Peer-Label Assisted Hierarchical Text Classification

Junru Song[1], **Feifei Wang**[2,3]*, **Yang Yang**[4]
[1]Institute of Statistics and Big Data, Renmin University of China
[2]Center for Applied Statistics, Renmin University of China
[3]School of Statistics, Renmin University of China
[4]Defense Innovation Institute, Chinese Academy of Military Science
{songjunru,feifei.wang}@ruc.edu.cn, bigyangy@gmail.com

Abstract

Hierarchical text classification (HTC) is a challenging task, in which the labels of texts can be organized into a category hierarchy. To deal with the HTC problem, many existing works focus on utilizing the parent-child relationships that are explicitly shown in the hierarchy. However, texts with a category hierarchy also have some latent relevancy among labels in the same level of the hierarchy. We refer to these labels as *peer labels*, from which the peer effects are originally utilized in our work to improve the classification performance. To fully explore the peer-label relationship, we develop a PeerHTC method. This method innovatively measures the latent relevancy of peer labels through several metrics and then encodes the relevancy with a Graph Convolutional Neural Network. We also propose a sample importance learning method to ameliorate the side effects raised by modelling the peer label relevancy. Our experiments on several standard datasets demonstrate the evidence of peer labels and the superiority of PeerHTC over other state-of-the-art HTC methods in terms of classification accuracy.

1 Introduction

Hierarchical text classification (HTC) is a multi-label text classification problem which aims to classify texts into categories that can be organized into a taxonomic hierarchy. It is an important problem in natural language processing and has attracted increasing attention in both industrial and academic fields. Typical HTC problems include patent categorization (Gomez and Moens, 2014), medical record coding (Cao et al., 2020), and product categorization (Cevahir and Murakami, 2016).

Due to the complexity of category hierarchy, the problem of hierarchical text classification is more challenging than plain text classification. The parent-child relationships between categories in adjacent levels of the hierarchy are usually defined in advance. Then a natural way to solve the HTC problem is to incorporate this prior knowledge into the model, i.e., making the model aware of the hierarchy. Building "hierarchy-aware" models is beneficial for HTC, which is particularly true for categories with few samples. Therefore, it has long been the main focus in HTC to figure out the most effective way of utilizing the category hierarchy to improve the classification performance.

In the past literature, existing approaches for HTC can be generally categorized into three groups: local approaches, global approaches, and local-global-combined ones (also known as hybrid approaches). The local approaches train local classifiers for every child label, every parent label or every level in the hierarchy (Shimura et al., 2018; Banerjee et al., 2019). The parameters of local classifiers are initialized in a top-down fashion according to the category hierarchy. However, these approaches usually contain a large number of parameters, and the whole hierarchy cannot be fully captured merely by parameter initialization. Global approaches, which are popular in recent years, aim to flatten HTC into a multi-label classification problem, and then incorporate the information of category hierarchy in various ways, such as using regularization terms (Gopal and Yang, 2013), modeling the architecture of category hierarchy (Zhou et al., 2020), and using contrastive learning (Wang et al., 2022). The local-global-combined approaches can be seen as an improvement on local approaches, which construct the information flow between local classifiers in more effective ways, and meanwhile utilize a global classifier to coordinate local ones (Wehrmann et al., 2018; Rojas et al., 2020). However, these models might still suffer from error propagation (Rojas et al., 2020), because the classification of child layers are dependent on that of their parents.

To the best of our knowledge, existing methods only exploit category relevancy that is explicitly

* Corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 3747–3758
July 9-14, 2023 ©2023 Association for Computational Linguistics

reflected in the hierarchy. For example, Gopal and Yang (2013) used a recursive regularization term in which the parameters of parent labels are expected to be similar to those of their children. Zhou et al. (2020) used a structure encoder for labels, in which the information of parent and child labels are integrated into each label's representation. However, there could still exist some latent relevancy among the labels in the same level, which could also be beneficial to the HTC problem. Take the BlurbGenreCollection dataset (Aly et al., 2019) as an example, which consists of descriptions and genres of books. In this dataset, two third-level book categories "World History" and "Travel: Asia" belong to different second-level categories "History" and "Travel", respectively. However, these two third-level categories both involve geographical and cultural contents. Therefore, intuitively they should share some common characteristics, and the classification of one category could benefit that of the other. The phenomenon that labels in the same level possess latent relevancy is similar to the "peer effect" existing among peer friends. Thus we call these labels as "peer labels" in this work.

To utilize the latent relevancy of peer labels to improve the HTC performance, we develop a PeerHTC method. It incorporates two types of label relationships: the parent-child relationship explicitly reflected in the hierarchy, and the peer-label relationship implicitly hidden in the hierarchy. We propose several measures to learn the relevancy structure among peer labels, and then utilize the Graph Convolutional Neural Network (GCN) to realize "feature sharing" among peer labels. To address the possible side effect caused by modeling peer labels, we also develop a measure to evaluate the degree of confusion between labels in the same level, and then assign different weights to training samples according to their contribution in alleviating label confusion. The PeerHTC method is realized through an embedded two-stage training approach, in which valuable information about latent relevancy of peer labels and the label confusion can be harvested from the first round of warm-up training and then enhances the second round for final classification.

The rest of this article is organized as follows: Section 2 introduces related works. Section 3 defines the HTC problem. Section 4 introduces the PeerHTC method in detail. Section 5 presents the experimental results on three datasets. Section 6 concludes the article. We share our codes on GitHub[1] for reproducibility.

2 Related Work

2.1 Local Approaches

The local approaches train local classifiers for each category or each level in the hierarchy. These local classifiers are initialized in a top-down fashion according to the category hierarchy so that knowledge learned by each parent classifier can be transferred to their children. For example, the method HTrans (Hierarchical Transfer Learning) (Banerjee et al., 2019) trained a binary classifier for each label, and then initialized the classifiers according to their parents. The method HFTCNN (Hierarchical Fine-tuning Based CNN) (Shimura et al., 2018) trained a multi-label classifier for each level in the category hierarchy, and then followed a similar approach for parameter initialization. However, these models usually have a large number of parameters to estimate and also suffer from insufficient use of the category hierarchy.

2.2 Global Approaches

Global approaches flatten HTC into a simple multi-label classification problem, and seek to incorporate the information of category hierarchy in various ways. For example, Gopal and Yang (2013) imposed recursive regularization on parameters of parent and child nodes. The method HiAGM (Zhou et al., 2020) includes two variants, i.e., HiAGM-LA and HiAGM-TP. In HiAGM-LA, texts and labels are encoded separately, and multi-label attention mechanism is used to extract label-wise features. A structure encoder is also used to aggregate prior category hierarchy information into label embeddings. In HiAGM-TP, label embeddings are not used and text features are directly propagated through the structure encoder. The method HTCInfoMax (Hierarchical Text Classification via Information Maximization) (Deng et al., 2021) seeks to improve HiAGM-LA with mutual information maximization that constrains text and label representations. The method HiMatch (Hierarchy-aware Label Semantics Matching Network) (Chen et al., 2021) projects the representations of words and labels into a common latent space and utilizes hierarchy-aware matching learning. The method HGCLR (Hierarchy-Guided Contrastive Learning) (Wang et al., 2022) models texts and labels separately only

[1] https://github.com/WoodySJR/PeerHTC

in the training process, and then incorporates the information of category hierarchy into the text encoder via contrastive learning.

2.3 Hybrid Approaches

Local-global-combined approaches(or hybrid approaches) can be seen as an improvement on local ones. The method HMCN (Hierarchical Multi-label Classification Networks) (Wehrmann et al., 2018) is probably the first hybrid model. In HMCN, local classifiers are arranged in series and global classification is conducted to coordinate these local classifiers. The method HARNN (Hierarchical Attention-based Recurrent Neural Network) (Huang et al., 2019) is another typical hybrid model. It shares a similar architecture with HMCN, but uses the multi-label attention mechanism to extract label-wise text features. However, since errors in the prediction of higher-level categories may provide misleading information for lower levels, these hybrid approaches might suffer from error propagation (Rojas et al., 2020).

3 Problem Formulation

We define the HTC problem in this section. Specifically, we first give the definition of a category hierarchy and its properties, and then define the HTC problem mathematically.

Definition 1. (Category Hierarchy) Assume there exists an H-level category hierarchy γ. All possible labels in γ are denoted by $\mathbb{C} = \{C^1, C^2, \cdots, C^H\}$, where $C^i = \{c_1, c_2, \cdots\} \in \{0,1\}^{|C^i|}$ is the label set in the i-th level, and $|C^i|$ is the total number of labels in C^i. Consequently, the total number of labels in $\mathbb{C}$ is $K = \sum_{i=1}^{H} |C^i|$. The category hierarchy γ is then defined to be a partially order set $(\mathbb{C}, \prec)$, where $\prec$ represents the superior-subordinate relationship between labels and it satisfies the following three properties:

- asymmetry: $\forall c_x \in C^i$ and $c_y \in C^j$, if $c_x \prec c_y$, then we have $c_y \not\prec c_x$.
- anti-reflexivity: $\forall c_x \in C^i$, we have $c_x \not\prec c_x$.
- transitivity: $\forall c_x \in C^i, c_y \in C^j$ and $c_z \in C^k$, if $c_x \prec c_y$ and $c_y \prec c_z$, then we have $c_x \prec c_z$.

Definition 2. (Hierarchical Text Classification, HTC) Given a category hierarchy $\gamma = (\mathbb{C}, \prec)$, assume there exist a total number of M documents denoted by $\mathbb{D} = \{(D_1, L_1), (D_2, L_2), \cdots, (D_M, L_M)\}$. Here D_d denotes the dth text document, which is typically a sequence of words, i.e., $D_d = \{w_{d1}, w_{d2}, \cdots, w_{dN_d}\}$, where N_d is the total number of words in document D_d. Define $L_d = \{l_{d1}, l_{d2}, \cdots, l_{dH}\}$ to be the label set of the dth document with the i-th level label set $l_{di} \subset C^i$. Then the goal of HTC is to train a classification model Ω based on γ and $\mathbb{D}$. Specifically, for an arbitrary text document D^*, we can predict its label set L^* through the classification model, i.e.,

$$\Omega(D^*, \gamma, \Theta) \rightarrow L^*,$$

where Θ is the parameters in the model Ω.

4 Methodology

In this section, we introduce the PeerHTC method in detail. We first introduce peer label learning and sample importance learning, and then propose a two-stage training procedure. The overall architecture of PeerHTC is illustrated in Figure 1.

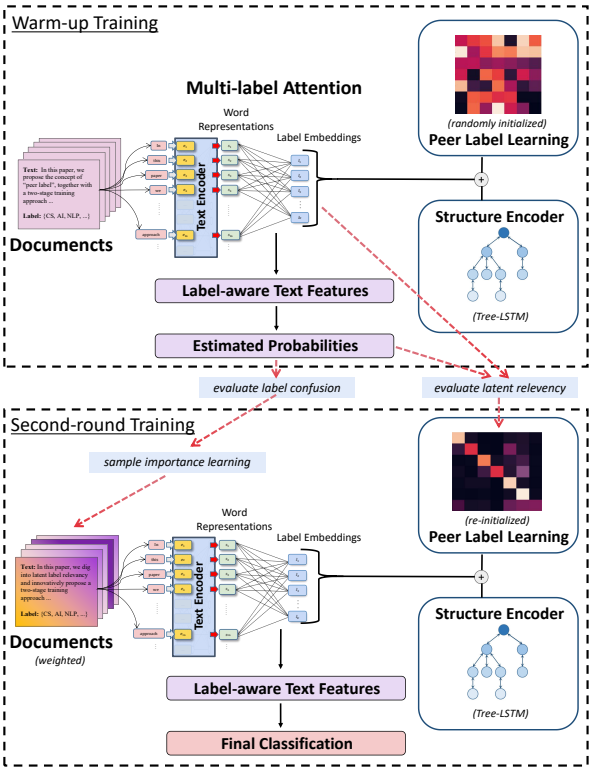


Figure 1: The model architecture of PeerHTC.

4.1 Peer Label Learning

4.1.1 Latent relevancy encoding by GCN

As we mentioned before, there exist latent relationships among peer labels, which are not explicitly expressed in the category hierarchy. Incorporating

the latent relevancy structure among peer labels could also benefit HTC. Motivated by this idea, we consider learning label relevancy from two perspectives. First, we follow the practice of HiAGM (Zhou et al., 2020) to learn the parent-child relationships between labels in adjacent levels. Second, we utilize GCN to incorporate the latent relevancy among peer labels into the label encoder.

We encode labels and texts separately in Peer-HTC. For labels, let $V \in \mathbb{R}^{d_v \times K}$ denote the initial label embeddings, where d_v is the embedding dimension and K is the total number of labels. Then following HiAGM (Zhou et al., 2020), we feed the initial embeddings V into a TreeLSTM encoder to learn the *hierarchy-aware* embeddings $H^{\updownarrow}$, where the symbol "$\updownarrow$" stands for the parent-child relationships. Actually, $H^{\updownarrow}$ is the concatenation of two sets of embeddings derived in top-down and bottom-up fashions, i.e., $H^{\updownarrow} = H^{\uparrow} \oplus H^{\downarrow}$; please refer to Zhou et al. (2020) for more details.

To characterize the latent relationships of peer labels, we use $H^{\leftrightarrow} = \text{GCN}(V)$, which is derived from latent label connections enabled by GCN. We refer to $H^{\leftrightarrow}$as *peer-aware* embeddings in the subsequent analysis. To fully explore the latent relevancy of peer labels, we propose GCN methods using two strategies. The first one is level-wise GCN, which only incorporates connections of labels in the same level. Specifically, define A to be the adjacent matrix that tells how labels should be connected. Define W and b to represent the weight matrix and bias term, which are both trainable. Let $\sigma(\cdot)$ denote ReLU non-linear activation function. Then in level-wise GCN, we first compute $H^{\leftrightarrow}_{(h)}$ for levels $1 \leq h \leq H$, and then concatenate them together. The detailed computation is shown below.

$$\begin{aligned} H^{\leftrightarrow}_{(h)} &= \sigma(A_{(h)} V^{\top}_{(h)} W_{(h)} + b_{(h)})^{\top}, \\ H^{\leftrightarrow} &= \text{concat}\{H^{\leftrightarrow}_{(1)}, H^{\leftrightarrow}_{(2)}, \cdots, H^{\leftrightarrow}_{(H)}\}. \end{aligned} \tag{1}$$

The second method is to use whole-hierarchy GCN, which is a single GCN for labels in the whole hierarchy. This strategy allows for label connections throughout the whole hierarchy. The peer-aware embeddings are then computed as follows:

$$H^{\leftrightarrow} = \sigma(AV^{\top} W + b)^{\top}. \tag{2}$$

After computing the hierarchy-aware embeddings $H^{\updownarrow}$ and peer-aware embeddings $H^{\leftrightarrow}$, we concatenate them together. Specifically, we have $H^{\leftrightarrow}$ concatenated column-wise with $H^{\updownarrow}$, the result of which is then put through a non-linear projection. This leads to the final label embeddings H^*, which is computed as follows:

$$H^* = \sigma\{W^* \cdot (H^{\updownarrow} \oplus H^{\leftrightarrow})\},$$

4.1.2 Initialization of the adjacent matrix A

By using GCN as the latent relevancy encoder to model peer labels, we need to specify the adjacent matrix A in advance, i.e., to tell how labels should be associated with graph edges. To this end, we propose a data-driven approach to initialize A. Inspired by the idea of knowledge distillation (Hinton et al., 2015), the estimated label probabilities (also called soft labels in knowledge distillation) contain extra knowledge on the relationships among different labels. Therefore, the estimated label probabilities can be regarded as a good source to learn the latent relevancy among peer labels. Specifically, if two labels are closely related with each other, their estimated label probabilities should tend to be correlated on the same sample. Therefore, a similarity measure between the estimated probabilities of two labels can reflect how closely they are related. Besides, recall that we have computed the label embeddings H^*, the similarity among which could also reflect label relevancy.

Based on the above considerations, we propose two methods for initializing A. In the first method, we adopt the non-parametric Spearman Rank Correlation Coefficient (SRCC) to measure the similarity between estimated probabilities. Let p_{dk} be the estimated probability of the dth document associated with the kth label. Recall there are a total of M documents. Hence we can compute the rank of p_{dk} among the estimated probabilities of the M documents (i.e., $p_{1k}, ..., p_{Mk}$), which is denoted by r_{dk}. We then compute $\bar{r}_k$, which is the average of r_{dk} among M documents. Then, we can compute absolute SRCC for any two labels k and j as follows

$$\rho^{\text{rank}}_{kj} = \left| \frac{\sum_{d=1}^{M} (r_{dk} - \bar{r}_k)(r_{dj} - \bar{r}_j)}{\sqrt{\sum_{d=1}^{M} (r_{dk} - \bar{r}_k)^2 \sum_{d=1}^{M} (r_{dj} - \bar{r}_j)^2}} \right|. \tag{3}$$

The SRCC measure can be computed on either training samples or test samples, since it does not require true labels.

In the second method, we measure label relevancy based on label embeddings. Specifically, let h_k and h_j be the embeddings for labels k and j, which are extracted from H^*. Then we can use the

absolute cosine similarity between them to measure their relevancy, i.e.,

$$\rho_{kj}^{\text{emb}} = \left| \frac{h_k^\top h_j}{\|h_k\| \cdot \|h_j\|} \right|. \tag{4}$$

After getting the similarity measures (ρ_{kj}^{rank} or ρ_{kj}^{emb}), they are aligned into a matrix and then normalized row-wise (except for the diagonal entries so they remain to be one). This leads to two matrices A^{rank} and A^{emb}, which would then be used in the initialization of GCN. We empirically compare the performance of different initialization methods of A; see Section 5.2.3 for the detailed results.

4.1.3 Multi-label attention

We adopt the multi-label attention mechanism to extract label-wise text features (Huang et al., 2019; Zhou et al., 2020; Deng et al., 2021). For the dth document with N_d words, let $\widetilde{s}_d = \{s_{d1}, ...s_{dN_d}\}$ denote the word representations derived from a text encoder. Recall that h_k is the embedding of label k, which is extracted from H^*. Then within the dth document, we can compute the attention score $\alpha_{kn}^{(d)}$ between the representation of the nth word and the embedding of label k, i.e.,

$$\alpha_{kn}^{(d)} = \frac{\exp\{s_{dn}^\top h_k\}}{\sum_{g=1}^{N_d} \exp\{s_{dg}^\top h_k\}}.$$

The value $\alpha_{kn}^{(d)}$ indicates how informative the nth word is to a certain label k within one document.

Note that in PeerHTC, label embeddings have now included two parts of information, i.e., the hierarchical relationship between parent and child labels and the latent relationship between peer labels. Hence, the attention score $\alpha_{kn}^{(d)}$ is also equipped with the ability to identify text features favoring labels closely related to label k. This leads to reinforced feature sharing. Finally, we calculate a weighted average $u_k^{(d)} = \sum_{g=1}^{N_d} \alpha_{kn}^{(d)} s_{dg}$ for label k. These features are then flattened and fed into a fully-connected network for final classification.

4.2 Sample Importance Learning

4.2.1 A metric for label confusion

Assisted by GCN to model the latent relevancy of peer labels, we achieve reinforced feature sharing that would enhance the classification of one category with the help of text features extracted by other closely related categories. However, a side effect emerges when we strengthen the similarity between the embeddings of peer labels by GCN. That is, it would make easily confused labels become even less distinguishable. To characterize this phenomenon, we first formalize a new concept called "label confusion". Specifically, we say there is confusion between two labels k and j, when one document belongs to label k but gets a high probability in another label j, or the other way around. Take two book categories named "Classics" and "Poetry" for example. They are prone to confusion since both of them involve some genteel expression. More intuitively, label confusion is pretty much like the case where a person gets confused when distinguishing between very similar objects.

To tackle this potential side effect, we first propose a metric to evaluate how easily any two labels can be confused. Let $\mathcal{L}(d)$ denote the true label set of the dth document. Assume we have label $k \in \mathcal{L}(d)$ but label $j \notin \mathcal{L}(d)$. Then the estimated probability of label j measures the confusion between these two labels on this document. To formulate this idea mathematically, let p_{dk} be the estimated probability that the dth document belongs to label k. Let c_{kj} denote the degree of confusion between labels k and j. Denote the index set $D_{kj} = \{d : 1 \le d \le M, k \in \mathcal{L}(d), j \notin \mathcal{L}(d)\}$. Then we can compute c_{kj} as follows,

$$c_{kj} = \frac{1}{|D_{kj}|} \sum_{d \in D_{kj}} p_{dj}. \tag{5}$$

4.2.2 Training with sample weighting

A document sample is said to be important in distinguishing labels k from j, if its label set contains k but not j. With the metric of label confusion c_{kj}, we can evaluate the importance of each training sample. Specifically, define β_{dk} to be the importance of the dth document with respect to a label k. Then in the case $k \notin \mathcal{L}(d)$, we set $\beta_{dk} = 1$. In the case $k \in \mathcal{L}(d)$, we specify β_{dk} as follows:

$$\beta_{dk} = 1 + \sum_{j \notin \mathcal{L}(d)} \{\exp(\tau c_{kj}) - 1\}, \tag{6}$$

where τ is a temperature hyperparameter controlling how radical we are in assigning sample weights. We then plug β_{dk} into the binary cross entropy loss (BCE) function, which is popularly used in HTC (Nam et al., 2014), i.e.,

$$L = -\sum_{d \in \mathbb{D}} \sum_{k \in \mathbb{C}} \beta_{dk} \{y_k \log(p_{dk}) + (1 - y_k) \log(1 - p_{dk})\},$$

where y_k is either 1 or 0 depending on whether $k \in \mathcal{L}(d)$ or not.

4.3 A Two-Stage Training Approach

As mentioned in Section 4.1 and Section 4.2, we use a data-driven approach to identify the adjacent matrix A and sample importance weights β_{dk}'s, which all rely on the estimated label probabilities and label embeddings. To obtain the adjacent matrix and sample importance weights, we develop a two-stage training approach. The first round is a warm-up training stage. We randomly initialize the adjacent matrix A in GCN, and assign equal weights to all training samples. Then we train the PeerHTC model for the first time. The obtained estimated label probabilities and label embeddings from the warm-up training are then used to compute the adjacent matrix and sample importance weights. Then we re-train the PeerHTC model for the second time with the updated adjacent matrices and sample weights. This leads to the final classification model. The whole procedure is illustrated in Algorithm 1.

Algorithm 1: The Two-Stage Training Approach of PeerHTC

Input: Training documents $\mathbb{D}$ and validation/test documents $\mathbb{D}^*$; label hierarchy γ;
Output: A hierarchical text classifier Ω.
for ***Warm-up training:*** **do**
- *Step 1.* Randomly initialize the adjacent matrix A in GCN, and assign equal weights to training samples;
- *Step 2.* Train the PeerHTC model and then save the estimated label probabilities on training and test samples, together with the label embeddings H^*;
- *Step 3.* Compute the adjacent matrices A^{rank} or A^{emb};
- *Step 4.* Compute the label confusion measure c_{jk} and the sample weight β_{dk}.

for ***Second-round training:*** **do**
- Re-initialize the adjacent matrix in GCN with A^{rank} or A^{emb}, and plug sample weights β_{dk} into the loss function. Then re-train the PeerHTC model to get the final classifier.

5 Experiments

5.1 Experimental Setup

Datasets. We use three datasets to explore the classification performance. The first one is Web-of-Science (WOS) dataset (Kowsari et al., 2017), which consists of abstracts of published papers from journals in Web of Science. The disciplines that each paper belongs to are regarded as the classification labels. The second dataset is BlurbGenreCollection (BGC) (Aly et al., 2019), which consists of advertising descriptions of books. The genres of books are regarded as classification labels, while the advertising descriptions are regarded as text documents. The last dataset consists of the textual names of retailing products (we refer to as Goods), which are collected by ourselves from a Chinese retailing company. In this dataset, each product belongs to a three-level product hierarchy. Among the three datasets, WOS and Goods are both for single-path HTC, i.e., each sample only has one single label in each level, whereas samples in BGC have multi-path labels, i.e., each sample is allowed to have multiple labels in the same level. Each dataset is randomly split into the training set (70%), validation set (15%) and test set (15%). The descriptive statistics of the three datasets are listed in Table 1. In addition, the intended use of public datasets and pre-trained models, as specified in their license or terms, was strictly obeyed in our work.

Statistics	BGC	WOS	Goods
# categories	146	141	225
# categories in level 1	7	7	20
# categories in level 2	46	134	80
# categories in level 3	77	-	125
# categories in level 4	16	-	-
# hierarchical levels	4	2	3
# average categories per instance	3.01	2	3
# average tokens per instance	157.5	250.0	12.9
# instances	91,892	46,985	14,969
# instances in training set	64,324	32,889	10,478
# instances in validation set	13,784	7,048	2,245
# instances in test set	13,784	7,048	2,246

Table 1: Descriptive Statistics of Datasets.

Baselines. In order to demonstrate the effectiveness of PeerHTC, we compare it with three naive approaches that treat HTC as a simple multi-label classification problem, along with four state-of-the-art HTC models. The three naive approaches are LSTM (Hochreiter and Schmidhuber, 1997), TextRCNN (Lai et al., 2015), and BERT (Devlin et al., 2018). The four state-of-the-art HTC models are briefly introduced as follows.

(1) *HMCN* (Wehrmann et al., 2018). It is probably the first hybrid approach that combines a sequence of local classifiers with global optimization.

(2) *HARNN* (Huang et al., 2019). It is also a hybrid approach, but utilizes attention mechanism and refines how information flows between levels.

(3) *HiAGM-LA* (Zhou et al., 2020). It encodes labels and documents separately, and utilizes multi-label attention mechanism to extract hierarchy-

aware text features.

(4) *HTCInfomax* (Deng et al., 2021). It is basically an improvement on top of HiAGM-LA via mutual information maximization.

Evaluation metrics. To measure the classification performance, we apply two standard evaluation metrics, i.e., the Micro-F1 and Macro-F1 (Gopal and Yang, 2013). Micro-F1 computes the overall precision and recall of all the labels, while Macro-F1 computes the average of F1 scores of all labels. As a result, Micro-F1 assigns greater weights to more frequent labels, while Macro-F1 treats all the labels equally.

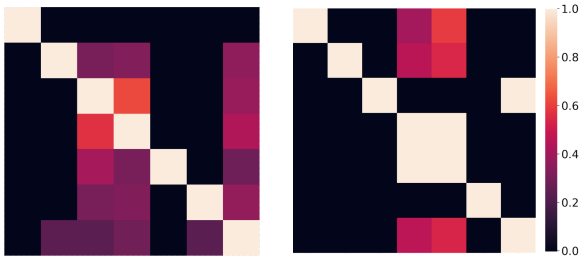
Implementation details. We use BERT as the text encoder in PeerHTC, and set the dimension of label embeddings as 256. The BERT models are pretrained on "book_corpus_wiki_en_uncased" and "wiki_cn_cased" for English and Chinese datasets respectively, both of which have 12 hidden layers and 768 hidden units. The vocabulary is created with words that appear no less than 5 times. We set the maximum length of token inputs as 100. The threshold for tagging is chosen to be 0.5. Model parameters are initialized according to the Xavier uniform (Glorot and Bengio, 2010) when random initialization is needed. We use the Adam optimizer (Kingma and Ba, 2014) with momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, a learning rate $\alpha = 1 \times 10^{-5}$, and a mini-batch size of 64. To prevent overfitting, we also use dropout (Srivastava et al., 2014) with the rate of 0.1, and weight decay (Loshchilov and Hutter, 2017) with the tuning weight equal to 1×10^{-7}. The parameter τ in equation (6) is set as 1.2.

For HTC competitors, the same parameter settings are adopted. We follow their original practices to use simple text encoders, but also replace them with BERT for a fair comparison. Specifically, in HARNN, HiAGM-LA and HTCInfoMax, we use a single-layer bidirectional LSTM as the text encoder; in HMCN, we use three parallel CNN layers with filter sizes $\{3, 4, 5\}$ and numbers of channel $\{100, 70, 70\}$ as the text encoder. For simple text encoders, 300-dimensional pretrained word embeddings GloVe (Pennington et al., 2014) and Fasttext (Bojanowski et al., 2017) are used on English and Chinese datasets respectively. Our hyperparameters are tuned on the validation set, taking both classification performance and the computation resources available into consideration, and the classification performances reported in our experimental results are evaluated on the test set. Our models are trained on two Tesla P100 GPUs.

5.2 Experimental Results

5.2.1 Evidence of peer labels

In order to demonstrate the existence of peer labels, we take the BGC and WOS datasets as examples and show the adjacent matrices of their first-level labels, which are computed using estimated label probabilities on training samples according to equation (3). As shown by Figure 2, some labels have higher degrees of relevancy with others, which can serve as useful prior knowledge for classification. For example, in the BGC dataset, "Classics" and "Poetry", corresponding to the intersection of the third row and fourth column in Figure 2(a), are closely related. In the WOS dataset, "Mechanical Aerospace Engineering (MAE)" and "Civil Engineering" (in the fourth row and fifth column) show extremely high relevancy. These findings verify the existence of peer label relevancy. However, compared with BGC and WOS datasets, the latent relevancy of peer labels is relatively weak for the Goods dataset, which is not shown to save space. This is largely due to the fact that the Goods dataset has a larger number of categories, which are more fine-grained and thus less relevant with each another.



(a) BGC (b) WOS

Figure 2: The adjacent matrices of first-level categories in the BGC and WOS datasets.

5.2.2 Comparison results

To explore the classification performance of PeerHTC, we compare this model with (1) naive classification approaches (i.e., LSTM, TextRCNN, BERT), and (2) state-of-the-art HTC approaches (i.e., HMCN, HARNN, HiAGM-LA, HTCInfoMax). For the HTC approaches, we also replace their original text encoders with BERT for better classification performance and a fair comparison with PeerHTC. To characterize the latent relevancy

of peer labels in PeerHTC, the GCN methods with the "level-wise" and "whole-hierarchy" strategies are considered; see Equations (1) and (2) for details. To decide the adjacent matrix A in GCN, we consider three methods: (1) $A^{\text{rank-train}}$, computed by the estimated label probabilities on training samples; (2) $A^{\text{rank-test}}$, computed by the estimated label probabilities on test samples; and (3) A^{emb}, computed by label embeddings. This results in a total of $2 \times 3 = 6$ choices for A. We report the performances of all these different choices for a comparison.

Table 2 reports the classification results of different models, from which we can draw the following conclusions. First, our method PeerHTC has achieved better classification performance than all naive approaches and HTC approaches, when evaluated by both Micro-F1 and Macro-F1 in the three datasets. Second, when replacing the traditional text encoders in HTC approaches by BERT, their classification performances have been largely improved. This reveals the great potential of pretrained models in HTC problems. Even so, our method PeerHTC still outperforms HTC approaches with BERT on the BGC and WOS datasets. On the Goods dataset, however, the method HTCInfoMax achieves the best performance while PeerHTC ranks second with comparable results. We believe that this slightly poorer performance of PeerHTC mainly results from the weak latent relevancy among peer labels, as remarked in Section 5.2.1.

Model	BGC		WOS		Goods	
	F1(c)	F1(a)	F1(c)	F1(a)	F1(c)	F1(a)
Naive approaches						
LSTM	48.08	26.49	49,23	33.80	88.29	75.91
TextRCNN	58.76	38.46	68.75	60.14	89.50	77.95
BERT	68.19	48.12	66.47	58.29	91.29	79.59
HTC approaches						
HMCN	63.77	43.02	70.31	61.62	89.24	77.73
HARNN	63.92	43.31	70.46	62.10	87.68	75.21
HiAGM-LA	67.62	48.65	72.44	63.21	89.87	78.27
HTCInfoMax	68.48	48.02	73.90	64.49	88.98	78.29
HTC approaches with BERT as text encoder						
HMCN	76.46	61.72	73.13	64.97	91.21	80.31
HARNN	75.92	59.98	72.60	65.45	91.89	81.06
HiAGM-LA	76.35	62.09	73.22	65.48	91.43	80.91
HTCInfoMax	76.66	61.11	73.54	65.08	**92.79**	**82.41**
Our approach						
PeerHTC	**77.47**	**63.54**	**74.24**	**67.38**	92.61	82.10

Table 2: Experimental results of different methods. F1(c) and F1(a) represent Micro-F1 and Macro-F1.

5.2.3 Ablation study

To further demonstrate the advantages of incorporating peer labels and using sample weights, we conduct the following ablation study. Specifically, we compare the following three models. The first one, denoted by "NA", is a naive HTC model without considering peer labels or sample weights. The second one, denoted by "OPL", is a variant of PeerHTC that only considers the latent relevancy among peer labels, but does not utilize the sample weights. The last one is our proposed PeerHTC, which considers both peer labels and sample weights. In addition, to explore the performances of using different adjacent matrices A, we report OPL with six different adjacent matrices, as described in Section 5.2.2. The detailed results are shown in Table 3. It is obvious that, nearly all OPL methods, as well as the PeerHTC method, have obtained better classification performance than the NA method. These results suggest that leveraging peer effect is beneficial to hierarchical text classification.

Models		BGC		WOS		Goods	
		F1(c)	F1(a)	F1(c)	F1(a)	F1(c)	F1(a)
NA		76.35	62.09	73.22	65.48	91.43	80.91
		whole-hierarchy					
OPL	$A^{\text{rank-train}}$	77.18	63.23	74.24	66.12	92.02	81.65
	$A^{\text{rank-test}}$	76.98	62.74	73.55	65.91	92.21	82.51
	A^{emb}	76.99	62.75	73.05	66.04	92.33	81.84
		level-wise					
	$A^{\text{rank-train}}$	77.06	62.81	73.72	66.84	92.14	81.72
	$A^{\text{rank-test}}$	76.62	62.57	73.54	66.97	92.19	82.03
	A^{emb}	77.09	62.71	73.99	66.59	92.22	81.95
PeerHTC		77.47	63.54	74.24	67.38	92.61	82.10

Table 3: Experimental results of ablation study. F1(c) and F1(a) represent Micro-F1 and Macro-F1.

We then compare the performances of using different adjacent matrices in OPL. As shown by Table 3, on the dataset BGC, the "whole-hierarchy" strategy works better in most cases. On the dataset WOS, the "level-wise" strategy generally works better. On the Goods dataset, the performances of "level wise" and "whole hierarchy" are rather comparable. When it comes to adjacent matrices computed using either sample probabilities or label embeddings, there is no obvious distinction between their performances, indicating all these methods can be helpful in revealing latent relationships among peer labels.

Finally, we focus on the effect of using sample weights. As we mentioned in Section 4.2, characterizing the latent relevancy of peer labels would create shortcuts between labels and may have potential side effect of label confusion. To cope with this problem, we first measure the degree of label confusion, then identify the importance of different training samples in alleviating label confusion, and finally plug these weights into the BCE loss func-

tion. As shown by Table 3, the PeerHTC method can improve the classification performance consistently on the three datasets, when compared with the OPL method. These results demonstrate the usefulness of sample weights in PeerHTC.

6 Conclusion

In this work, we originally propose the concept of "peer labels" to characterize the phenomenon that labels in the same level have latent relevancy. To utilize these peer labels to enhance HTC, we develop the PeerHTC method. We exploit GCN as an encoder for latent relevancy among peer labels, and reinforce feature sharing among these closely related peer labels. We also use a novel technique to assign training sample weights based on their importance in alleviating label confusion. The above procedures are embedded in a two-stage training approach. Our experimental results demonstrate the evidence of peer labels in real datasets and the generally better performance of PeerHTC against other state-of-the-art HTC methods. In terms of application, one would expect a higher lift in performance from PeerHTC when the granularity of categorization is relatively low, as demonstrated by our experimental results. We also suggest that one should carry out exploratory analysis or take into account domain knowledge, in order to decide the extent of peer-label relevancy for a specific dataset.

Limitations

In this work, we adopt a data-driven approach to identifying latent relevancy. However, we believe that external knowledge such as knowledge graphs could also be of great help for this purpose, and is thus one of the directions of our future work.

Acknowledgements

Feifei Wang is the corresponding author. This work is supported by National Natural Science Foundation of China (No.72001205), the Fundamental Research Funds for the Central Universities and the Research Funds of Renmin University of China (21XNA026). Besides, we thank all the anonymous reviewers for their valuable suggestions.

References

Rami Aly, Steffen Remus, and Chris Biemann. 2019. Hierarchical multi-label classification of text with capsule networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 323–330.

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.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Shengping Liu, and Weifeng Chong. 2020. Hypercore: Hyperbolic and co-graph representation for automatic icd coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3105–3114.

Ali Cevahir and Koji Murakami. 2016. Large-scale Multi-class and Hierarchical Product Categorization for an E-commerce Giant. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 525–535.

Haibin Chen, Qianli Ma, Zhenxi Lin, and Jiangyue Yan. 2021. Hierarchy-aware label semantics matching network for hierarchical text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4370–4379.

Zhongfen Deng, Hao Peng, Dongxiao He, Jianxin Li, and Philip S Yu. 2021. HTCInfoMax: A global model for hierarchical text classification via information maximization. *arXiv preprint arXiv:2104.05220*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Juan Carlos Gomez and Marie-Francine Moens. 2014. A survey of automated hierarchical classification of patents. In *Professional search in the modern world*, pages 215–249. Springer.

Siddharth Gopal and Yiming Yang. 2013. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD international conference*

on Knowledge discovery and data mining, pages 257–265.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Wei Huang, Enhong Chen, Qi Liu, Yuying Chen, Zai Huang, Yang Liu, Zhou Zhao, Dan Zhang, and Shijin Wang. 2019. 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.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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. IEEE.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classification—revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer.

Jeffrey Pennington, Richard Socher, and Christopher D 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.

Kervy Rivas Rojas, Gina Bustamante, Arturo Oncevay, and Marco A Sobrevilla Cabezudo. 2020. Efficient strategies for hierarchical text classification: External knowledge and auxiliary tasks. *arXiv preprint arXiv:2005.02473*.

Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto. 2018. HFT-CNN: Learning hierarchical category structure for multi-label short text categorization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 811–816.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Zihan Wang, Peiyi Wang, Lianzhe Huang, Xin Sun, and Houfeng Wang. 2022. Incorporating Hierarchy into Text Encoder: a Contrastive Learning Approach for Hierarchical Text Classification. *arXiv preprint arXiv:2203.03825*.

Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. 2018. Hierarchical multi-label classification networks. In *International conference on machine learning*, pages 5075–5084. PMLR.

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.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations are discussed in Section 7.

☒ A2. Did you discuss any potential risks of your work?
To the best of our knowledge, our work does not involve any potential risk.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
The abstract is at the very beginning of the paper, while the introduction is in section 1.

☒ A4. Have you used AI writing assistants when working on this paper?
We did not use any AI writing assistant.

B ☑ Did you use or create scientific artifacts?

Some scientific artifacts(eg. datasets and pre-trained models) were used in our experiments in Section 5.

☑ B1. Did you cite the creators of artifacts you used?
They are cited in Section 5.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section 5.1.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 5.1.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Data used in our work do not involve such problems.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Introduction to the datasets is included in Section 5.1.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Yes. In Section 5.

C ☑ Did you run computational experiments?

Section 5.

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Our computing infrastructure is introduced in Section 5. The number of parameters and GPU hours involved in our experiments are on a reasonable scale and we believe they will not cause any difficulty to reproduction. Thereby, they are omitted to save space.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Our experimental results are derived on the test set in a single run, which is pointed out in Section 5.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Such implementation details are included in Section 5.1.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.