-GCN, which allows us to aggregate the node’s text-enriched local neighborhood and long metapath features (through semantic signals and reformulated hyperbolic graph convolution) from sparse adjacency tensors in the hyperbolic space. In this section, We detail the (i) methodology of integrating semantic features with graph tensors, (ii) sparse HGCN layer to encode hierarchical and graph structure information efficiently, and (iii) aggregation through self-attention to improve model robustness.

**Incorporating Semantics into Adjacency Tensor:** To integrate the nodes’ textual information with the graph structure, we need to enrich the heterogeneous graph’s adjacency tensor with the nodes’ semantic features. For this, we extract the nodes’ semantic signals using a pre-trained language model ($LM$) [51]. We encode the node’s text sequence $s$ to a vector $t \in \mathbb{R}^{T'}$. Each dimension of vector $t$ denotes a unique semantic feature and thus, each feature needs to be added to a single adjacency matrix. To achieve this efficiently, let us assume that $A_k$ is a stack of $D$ repetitions of the adjacency matrix $c_k$. To each matrix in the stack $A_k$, we add each unique dimension of $t$ to the corresponding matrix as the nodes’ semantic and positional signal particularly for that dimension (illustrated in Figure 4).

$$t_i = LM(s_i), \quad t_j = LM(s_j)$$

$$A_k[d, i, :] = t_i[d], \quad A_k[d, :, j] = t_j[d] \quad \forall d : 1 \rightarrow D$$

where $A_k[d, i, :]$ represents the $i^{th}$ row in the $d^{th}$ matrix of $A_k$ and $A_k[d, :, j]$ represents the $j^{th}$ column in the $d^{th}$ matrix of $A_k$. $t_i[d]$ and $t_j[d]$ are the $d^{th}$ dimension of their respective semantic signals. The update operations given above ensure that the adjacency tensor $A_k$ contains information on the semantic signals at the appropriate position in the graph structure. Thus, an efficient encoding of $A_k$ allows us to capture both the structural information and semantic content of the underlying nodes. We achieve this through the sparse HGCN layer.

**Sparse Hyperbolic Graph Convolution:** To encode the graph structure and latent hierarchy, we need to leverage the adjacency tensor’s sparsity in the hyperbolic space for computational efficiency. To achieve this, we reformulate the hyperbolic graph convolution in the following manner. The graph convolution layer has two operations, namely, feature transformation and aggregation, which are achieved through convolution with a filter map of trainable curvature and pooling, respectively. For a matrix of size $m_r \times m_c$ and filter map $f \times f$, graph convolution requires $\approx (m_r - f) \times (m_c - f)$ operations. However, given the high sparsity of adjacency matrices, operations on zero-valued cells will return zero gradients and, thus not contribute to the learning process. Hence, we only apply the filter transformation to adjacency tensor cells with nonzero values and ignore the zero-valued cells. For an input adjacency tensor with elements $x \in A_k$,

$$o_{p,l} = W_{f,l} \ast x_{p,l-1} \ast c_l \ast b_l \quad \forall x_{p,l-1} \neq 0$$

$$a_{p,l} = \exp_{p,l-1} \left( \sum_{p} \alpha_{p} \log_{c_l}(o_{p,l}) \right)$$

$$h_{p,l} = \sigma_{c_l}(a_{p,l})$$

where $a_{p,l}$ represents the output of feature transformation at the layer $l$ for non-zero input elements $x_{p,l-1}$ of previous layer’s $l-1$ adjacency tensor with learnable feature map $W_{f,l}$. $c_i$ and $b_i$ represent the Poincaré ball’s curvature and bias at layer $l$, respectively. $\ast$ and $\ast$ are the Möbius operations of addition and scalar multiplication, respectively.
\(a_{p,t}\) is the output of the scalar-attention \([32]\) over the outputs with attention weights \(\alpha\), and \(h_{p,t}\) is the layer’s output after non-linear hyperbolic activation. The initial layers aggregate the sparse neighborhoods into denser cells. As the adjacency tensors progress through the layers, the features are always of a lower resolution than the previous layer (aggregation over aggregation), and thus aggregation in the later layers results in graph-level metapath features, as depicted in Figure 5. Note that the computational complexity of calculating \(a_{p,t}\) in sparse graph convolutions is \(O(V^2(1-R))\) when compared to \(O(V^2)\) of dense graph convolutions\([4]\). This indicates a reduction in the total number of computations by a factor of \((1-R) \approx 10^{-4}\). Prior hyperbolic approaches could not utilize sparse convolutions because the hyperbolic operation could not be performed on splits of the adjacency tensor encodings \(A\) of all edge types \(k\), we aggregate the features of adjacency tensor with the semantic residual network connection sends node signals to the adjacency tensor and also passes information to the multi-step loss function. The balance between semantic residual network and hyperbolic graph convolution leads to robustness against noisy text or graphs (evaluated empirically in Section 4.6).

\[y_k = P_\theta(e_k|x) = P_\theta(z = 1|x)P_\theta(y = e_k|x)\]  \hspace{2cm} (11)

\[L(y_k, \hat{y}_k) = -\sum_{k=1}^{K} \hat{y}_k \log(y_k)\]  \hspace{2cm} (12)

where \(x\) and \(\theta\) are the input and model parameters, respectively. \(L\) is the cross entropy loss that needs to be minimized. Although we use this generalized link prediction as the task of interest in this paper, TESH-GCN can be applied to any task such as node/graph classification by replacing the loss with the appropriate loss.

### 3.4 Implementation Details

We implemented TESH-GCN using Pytorch \([33]\) on eight NVIDIA V100 GPUs with 16 GB VRAM. For gradient descent, we used Riemannian Adam \([34]\) with standard \(\beta\) values of 0.9 and 0.999 and an initial learning rate of 0.001. Number of dimensions \((D)\) and number of layers \((L)\) is empirically selected based on performance-memory trade-off. Figure 6 presents the memory-performance trade-off for different choices of parameters \(D\) and \(L\). We observe that the \(D = 8\) and \(L = 8\) provides the best performance for the memory required. Hence, we chose them for the final implementation of our model. For non-linearity, we used the hyperbolic activation function, given in Eq. (1). The sparsity in the model variables is handled using the torch-sparse library\([6]\). While this library and other similar ones handle the operational sparsity of the graphs, previous GNN-based approaches need to locally convert the sparse tensors to the corresponding dense format for their layer operations.

\[^4\]Practically, for a graph with \(10^5\) nodes and a sparsity of \(10^{-4}\), dense graph convolution requires 80GB of memory (assuming double precision) for one layer, whereas, sparse graph convolution only requires 1MB of memory for the same. This allows us to utilize the entire adjacency tensor, while previous approaches can only rely on the local neighborhood.

\[^5\]The transformation from hyperbolic space to tangent space with logarithmic map is required for attention-based aggregation as such formulation is not well-defined for the hyperbolic space.
In TESH-GCN, the conversion is not required because all operations in Sparse-HGCN are directly performed on the sparse tensor as it only considers the non-zero elements of the tensor. Each convolution operation moves up one-hop in the nodes’ neighborhood. Hence, the number of graph convolution layers should at least be the maximum shortest path between any two nodes in the graph. For a dataset, this is empirically calculated by sampling nodes from the graph and calculating the maximum shortest path between them. For the datasets in our experiments, we used 8 layers \((L = 8)\) to extract local neighborhoods in the early layers and metapath structures in the later layers. The main adjacency tensor can be split either over the number of semantic signals \((D)\) or the number of edge types \((K)\). We chose the latter because each adjacency tensor needed a separate GPU and it was more efficient and convenient to control the training process, given that the number of edge types is lesser than the number of semantic signals in our experiments. Algorithm 1 provides the pseudocode for training the model.

4 EXPERIMENTAL SETUP

In this section, we describe our experimental setup and investigate the following research questions (RQs):

1) RQ1: Does TESH-GCN perform better than the state-of-the-art approaches for the task of link prediction?
2) RQ2: What is the contribution of TESH-GCN’s individual components to the overall performance?
3) RQ3: How does TESH-GCN compare against previous approaches in time and space complexity?
4) RQ4: How robust is TESH-GCN against noise in the graph and its corresponding text?
5) RQ5: Can we comprehend the results of TESH-GCN?

4.1 Datasets Used

For the datasets, we select the following widely used publicly available network benchmark datasets where the nodes contain certain semantic information in the form of text attributes. Also, the choice of the datasets is driven by the diversity of their hyperbolicity to test performance on different levels of latent hierarchy (lower hyperbolicity implies more latent hierarchy).

1) Amazon \[^{[35]}\] is a heterogeneous e-commerce graph dataset that contains electronic products as nodes with title text connected by edges based on the purchase information. The edge types are also_buy (products bought together) and also_view (products viewed in the same user session).
TABLE 2: Dataset statistics including no. of nodes (V), edges (E), edge types (K), hyperbolicity (δ), and sparsity ratio (R).

| Dataset     | V       | E       | K       | δ       | R (%)  |
|-------------|---------|---------|---------|---------|--------|
| Amazon      | 368,871 | 6,471,233 | 2       | 2       | 99.99  |
| DBLP        | 37,791  | 170,794 | 3       | 4       | 99.99  |
| Twitter     | 81,306  | 1,768,149 | 1       | 1       | 99.97  |
| Cora        | 2,708   | 5,429   | 1       | 11      | 99.92  |
| MovieLens   | 10,010  | 1,122,457 | 3       | 2       | 99.00  |

4.4 RQ2: Ablation Study

In this section, we study the importance of different components and their contribution to the overall performance of our model. The different components we analyze in our ablation study are: (i) the semantic text signal, (ii) the hyperbolic transformations, (iii) the residual network, and (iv) the multi-step loss. The ablation study is conducted on the same datasets by calculating the evaluation metrics after freezing the parameters of the component of interest in the model. The results of the study are presented in Table 9.

The results show that the text signal contributes to a significant margin on different evaluation metrics. Additionally, we notice that the performance improvement of hyperbolic models (HGCN and TESH-GCN) is more on datasets with lower hyperbolicity (higher latent hierarchy). This shows that hyperbolic space is better at extracting hierarchical features from the graph structures. Furthermore, we see that the performance decreases a little without the residual network. However, it does not justify the additional parameters but it adds robustness against noisy graph and text (evaluation in Section 4.4), so we use this variant in our final model. Another point of note is that text-based frameworks are better than graph approaches in datasets with good semantic information such as Amazon, whereas, graph-based approaches are better on well-connected graphs such as Cora. However, TESH-GCN is able to maintain good performance in both the scenarios, demonstrating its ability to capture both semantic and structural information from the dataset.

4.3 RQ1: Performance on Link Prediction

To analyze the performance of TESH-GCN, we compare it against the state-of-the-art baselines using standard graph datasets on the task of link prediction. We input the node-pairs \((v_i, v_j)\) with the corresponding text sequence \((s_i, s_j)\) to the model and predict the probability that an edge type \(e_k\) connects them as \(y_{ik} = P_{e_k}(v_i, v_j, s_i, s_j)\). We evaluate our model using 5-fold cross validation splits on the following standard performance metrics: Accuracy (ACC), Area under ROC curve (AUC), Precision (P), and F-score (F1). For our experimentation, we perform 5-fold cross validation with a training, validation and test split of 8:1:1 on the edges of the datasets. Table 4 provides the number of samples and sparsity of each split in the dataset. The results on the test set are presented in Table 5.

From the experimental results, we observe that TESH-GCN is able to outperform the previous approaches by a significant margin on different evaluation metrics. Additionally, we notice that the performance improvement of hyperbolic models (HGCN and TESH-GCN) is more on datasets with lower hyperbolicity (higher latent hierarchy). This shows that hyperbolic space is better at extracting hierarchical features from the graph structures. Furthermore, we see that the performance decreases a little without the residual network. However, it does not justify the additional parameters but it adds robustness against noisy graph and text (evaluation in Section 4.4), so we use this variant in our final model. Another point of note is that text-based frameworks are better than graph approaches in datasets with good semantic information such as Amazon, whereas, graph-based approaches are better on well-connected graphs such as Cora. However, TESH-GCN is able to maintain good performance in both the scenarios, demonstrating its ability to capture both semantic and structural information from the dataset.
4.5 RQ3: Complexity Analysis

One of the major contributions of TESH-GCN is its ability to efficiently handle sparse adjacency tensors in its graph convolution operations. To compare its performance to previous graph- and text-based methods, we analyze the space and time complexity of our models and the baselines. The space complexity is studied through the number of parameters and time complexity is reported using the training and inference times of the models. We compare the space and time complexity of our models using large graphs of different sparsity ratios ($R$) (by varying the number of edges/links on a graph with $10^4$ nodes). The different sparsity ratios considered in the evaluation are $1 - 10^{-r}$ for $r \in [0, 5]$. Figure 8 and Table 5 shows the comparison of different GCN based models’ training time on varying sparsity ratios and inference times on different datasets, respectively. Table 6 presents the number of parameters and space complexity of the different baselines in comparison to TESH-GCN. From

| Datasets | Models | Amazon | DBLP | Twitter | Cora | MovieLens |
|----------|--------|--------|------|---------|------|-----------|
|          | N      | R(%)   | N    | R(%)    | N    | R(%)      |
|          |        |        |      |         |      |           |
| Amazon   | 5,176,986 | 99.99 | 647,123 | 99.99 | 647,124 | 99.99  |
| DBLP     | 1,36,635   | 99.99 | 17,079  | 99.99 | 17,080  | 99.99  |
| Twitter  | 1,414,519  | 99.97 | 176,815 | 99.99 | 176,815 | 99.99  |
| Cora     | 4,343     | 99.94 | 543    | 99.99 | 543    | 99.99  |
| MovieLens| 897,966  | 99.10 | 112,245 | 99.88 | 112,246 | 99.88  |

Table 5: Inference times (in milliseconds) of our model and various GCN-based baseline methods on different datasets.

| Models     | Amazon | DBLP | Twitter | Cora | MovieLens |
|------------|--------|------|---------|------|-----------|
| GCN        | 719    | 723  | 728     | 735  | 744       |
| HGCN       | 745    | 757  | 758     | 763  | 774       |
| TextGNN    | 1350   | 1368 | 1375    | 1394 | 1395      |
| Graphormer | 1423   | 1430 | 1441    | 1442 | 1458      |
| TESH-GCN   | 787    | 794  | 803     | 817  | 822       |

Table 6: The number of non-trainable (in millions) and trainable (in thousands) parameters of all the comparison methods. We also report the space complexity in terms of the number of nodes ($V$), maximum text length ($S$), and sparsity measure $N = \frac{1}{1-R} \approx 10^4$. 

| Model     | Non-Train (M) | Train (K) | Complexity |
|-----------|---------------|-----------|------------|
| C-DSSM    | 0             | 38        | $O(S)$     |
| BERT      | 110           | 1600      | $O(S^2)$   |
| XLNet     | 110           | 1600      | $O(S^2)$   |
| GraphSage | 0             | 4800      | $O(V^2)$   |
| GCN       | 0             | 4800      | $O(V^2)$   |
| HGCN      | 0             | 9600      | $O(2V^2)$  |
| TextGNN   | 110           | 6400      | $O(SV^2)$  |
| TextGNN   | 110           | 6400      | $O(SV^2)$  |
| Graphormer| 100           | 7600      | $O(SV^2)$  |
| TESH-GCN  | 110           | 78        | $O\left(\frac{2SV^2}{N}\right)$ |
the time complexity analysis, we notice that TESH-GCN consistently takes much less training time than the other GCN-based and hybrid approaches in high sparsity graphs. This shows that the current GCN-based approaches do not handle the sparsity of the adjacency tensor. However, the overhead of specialized graph convolution layer in TESH-GCN leads to a poor time complexity for cases with high graph density ($R < 0.9$). From the comparison of inference times, given in Table 5, we notice that TESH-GCN’s inference time is comparable to the graph-based baselines and significantly lesser than hybrid baselines. Figure 9 provides the effect of sparsity on the inference time of our model and the baselines. We note that TESH-GCN is able to outperform other hybrid graph-text baselines and needs similar inference time as the baselines that only consider the local neighborhood of its nodes. TESH-GCN is faster for high sparsity graphs but the overhead of specialized graph convolutions takes more time than other baselines on high density graphs.

The space complexity analysis clearly shows that TESH-GCN uses much lesser number of model parameters than baselines with comparable performance. Also, the complexity shows the dependence of text-based approaches on only the textual sequence length, whereas, the graph based are dependent on the number of nodes. However, TESH-GCN is able to reduce the space complexity by a factor of the sparsity ratio and only consider informative non-zero features from the adjacency tensors, leading to a decrease in the number of trainable parameters.

### 4.6 RQ4: Model Robustness

To test the robustness of our model, we introduce varying levels of noise into the Amazon graph by (i) node drop: dropping n% percentage of nodes, (ii) text replacement: replacing n% percentage of the text, and (iii) hybrid noise: dropping n% of nodes and replacing n% of text. We compare the performance of our model and the baselines across different values of $n = 10, 20, 30, 40$, and 50. The results for the robustness evaluation are given in Figure 7.

First, we highlight the main observations, that node drop and text replacement only affects graph-based and text-based approaches, respectively (and does not affect them vice versa). In the case of hybrid baselines, we still note a decrease in performance for both the noise variants. This implies that the text and graph features in the baselines do not complement each other. In the case of TESH-GCN, we note that both the noise variants do not cause any significant performance loss. This shows that the complementary nature of the semantic residual network and hyperbolic graph convolution network leads to an increased robustness against noise in either the text or graph. In the third scenario with hybrid noise, we see a reduction of $\approx 25\%$
performance in text-based and graph-based baselines and \( \approx 50\% \) in hybrid baseline with a 50\% noise. However, we notice that, although TESH-GCN is a hybrid model, we only observe a 25\% performance loss with 50\% noise, implying the effectiveness of text-graph correspondence in the scenario of hybrid noise as well. Thus, we conclude that TESH-GCN is robust against noise in either graph or text, but vulnerable, albeit less than other hybrid baselines, to a joint attack on both the graph and text.

### 4.7 RQ5: Model Explainability

Model comprehension is a critical part of our architecture as it helps us form a better understanding of the results and explain the model’s final output. To understand TESH-GCN’s link prediction, we look at the different metapaths that connect the input nodes as well as the text in the metapaths’ nodes that receive the most attention \( (\alpha_k) \). For this, we follow the graph convolution and attention pooling operations through the layers in the network and extract the most critical metapaths chosen by the model to arrive at the prediction. The methodology for extracting the metapaths with their corresponding weightage in the final link prediction is presented in Algorithm 2. Figure 10 depicts some metapaths extracted from the Amazon dataset. In Figures 10a and 10b, we note that TESH-GCN aggregates information from multiple long (4-hop) metapaths between the input nodes for prediction. Additionally, we see tokens in the node’s text being emphasized (having higher attention weight) based on the edges through which they propagate their semantic information, e.g., in Figure 10b, we observe that key tokens: Pirates of the Caribbean and Necklace propagate the semantic information to match with additional relevant tokens such as Cursed Aztec, Medallion, Pendant and coin to establish the edge also_buy between the input nodes. Thus, we observe the role of different metapaths as

#### Algorithm 2: Explaining results through Metapaths

**Input:** Input \((v_i, s_i, v_j, s_j)\), Predictor \(P_0\);

**Output:** Metaph set \(M\), Class prediction \(y_k\);

1. Initialize metapath set \(M = \emptyset\);
2. \(t_i \leftarrow LM(s_i), t_j \leftarrow LM(s_j)\);
3. for \(e_k \in E\) do
4. Initialize metapath for \(e_k\), \(M_k = \emptyset\);
5. # stack D-repetitions of adjacency matrix
6. \(A_k = stack(E_k, D)\);
7. \(A_k[d, i, :] = t_i[d], A_k[d, j, :] = t_j[d]\)
8. \(x_0 = A_k\)
9. # Run through \(L\) graph convolution layers
10. for \(l : 1 \rightarrow L\) do
11. \(h_{p,t} = W_l \otimes_{c_l} x_{p,t-1} \otimes_{c_l} b_l \quad \forall x_{p,t-1} \neq 0\)
12. \(a_{p,t} = exp^{c_l} \left( \sum_{p} \alpha_p \log^2(a_{p,t}) \right)\)
13. \(h_{p,t} = \sigma_{c_l}(a_{p,t})\)
14. \(M_k = M_k \cup \arg \max_{h_{p,t}} h_{p,t}\)
15. end
16. \(h_{k,L} = h_{p,t}\)
17. end
18. # Attention over outputs
19. \(h_{k,L} = \sum_{k} \alpha_k h_{k,L}\)
20. # Extracted metapath \(M_k\) with attention weight \(\alpha_k\)
21. \(M = M \cup (M_k, \alpha_k)\)
22. \(h_L = h_{1,L} \odot h_{2,L} \odot ... \odot h_{k,L}\)
23. \(out(A) = h_L \odot t_i \odot t_j\)
24. # Predicted class probability
25. \(y_k = \text{softmax}(\text{dense}(out(A)))\)
26. return \(M, y_k\)
well as semantic information in the message propagation towards the downstream task of link prediction.

5 Conclusion

In this paper, we introduced Text Enriched Sparse Hyperbolic Graph Convolution Network (TESH-GCN), a hybrid graph and text based model for link prediction. TESH-GCN utilizes semantic signals from nodes to aggregate intra-node and inter-node information from the sparse adjacency tensor using a reformulated hyperbolic graph convolution layer. We show the effectiveness of our model against the state-of-the-art baselines on diverse datasets for the task of link prediction and evaluate the contribution of its different components to the overall performance. Additionally, we demonstrate the optimized memory and faster processing time of our model through space and time complexity analysis, respectively. Furthermore, we also show TESH-GCN’s robustness against noisy graphs and text and provide a mechanism for explaining the results produced by the model.

References

[1] N. Choudhary, N. Rao, S. Katariya, K. Subbian, and C. K. Reddy, “ANTHEM: Attentive hyperbolic entity model for product search,” in Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, ser. WSDM ’22. New York, NY, USA: Association for Computing Machinery, 2022, p. 161–171. [Online]. Available: https://doi.org/10.1145/3488650.3488945

[2] J. Leskovec and J. Mcauley, “Learning to discover social circles in ego networks,” in Advances in Neural Information Processing Systems, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/7af614fd6cc3249f41680b9896beedeb-Paper.pdf

[3] T. Cohen, “Infectious Diseases of Humans: Dynamics and Control,” JAMA, vol. 268, no. 23, pp. 3381–3381, 12 1992. [Online]. Available: https://doi.org/10.1001/jama.1992.03490230111047

[4] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, “Collective classification in network data,” Al Magazine, vol. 29, no. 3, pp. 93–106, 2008.

[5] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD ’14. New York, NY, USA: ACM, 2014, pp. 701–710. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623732

[6] A. Grover and J. Leskovec, “node2vec: Scalable feature learning for networks,” in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.

[7] W. L. Hamilton, R. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in Proceedings of the 31st International Conference on Neural Information Processing Systems, ser. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 1025–1035.

[8] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ay9yG

[9] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph Attention Networks,” International Conference on Learning Representations, 2018, accepted as poster. [Online]. Available: https://openreview.net/forum?id=rJXmpik2Z

[10] L. Wu, H.-F. Yu, N. Rao, J. Sharpnack, and C.-J. Hsieh, “Graph dna: Deep neighborhood aware graph encoding for collaborative filtering,” in International Conference on Artificial Intelligence and Statistics. PMLR, 2020, pp. 776–787.

[11] O. Ganea, G. Bécigneul, and T. Hofmann, “Hyperbolic neural networks,” in Advances in neural information processing systems, 2018, pp. 5345–5355.

[12] I. Chami, Z. Ying, C. Rê, and J. Leskovec, “Hyperbolic graph convolutional neural networks,” in Advances in Neural Information Processing Systems, 2019, pp. 4869–4880.

[13] J. Zhu, Y. Cui, Y. Liu, H. Sun, X. Li, M. Pelger, T. Yang, L. Zhang, R. Zhang, and H. Zhao, “Textgcn: Improving text encoder via graph neural network in sponsored search,” in Proceedings of the Web Conference 2021, ser. WWW ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2848–2857. [Online]. Available: https://doi.org/10.1145/3442381.3449842

[14] L. Yao, C. Mao, and Y. Luo, “Graph convolutional networks for text classification,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 7370–7377, Jul. 2019. [Online]. Available: https://openreview.net/forum?id=rJXHskRfQ

[15] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu, “Do transformers really perform badly for graph representation?” in Advances in Neural Information Processing Systems, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. [Online]. Available: https://openreview.net/forum?id=rJXHskRfQ

[16] X. Fu, J. Zhang, Z. Meng, and I. King, MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding. New York, NY, USA: Association for Computing Machinery, 2020, p. 2331–2341. [Online]. Available: https://doi.org/10.1145/3388097}

2021, p. 776–787.

2020, pp. 1536–1547. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.139

2019, pp. 4869–4880.

2008.

2015, pp. 1067–1077.

2015, pp. 1067–1077.

2016, pp. 1225–1234.

2021, pp. 7370–7377.

2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

2016, pp. 855–864.

2016, pp. 1025–1035.

2017, Conference Track Proceedings. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ay9yG

2017.

2016, accepted as poster. [Online]. Available: https://openreview.net/forum?id=rJXmpik2Z

2020, pp. 776–787.

2020, pp. 5345–5355.
over knowledge graphs,” in Proceedings of the Web Conference 2021, ser. WWW ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1373–1384. [Online]. Available: https://doi.org/10.1145/3442381.3449974

[29] K. Shinmura, Y. Mukata, and T. Harada, “Hyperbolic neural networks++,” in International Conference on Learning Representations, 2021. [Online]. Available: https://openreview.net/forum?id=Ec58b0UwbA

[30] M. Gromov, Hyperbolic Groups. New York, NY: Springer New York, 1987, pp. 75–263. [Online]. Available: https://doi.org/10.1007/978-1-4613-9586-7_3

[31] K. Song, X. Ian, T. Qin, J.-Liu, and T.-Y. Liu, “Mpnnet: Masked and permuted pre-training for language understanding,” arXiv preprint arXiv:2004.09297, 2020.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in neural information processing systems, 2017, pp. 5998–6008.

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., “Pytorch: An imperative style, high-performance deep learning library,” Advances in neural information processing systems, vol. 32, pp. 8026–8037, 2019.

[34] G. Becigneul and O.-E. Ganea, “Riemannian adaptive optimization methods,” in International Conference on Learning Representations, 2019. [Online]. Available: https://openreview.net/forum?id=r1eiq09K7

[35] R. He and J. J. McAuley, “Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering,” in Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 – 15, 2016, J. Bourdeau, J. Hendler, R. Nkambou, I. Horrocks, and B. Y. Zhao, Eds. ACM, 2016, pp. 507–517. [Online]. Available: https://doi.org/10.1145/2872427.2883037

[36] M. Ji, Y. Sun, M. Danilevsky, J. Han, and J. Gao, “Graph regularized transductive classification on heterogeneous information networks,” in Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, 2011, pp. 570–586.

[37] R. A. Rossi and N. K. Ahmed, “The network data repository with interactive graph analytics and visualization,” in AAAI, 2015. [Online]. Available: https://networkrepository.com

[38] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” ACM Trans. Interact. Intell. Syst., vol. 5, no. 4, Dec. 2015. [Online]. Available: https://doi.org/10.1145/2827877

[39] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnili, “Learning semantic representations using convolutional neural networks for web search,” in Proceedings of the 23rd International Conference on World Wide Web, ser. WWW ’14 Companion. New York, NY, USA: Association for Computing Machinery, 2014, p. 373–374. [Online]. Available: https://doi.org/10.1145/2567948.2577348

[40] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, “Learning deep structured semantic models for web search using clickthrough data,” in Proceedings of the 22nd ACM international conference on Information and Knowledge Management, 2013, pp. 2333–2338.

[41] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” in Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alchê-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/6e055db75840e6673e6e3ee67cc0f-Paper.pdf

[42] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

Nurendra Choudhary is a Ph.D. student in the department of Computer Science at Virginia Tech. His research, under advisor Dr. Chandan Reddy, is focused on representation learning in the fields of graph analysis and product search. He has published several peer-reviewed papers in top-tier conferences and journals including ACM TIST, WWW, NeurIPS, WSDM, KDD and COLING. He has received his M.S. in Computational Linguistics from International Institute of Information Technology, during which he received the Best Paper Award at CIICLING, 2018.

Nikhil Rao is a senior scientist at Amazon where he works on large scale graph modeling and algorithms to improve Amazon Search. Prior to joining Amazon, he was a researcher at Technicolor AI Labs in Palo Alto. Nikhil’s research interests and expertise span large scale optimization, data modeling and mining, and developing algorithms that take advantage of structure present in the data. Nikhil has published several papers in top-tier conferences and journals. He is the recipient of the ICES Post Doctoral Fellowship award from UT Austin, and the IEEE Best Student Paper award. He holds a Ph.D. in Electrical and Computer Engineering from UW Madison.

Karthik Subbian is a principal scientist at Amazon with more than 17 years of industry experience. He leads a team of scientists and engineers to improve search quality and trust. He was a research scientist and lead at Facebook, before coming to Amazon, where he had led a team of scientists and engineers to explore information propagation and user modeling problems using the social network structure and its interactions. Earlier to that, he was working at IBM T.J. Watson research center in the Business Analytics and Mathematical Sciences division. His areas of expertise include machine learning, information retrieval, and large-scale network analysis. More specifically, semi-supervised and supervised learning in networks, personalization and recommendation, information diffusion, and representation learning. He holds a masters degree from the Indian Institute of Science (IISc) and a Ph.D. from the University of Minnesota, both in computer science. Karthik has won numerous prestigious awards, including the IBM Ph.D. fellowship, best paper award at SIAM Data Mining (SDM) conference 2013, and INFORMS Edelman laureate award 2013.

Chandan K. Reddy is a Professor in the Department of Computer Science at Virginia Tech. He received his Ph.D. from Cornell University and M.S. from Michigan State University. His primary research interests are machine learning and data analytics with applications to various real-world domains including healthcare, transportation, social networks, and e-commerce. He has published over 150 peer-reviewed articles in leading conferences and journals. He received several awards for his research work including the Best Application Paper Award at ACM SIGKDD conference in 2010, Best Poster Award at IEEE VAST conference in 2014, Best Student Paper Award at IEEE ICDM conference in 2016, and was a finalist of the INFORMS Franz Edelman Award Competition in 2011. He is serving on the editorial boards of ACM TKDD, IEEE Big Data, and ACM TIST journals. He is a senior member of the IEEE and distinguished member of the ACM.