Hierarchical Multi-label Text Classification with Horizontal and Vertical Category Correlations

Linli Xu[1], **Sijie Teng**[1], **Ruoyu Zhao**[1], **Junliang Guo**[1], **Chi Xiao**[1], **Deqiang Jiang**[2], **Bo Ren**[2]
[1]Anhui Province Key Laboratory of Big Data Analysis and Application,
School of Computer Science and Technology, University of Science and Technology of China
[2]Tencent YouTu Lab
linlixu@ustc.edu.cn, {yunmo, zry1997, guojunll, xiaochi}@mail.ustc.edu.cn
{dqiangjiang, timren}@tencent.com

Abstract

Hierarchical multi-label text classification (HMTC) deals with the challenging task where an instance can be assigned to multiple hierarchically structured categories at the same time. The majority of prior studies either focus on reducing the HMTC task into a flat multi-label problem ignoring the vertical category correlations or exploiting the dependencies across different hierarchical levels without considering the horizontal correlations among categories at the same level, which inevitably leads to fundamental information loss. In this paper, we propose a novel HMTC framework that considers both vertical and horizontal category correlations. Specifically, we first design a loosely coupled graph convolutional neural network as the representation extractor to obtain representations for words, documents, and, more importantly, level-wise representations for categories, which are not considered in previous works. Then, the learned category representations are adopted to capture the vertical dependencies among levels of category hierarchy and model the horizontal correlations. Finally, based on the document embeddings and category embeddings, we design a hybrid algorithm to predict the categories of the entire hierarchical structure. Extensive experiments conducted on real-world HMTC datasets validate the effectiveness of the proposed framework with significant improvements over the baselines.

1 Introduction

As a fundamental problem in natural language processing (NLP), text classification is the task of assigning a given document to one or multiple categories according to its textual content. In practice, many documents are tagged with multiple categories that can be organized in a tree or a Directed Acyclic Graph (DAG) (Wehrmann et al., 2018), which poses a more challenging task. These categories can be organized into different levels of the hierarchical structure, and the task of assigning multiple hierarchically structured categories to documents is known as hierarchical multi-label text classification (HMTC).

For the hierarchical multi-label classification problem, it is essential to model the dependencies among categories in the hierarchical structure. The vertical correlations capture the dependencies of categories at different levels, and the horizontal correlations reflect the relationships at the same level. Straightforwardly, the HMTC problem can be reduced to a flat multi-label problem (Fall et al., 2003), which simply ignores the vertical category correlations. To address that, attempts have been made to exploit the hierarchical dependencies to improve the classification performance. Among them, (Costa et al., 2007) adopts a top-down way to generate a hierarchical structure of decision-tree-based classifiers to predict categories at the corresponding hierarchical level. (Wehrmann et al., 2018) proposes a unified framework that combines the local outputs of each category hierarchical level and the global output of the entire network. (Huang et al., 2019a) designs a hierarchical attention-based memory unit to model the dependencies among different levels in a top-down fashion. However, horizontal correlations between categories at the same hierarchical level are usually ignored, resulting in a lack of information transfer of label characteristics at the same level.

In principle, the characteristics of categories are encoded in both the horizontal correlations among categories at the same level and the vertical dependencies between categories at different levels in the hierarchical organization. Precisely, the vertical correlation measures the top-down relevance of a text node's tags, while the horizontal correlation is responsible for enhancing the information transfer of labels within the same hierarchical level. An example is shown in Figure 1. The red area reveals the vertical dependency of categories, which

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2459–2468
November 7–11, 2021.

has been widely exploited in HMTC tasks. In the meantime, the blue area represents the horizontal correlation of categories, which is latent but essential. For instance, when we are confident that the document is tagged with *hierarchical classification*, it is more likely that the document is associated with *multi-label classification* rather than *single label classification*. To fully leverage the hierarchical structure of categories, this paper integrates the horizontal and vertical correlations extracted by a novel graph convolutional neural network (GCN).

The graph convolutional neural network (Kipf and Welling, 2016) has been successfully applied to text classification due to its capabilities in handling complex structures. A GCN-based text classification model generally works by treating the training documents, test documents, and words as nodes to construct a single huge graph, which incurs the following issues in hierarchical text classification. Firstly, during network training, the category information which provides valuable semantic information is missing. Secondly, organizing all nodes in one graph expands the size of the graph, which not only leads to the increased difficulty of training, but also brings information confusion between nodes.

To address the problems discussed above, in this paper, we propose an integrated framework named *Horizontally and Vertically Hierarchical Multi-label Text Classification* (HVHMC) to exploit the vertical and horizontal category correlations simultaneously by introducing a newly designed *Loosely Coupled Graph Convolutional Network* (LCGCN) as the representation component. Specifically, we include the category nodes together with word nodes and document nodes in LCGCN by constructing two separate graphs: the core graph containing words and categories, and the document-word graph. Then, based on the category representations learned based on the loosely coupled graph neural network, a level-wise category correlation matrix is calculated, which captures the horizontal dependencies among categories and facilitates the semantic transfer among categories at each hierarchical level. Finally, a hybrid algorithm is proposed to further incorporate vertical dependencies of categories by integrating the label information on the hierarchical path. The predictions of each hierarchical level and the overall hierarchical structure will be combined as the final results.

The contributions of our paper are as follows:

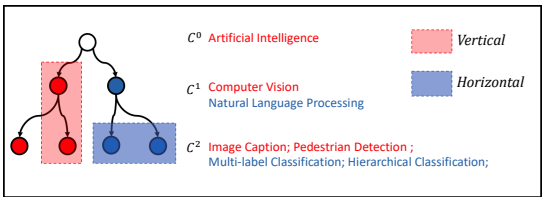

Figure 1: The horizontal and vertical correlations in the category hierarchical structure. The red area reveals the vertical dependency of categories. Meanwhile, the blue area represents the horizontal correlation of categories.

- We propose the *Loosely Coupled Graph Convolutional Neural Network* (LCGCN) as the representation component. By dealing with the core graph and the document-word graph separately, this feature extraction approach can greatly enhance the quality of semantic representations and reduce the training expenses while maintaining the performance;

- We propose a unified framework HVHMC for the hierarchical multi-label text classification problem that integrates vertical and horizontal category dependencies simultaneously, with significant improvements over the baselines on three real datasets.

2 Related Works

In this section, we mainly review the related studies on hierarchical multi-label classification and graph convolutional neural networks.

2.1 Hierarchical Multi-label Classification

To leverage the category hierarchical structure in the hierarchical multi-label classification problem, (Cai and Hofmann, 2004) considers the parent-child dependency of categories by organizing the discriminant functions in a way that mirrors the category hierarchical structure. (Banerjee et al., 2019) introduces a Hierarchical Transfer Learning approach (HTrans), where classifiers at lower levels in the hierarchy are initialized using parameters of the parent classifier and fine-tuned on the child category classification tasks. (Wehrmann et al., 2018) proposes a hybrid framework, Hierarchical Multi-label Classification Networks (HMCN), which can simultaneously take both the local category correlations and global information from the entire category hierarchical structure into account. Based on HMCN, (Huang et al., 2019b) further

proposes a level-wise attention-based recurrent network (HARNN) to model the category dependencies among different levels. In (Zhou et al., 2020), the hierarchy is formulated as a directed graph, and hierarchy-aware structure encoders are introduced to model label dependencies. (Shen et al., 2021) conducts HMTC based on only class surface names as supervision signals, and generalizes the classifier via multi-label self-training. However, the approaches mentioned above do not consider horizontal and vertical correlations jointly.

2.2 Graph Convolutional Neural Networks

The Graph Convolutional Network (GCN) has attracted extensive attention recently (Zhang et al., 2018; Niepert et al., 2016; Scarselli et al., 2008) for its advantages of capturing non-consecutive and long-distance information. Various attempts have been made to apply GCNs to text classification. Among them, (Yao et al., 2019) introduces the Text-GCN model by building a single huge graph for the whole text corpus based on word co-occurrence and document word associations. Then the graph is fed into a two-layer GCN model to obtain the representations of both words and documents under the supervision of tagged instances. (Xin et al., 2021) considers heterogeneous label information which is ignored in Text-GCN, and incorporates the label information while building the graph by adding text-label-text paths. To alleviate the high memory consumption problem of Text-GCN, (Huang et al., 2019a) proposes constructing a text level graph for every given document. In comparison, (Peng et al., 2018) focuses on converting arbitrary graphs to a very regular one to be processed by a local convolution operator (Niepert et al., 2016). In this paper, a novel graph neural network is proposed to decouple the core graph and the document-word graph, which helps to improve the quality of semantic representations of graph nodes while reducing computational costs.

3 METHODOLOGY

This section introduces the proposed framework of Horizontally and Vertically Hierarchical Multi-label Text Classification (HVHMC) in details.

Problem Definition In the HMTC problem, there are a set of documents, each document contains the text description and its expected categories, which are organized in a hierarchical structure. Before defining the HMTC problem, we first give a detailed description of the hierarchical structure and documents.

Given the possible categories in H hierarchical levels $\mathbb{C} = (C^1, C^2, \cdots, C^H)$, where $C^h = \{c_1, c_2, \cdots\} \in \{0,1\}^{|C^h|}$ is the set of possible categories at the h-th hierarchical level and $T = \sum_{h=1}^{H} |C^h|$ is the total number of categories. A set of M documents with hierarchical categories can be denoted as $\mathcal{X} = \{(D_1, L_1), (D_2, L_2), \cdots, (D_M, L_M)\}$, where $D_i = \{w_{i1}, w_{i2}, \cdots, w_{iN}\}$ represents a sequence of N words from the word set $\mathbb{W} = \{w_1, w_2, \cdots, w_W\}$ with vocabulary size W. And $L_i = \{l_1, l_2, \cdots, l_H\}$ is the set of hierarchical categories assigned to the document D_i, where $l_h \subset C^h$. Given a set of documents and the corresponding set of hierarchical categories, the goal of the HMTC problem is to integrate the document texts $\mathbb{D} = \{w_1, w_2, \cdots, w_N\}$ and the corresponding set of hierarchical categories $\mathbb{C}$ to learn a classification model, which can be used to predict the hierarchical categories for documents.

3.1 Loosely Coupled Graph Convolutional Neural Networks

We consider the graph convolutional neural network for feature representation due to its advantages of capturing non-consecutive and long-distance information.

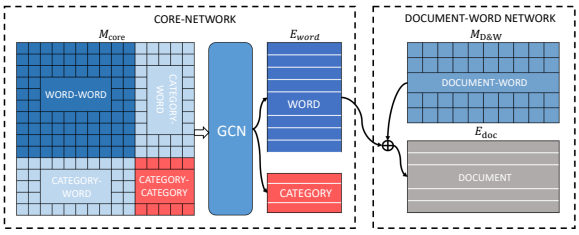

Figure 2: Overview of the loosely coupled graph convolutional neural network (LCGCN).

A natural idea of constructing a graph is to build a large graph containing all the information based on the affiliation of document nodes and word nodes, the correspondence between document nodes and category nodes, as well as the correspondence between word nodes and category nodes.

However, this tightly coupled way of constructing a large graph has some disadvantages. First, there are too many types of nodes in the graph, and the scale of the graph is huge, which is not conducive to representation learning of nodes. In ad-

dition, different kinds of correlations will increase the number of edges on the graph, which is more likely to result in the over-smoothing problem (Li et al., 2021).

Therefore, as Figure 2 shows, we propose a loosely coupled graph convolutional neural network (LCGCN), which consists of a separate core graph with words and categories, and a document-word graph with documents and words. The core graph is used to extract the word and category embeddings and the document-word graph captures the relationships between documents and words.

For the core graph M_{Core}, the edges can be divided into three types: word-word, category-word, and category-category. The weights between words in M_{Core} are calculated by the point-wise mutual information (PMI), a common measure for word associations. The weight of category i and word j is calculated as

$$M_{ij} = p(j|i) \cdot \log \frac{1}{p(j)}, \tag{1}$$

where $p(j|i)$ represents the frequency of word j in documents related to category i and $p(j)$ is the frequency of word j in the whole corpus. The higher the value, the more relevant the word is to the category. The weight between categories is calculated based on the co-occurrence of categories in the training data.

The document-word graph is constructed according to the dependency between document nodes and word nodes. The weight between a document node and a word node is calculated as the term frequency-inverse document frequency (TF-IDF).

After the core graph and document-word graph are constructed, the process of representation learning is divided into two consecutive steps. Firstly, we apply a one-layer GCN to the core graph M_{Core} and obtain word embeddings as well as category embeddings of the k-th layer:

$$H^k = [E^k_{\text{word}}; E^k_{\text{cat}}] = \sigma(\tilde{M}_{\text{Core}} H^{(k-1)} W^{(k-1)}), \tag{2}$$

where $E^k_{\text{word}} \in \mathbb{R}^{W \times d}$ and $E^k_{\text{cat}} \in \mathbb{R}^{T \times d}$ correspond to the updated representations of words and categories respectively, and d is the dimension of embeddings. $\tilde{M}_{\text{Core}} = D^{-\frac{1}{2}} M_{\text{Core}} D^{-\frac{1}{2}}$ is the normalized adjacency matrix and $W^{(k-1)} \in \mathbb{R}^{d \times d}$ is the parameter matrix. σ is a non-linear activation function. $H^{(k-1)}$ is the output of the previous layer and H^0 is the initial embedding matrix, which is initialized with the word and category vectors obtained by the pre-trained GloVe model (Pennington et al., 2014).

Then, the document representations are calculated by multiplying the document-word matrix $M_{\text{D\&W}}$ and the learned word embeddings E_{word} of the final layer of LCGCN:

$$E_{\text{doc}} = M_{\text{D\&W}} \cdot E_{\text{word}}, \tag{3}$$

where $E_{\text{doc}} \in \mathbb{R}^{M \times d}$ is the document representation matrix. By stacking multiple LCGCN layers, we can incorporate higher order neighborhood information to obtain high-quality representations.

It is worth mentioning that we do not use another GCN to train the document-word graph due to the following reason. In the bipartite graph of documents and words, document nodes are connected by shared word nodes. Therefore, in a GCN with more than two layers, a document node will absorb information from other document nodes, resulting in information confusion. So we use matrix multiplication to linearly aggregate word embeddings to generate document node representations instead, which is equivalent to a one-layer GCN.

By decoupling the tightly coupled GCN network into two graphs, we obtain a loosely coupled graph convolutional network. The benefits of splitting the three types of nodes into two separate graphs in the loosely coupled way are as follows:

- Firstly, the purpose of introducing the document nodes and the category nodes into the graph neural network is to enhance the semantic quality of the representations as to the guiding labels. However, these two types of nodes have different guiding directions for the word nodes. Specifically, the document-word graph plays the role of language models as in neural machine translation (NMT) tasks, while the core graph learns the lexical differences of different categories. If the nodes of documents and categories are integrated in one heterogeneous graph, information confusion may degrade the training quality.

- Secondly, a significant problem of GCN training is the excessive smoothness (Li et al., 2021), or the high similarity of nodes' representations caused by excessive dissemination of information through the edges in the graph. In general, tightly coupled graphs involve more information propagation paths

than loosely coupled graphs. For example, in a tightly coupled graph, word nodes can transfer information across different types of label guiding nodes (e.g., through word-category-document-word paths), while the information between two word nodes in a loosely coupled way can only be transmitted through the same type of label guiding nodes (e.g., through word-category-word paths or word-document-word paths). Therefore, the training of a loosely coupled graph is less prone to over-smoothing.

3.2 Category Correlations

Based on the word and category representations obtained by LCGCN, we further introduce our methods to extract the hierarchical dependencies between categories. As discussed earlier, most previous works on hierarchical multi-label classification only focus on exploiting vertical category correlations. In our framework, we also consider horizontal correlations among categories in addition to vertical correlations.

In this subsection, we start with a discussion of the horizontal and vertical correlations in the hierarchical multi-label text classification task and then propose an integrated framework that utilizes the category representations produced above to model both types of category correlations.

3.2.1 Horizontal Category Correlations

At each category hierarchical level, a given document may be associated with multiple categories.

Here, we define a correlation matrix based on the learned category representations:

$$S^h = \text{softmax}(\lambda_s^h \frac{E_{\text{cat}}^h}{\sqrt{\|E_{\text{cat}}^h\|^2}} \cdot \frac{{E_{\text{cat}}^h}^T}{\sqrt{\|{E_{\text{cat}}^h}^T\|^2}}), \tag{4}$$

where $S^h \in \mathbb{R}^{|C^h| \times |C^h|}$ is the correlation matrix at the h-th level and $\lambda_s^h \in \mathbb{R}$ is a regularization parameter. Then, the category label matrix Y^h at the h-th hierarchical level, which may be incomplete or noisy, can be augmented using the correlation matrix S^h:

$$\tilde{Y^h} = S^h \cdot Y^h \tag{5}$$

where $\tilde{Y^h}$ is the supplementary label matrix. The regularization parameter λ_s^h can be used to adjust the degree of augmentation. Specifically, λ_s^h of a large value intensifies the discrimination of different classes, while a small λ_s^h makes the correlation matrix S^h smooth, and the category matrix will be augmented accordingly. Consequently, by introducing the horizontal correlations, the distinction of categories' representations can be enhanced.

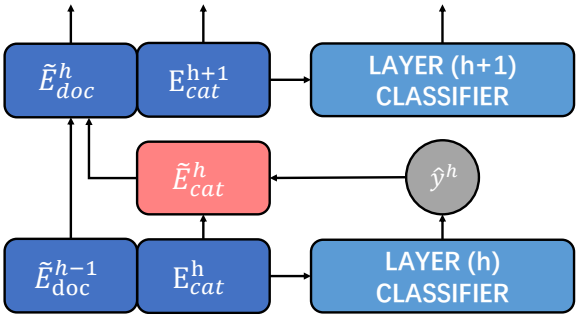

Figure 3: The vertical category correlations

3.2.2 Vertical Category Correlations

Different from traditional multi-label text classification tasks, the HMTC problem has explicit hierarchical dependencies among categories at different levels. For example, when it is determined that a document is associated with the category *natural language processing* at the current hierarchical level, the document is more likely to be tagged with *multi-label text classification* than *pedestrian detection* at the next hierarchical level. Thus, it is critical to consider the information from the previous hierarchical level.

Suppose $\hat{y}^h$ is the predicted output at the h-th hierarchical level, it is obvious that $\hat{y}^h$ contains the information regarding the probability distribution of the categories at that level. Thus, $\hat{y}^h$ can be utilized to integrate the representations of all the categories at the h-th hierarchical level:

$$\tilde{E}_{\text{cat}}^h = \text{softmax}(\hat{y^h}) \cdot E_{\text{cat}}^h, \tag{6}$$

where E_{cat}^h denotes the vector representations for categories at the h-th hierarchical level. For the h-th hierarchical classification task, it receives the concatenation of the document representations at the $(h-1)$-th level and $\tilde{E}_{\text{cat}}^h$ of the current hierarchical level as the input, and the output is the updated document embeddings at the h-th hierarchical level which integrate the document embeddings and category information:

$$\tilde{E}_{\text{doc}}^h = \text{MLP}[E_{\text{doc}}^{h-1}; \tilde{E}_{\text{cat}}^h] \tag{7}$$

where $\tilde{E}_{\text{doc}}^h \in \mathbb{R}^{M \times d}$, $\tilde{E}_{\text{doc}}^0 = E_{\text{doc}}$.

By considering the probability distribution of the categories at the previous layer, the model can cap-

ture the dependencies among categories at different levels.

3.3 Loss Function

We adopt a multiple output framework proposed in (Wehrmann et al., 2018) for prediction. The framework consists of a series of local classifiers per hierarchical level and a global classifier for the entire hierarchical structure. The local classification task at each hierarchical level is a multi-class classification task, where we use document embeddings and category embeddings at the corresponding hierarchical level. The local predictions at the hierarchical level h are calculated by:

$$P^h_{\text{loc}} = \sigma(W^h_{\text{loc}} \tilde{E}^h_{\text{doc}} + b^h_{\text{loc}}) \tag{8}$$

where $P^h_{\text{loc}} \in \mathbb{R}^{|C^h| \times d}$. $W^h_{\text{loc}} \in \mathbb{R}^{|C^h| \times M}$ denotes the classifier parameters and b^h_{loc} is the corresponding bias vector.

The global prediction of the entire hierarchical structure is a multi-label classification task. Here we aggregate the document embeddings at each hierarchical level to obtain the global embeddings, which is:

$$E_{\text{glob}} = \text{Aggregator}(\tilde{E}^1_{\text{doc}}, \cdots, \tilde{E}^H_{\text{doc}}) \tag{9}$$

where $E_{\text{glob}} \in \mathbb{R}^{M \times d}$. $\text{Aggregator}(\cdot)$ is the aggregate function. Many aggregators can be applied, such as weighted average, max-pooling, LSTM, etc. In our experiments, we use weighted average because it performs the best among these aggregators.

The global prediction for the entire hierarchical structure is given by:

$$P_{\text{glob}} = \sigma(W_{\text{glob}} E_{\text{glob}} + b_{\text{glob}}), \tag{10}$$

where $W_{\text{glob}} \in \mathbb{R}^{T \times M}$ is the weight matrix, b_{glob} is the corresponding bias vector, H is the number of hierarchical levels. All the document representations E^h_{doc} and category information are concatenated as the feature representation E_{glob} for the global classifier. This combination step can not only make full use of the feature information at different levels, but also alleviate the over smoothing problem in GCNs. The final predictions P_F consist of local and global predictions:

$$P_F = \beta(P^0_{\text{loc}} \oplus P^1_{\text{loc}} \oplus \cdots P^H_{\text{loc}}) + (1-\beta) P_{\text{glob}}, \tag{11}$$

where $\beta \in [0, 1]$ is a balancing parameter and $\oplus$ is the concatenating operator. Given P^h_{loc} and P_F, the local and global losses are calculated as follows:

$$\begin{aligned} \mathcal{L}_{\text{loc}} &= \sum_{h=1}^{H} \left[\mathcal{E}\left(P^h_{\text{loc}}, \tilde{Y}^h \right) \right], \\ \mathcal{L}_{\text{glob}} &= \mathcal{E}\left(P_F, Y \right), \end{aligned} \tag{12}$$

where $\mathcal{E}$ is the cross-entropy loss. Note that in the local loss function $\mathcal{L}_{\text{loc}}$, the augmented category matrix $\tilde{Y}^h$ is used to replace the original matrix Y^h. The final loss function for the whole framework is

$$\min_W \left(\mathcal{L}_{\text{loc}} + \mathcal{L}_{\text{glob}} + \lambda \|W\|^2 \right),$$

where $\|W\|^2$ is the ℓ_2 norm hyperparameter.

4 Experiments

In this section, we conduct extensive experiments on three real-world datasets to evaluate our proposed approach. We first introduce the datasets and baselines followed by the experimental results.

4.1 Datasets

Experiments are conducted on three real-world datasets: Arxiv Academic Papers dataset, Patent Documents dataset, and WOS-46985 dataset. The statistics of these datasets are summarized in Table 1.

Arxiv Academic Papers Dataset (AAPD) This dataset is built by (Yang et al., 2018). There are 55,840 abstracts of academic papers with the related subjects in this dataset. We process and augment the dataset with a two-level category hierarchical structure. There are 9 subjects (e.g., cs, math) at the first hierarchical level and 52 categories (e.g., cs.CV, cs.AI) at the second level.

Patent Dataset This dataset is collected from USPTO by (Huang et al., 2019a). It includes 100,000 granted US patents, and each patent document is associated with multiple hierarchical categories that are structured as a four-level hierarchical structure.

WOS-46985 This dataset is built by (Kowsari et al., 2017). There are 46985 published papers categorized into seven domains: Computer Science, Electrical Engineering, Psychology, Mechanical Engineering, Civil Engineering, Medical Science, Biochemistry. These papers can be further categorized into 134 sub-domains.

Table 1: Details of datasets.

Statistics	AAPD	Patent	WOS-46985
# instances	55,840	100,000	46,985
# hierarchical levels	2	4	2
# categories in level 1	9	9	7
# categories in level 2	52	128	134
# categories in level 3	-	661	-
# categories in level 4	-	8,364	-
# total categories	61	9,162	141
avg. # words per sample	163	64	128

4.2 Baselines and Parameter Settings

We compare our method with the following baselines:

- **Clus-HMC** (Vens et al., 2008): Clus-HMC is a decision tree based approach. It utilizes the global information to predict all categories at the same time.
- **HMC-LMLP** (Cerri et al., 2016): HMC-LMLP incrementally trains a set of neural networks, each of which is responsible for predicting categories at each hierarchical level.
- **HMCN-F** (Wehrmann et al., 2018): HMCN-F is a feed-forward network based approach that takes predictions of each hierarchical level and the entire hierarchical structure into consideration.
- **HMCN-R** (Wehrmann et al., 2018): HMCN-R is a recurrent version of HMCN-F that combines local and global predictions together.
- **HARNN** (Huang et al., 2019b): HARNN employs hierarchical attentive neural networks to model the dependencies among different levels of the hierarchical structure.

In the experiment, we apply a grid search for hyperparameters: specifically, we tune λ_s^h in [0.4,0.5,...,0.7], β in [0.3,0.4,...,0.7], λ in [0.4,0.5,...,0.8], the learning rate is tuned in [0.0001,0.0005,0.001,0.005].

4.3 Experimental Results

We first conduct experiments on the datasets of AAPD and Patent. The proposed model HVHMC and its variant HVHMC-NEG are compared to the baselines. In HVHMC-NEG, the negative sampling+ strategy is adopted, where we select the example with the farthest distance in the tree label structure as the negative sample. In addition, the triplet loss in NLP tasks (Ein Dor et al., 2018) is used to replace the cross entropy loss in HMHMC. Results are summarized in Table 2 and Table 3.

Table 2: Classification performance on AAPD

Model	Precision	Recall	Micro-F1
Clus-HMC	56.1	51.2	53.5
HMC-LMLP	86.4	70.5	77.7
HMCN-R	86.3	66.8	75.3
HMCN-F	86.6	65.9	74.9
HARNN	86.8	72.3	78.8
HVHMC	87.1	74.8	80.5
HVHMC-NEG	**87.4**	**76.2**	**81.4**

Table 3: Classification performance on Patent

Model	Precision	Recall	Micro-F1
Clus-HMC	41.9	34.5	37.9
HMC-LMLP	69.2	38.0	49.0
HMCN-R	68.4	39.5	50.1
HMCN-F	70.4	37.6	49.1
HARNN	**74.2**	42.5	54.1
HVHMC	73.3	44.2	54.4
HVHMC-NEG	74.1	**45.1**	**56.1**

In the comparison, it is notable that the proposed approaches obtain the best results in terms of all evaluation measures on AAPD. Meanwhile, they also achieve the best performance in Recall and Micro-F1 on the Patent dataset, with a slightly smaller Precision. This justifies incorporating categories when learning representations can provide auxiliary information, which helps to extract the horizontal and vertical dependencies among categories and facilitate the classification process.

The possible reasons that HVHMC-NEG achieves a slightly lower Precision on Patent are two-fold. On the one hand, the text lengths in Patent are relatively short, which increases the sparsity of the document-word graph and affects the model performance. On the other hand, the increasing number of categories per level leads to a reduced number of word nodes belonging to each category, which weakens the label augmentation effect in the horizontal correlations.

Unlike Patent and AAPD, in the WOS-46985 dataset, each instance is associated with only one category at every hierarchical level. As can be observed from Table 4, similarly to the results on Patent and AAPD, HVHMC outperforms the other

approaches by a relatively large margin, which further confirms the effectiveness of exploiting the category representations and correlations.

Table 4: Classification performance on WOS-46985

Model	Precision	Recall	Micro-F1
Clus-HMC	47.1	50.1	48.6
HMC-LMLP	70.3	66.0	68.1
HMCN-R	68.4	63.1	65.6
HMCN-F	69.6	65.3	67.4
HARNN	72.5	**74.1**	73.3
HVHMC	**77.9**	74.0	74.3
HVHMC-NEG	**77.9**	**74.1**	**74.3**

4.4 Ablation Study

We proceed to conduct ablation studies to verify the effect of each component of the proposed HVHMC model. The first investigation is to validate the necessity of a loosely coupled GCN and the effectiveness of the categories introduced in LCGCN. We first construct a tightly coupled GCN that includes three types of nodes in one heterogeneous graph and find it unable to produce reliable representations. To further verify the effects of introducing category information and learning category embeddings in LCGCN, we make predictions based on directly calculating the cosine similarities between each category embedding and all document embeddings in the AAPD dataset and the WOS-46985 dataset. We calculate the average recall rates of the top 20, 50, 70 of the document nodes that are most similar to the category nodes to measure the improvements brought by the category information.

Table 5: Effects of introducing category information

Recall	Top 20	Top 50	Top 70	All
AAPD	86.5	84.2	89.9	74.8
WOS-46985	87.1	88.6	89.3	74.0

In Table 5 we compare the average recall rates of the top 20, 50, 70 similar document nodes with the average recall rate of all documents as shown in Table 2 and Table 4. The average recall of documents nodes close to category nodes in the embedded semantic space is obviously better than the average recall of all document nodes, indicating that the introduction of category information in the loosely coupled GCN provides a label guiding effect when learning the semantic embeddings.

Next, to identify the effects of the horizontal correlations and vertical correlations, we consider three variants of the HVHMC model on AAPD: HVHMC w/o h, which ignores the horizontal correlations, HVHMC w/o v, which ignores the vertical dependencies, and HMHMC-Pur which works without the vertical and horizontal correlations.

Table 6: Comparison of the variants on AAPD

Model	Precision	Recall	Micro-F1
HVHMC w/o h	86.5	74.2	79.9
HVHMC w/o v	**87.1**	72.6	79.3
HVHMC-Pur	86.5	71.9	78.6
HVHMC	**87.1**	**74.8**	**80.5**

From Table 6, we can find that HVHMC achieves the best performance compared to the variants. Removing either the horizontal or the vertical correlations results in a performance degradation of HVHMC w/o h or HVHMC w/o v, which still outperforms HVHMC-Pur, illustrating the importance of incorporating the horizontal and vertical correlations in the proposed HVHMC framework.

4.5 Visualization of Horizontal Correlations

Besides evaluating the classification performance, we also visualize the horizontal correlations among categories which are critical for augmenting the category matrix. Figure 4 illustrates the heatmap of the correlation matrix of the first hierarchical level in AAPD. Each cell represents the correlation between two categories.

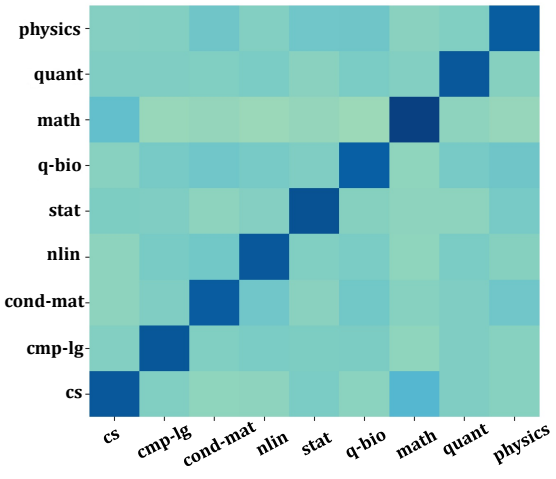

Figure 4: Heatmap of the horizontal correlation matrix

As illustrated in Figure 4, some correlations are reflected in the heatmap. For example, *math* is semantically closer to *cs* than the other categories.

These results indicate that our proposed LCGCN provides an elegant way to capture the horizontal category correlations by learning semantic representations of categories. Compared to estimating category correlations by the co-occurrence of categories in the training data, our approach integrates the association between categories and words, which helps capture the latent semantic correlations.

5 Conclusion

This paper proposes a novel hierarchical multi-label text classification approach named Horizontally and Vertically Hierarchical Multi-label Text Classification (HVHMC). We first design a loosely coupled graph convolutional neural network as the representation layer, capturing the word-to-word, category-to-word, and category-to-category associations. After the category representations are learned, both the horizontal and vertical category correlations are considered to facilitate the hierarchical classification process. Finally, extensive experiments are conducted to verify the effectiveness of the proposed framework.

Acknowledgments

This research was supported by Anhui Provincial Natural Science Foundation (2008085J31).

References

Siddhartha Banerjee, Cem Akkaya, Francisco Perez-Sorrosal, and Kostas Tsioutsiouliklis. 2019. Hierarchical transfer learning for multi-label text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6295–6300, Florence, Italy. Association for Computational Linguistics.

Lijuan Cai and Thomas Hofmann. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 78–87. ACM.

Ricardo Cerri, Rodrigo C Barros, André CPLF de Carvalho, and Yaochu Jin. 2016. Reduction strategies for hierarchical multi-label classification in protein function prediction. *BMC bioinformatics*, 17(1):373.

Eduardo P Costa, Ana C Lorena, André CPLF Carvalho, Alex A Freitas, and Nicholas Holden. 2007. Comparing several approaches for hierarchical classification of proteins with decision trees. In *Brazilian symposium on bioinformatics*, pages 126–137. Springer.

Liat Ein Dor, Yosi Mass, Alon Halfon, Elad Venezian, Ilya Shnayderman, Ranit Aharonov, and Noam Slonim. 2018. Learning thematic similarity metric from article sections using triplet networks. pages 49–54.

Caspar Fall, Attila Törcsvári, K. Benzineb, and G. Karetka. 2003. Automated categorization in the international patent classification. *SIGIR Forum*, 37:10–25.

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2019a. Text level graph neural network for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3444–3450, Hong Kong, China. Association for Computational Linguistics.

Wei Huang, Enhong Chen, Qi Liu, Yuying Chen, Zai Huang, Yang Liu, Zhou Zhao, Dan Zhang, and Shijin Wang. 2019b. Hierarchical multi-label text classification: An attention-based recurrent network approach. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1051–1060.

Thomas Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371.

Guohao Li, Matthias Mueller, Guocheng Qian, Itzel Perez, Abdulellah Abualshour, Ali Thabet, and Bernard Ghanem. 2021. Deepgcns: Making gcns go as deep as cnns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1.

Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, pages 1063–1072. International World Wide Web Conferences Steering Committee.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Jiaming Shen, Wenda Qiu, Yu Meng, Jingbo Shang, Xiang Ren, and Jiawei Han. 2021. Taxoclass: Hierarchical multi-label text classification using only class names. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4239–4249.

Celine Vens, Jan Struyf, Leander Schietgat, Sašo Džeroski, and Hendrik Blockeel. 2008. Decision trees for hierarchical multi-label classification. *Machine learning*, 73(2):185.

Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. 2018. Hierarchical multi-label classification networks. In *International Conference on Machine Learning*, pages 5225–5234.

Yuan Xin, Linli Xu, Junliang Guo, Jiquan Li, Xin Sheng, and Yuanyuan Zhou. 2021. Label incorporated graph neural networks for text classification. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8892–8898. IEEE.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: Sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3915–3926, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.

Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2018. Deep learning on graphs: A survey. *arXiv preprint arXiv:1812.04202*.

Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu. 2020. Hierarchy-aware global model for hierarchical text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1106–1117, Online. Association for Computational Linguistics.