Mining Knowledge within Categories in Global and Local Fashion for Multi-Label Text Classification

Sheng Bi, Peng Shi, Yuntao Du, Bin Jin, Lingshuang Yu
National Key Laboratory for Novel Software Technology, Nanjing University
Department of Computer Science and Technology
Nanjing, China
benson.bi.cs@gmail.com,spwannasing@gmail.com,dz1833005@smail.nju.edu.cn, kimbin@smail.nju.edu.cn,yu.lingshuang@outlook.com

Abstract—Multi-label text classification (MLTC) is an important task in natural language processing, which assigns multiple labels to each text in the dataset. Typical method like Binary Relevance (BR) is arguably the most intuitive solution for the task. It works by decomposing the multi-label learning task into a number of independent binary learning tasks while ignoring the correlation between labels. Recently, neural network models attract much attention. Researchers view the MLTC task as a sequence generation problem. Although some new methods based on generative model (e.g. sequence-to-sequence), such as novel decoder structure and various attention mechanisms, can improve the performance. These methods still have some shortcomings, such as unreasonable loss function, unclear ordering of target labels. To address these limitations, we propose a simple and effective novel model, which combines the merits of neural network and BRs methods. Our model also takes into account the categories and levels of labels. We decompose the MLTC problem to binary classification, together with global and local extractor to avoid the impact of label ordering and cumulative error. Experimental results show that our model achieves an improvement of 3.0% micro-F1 and a reduction of 6.0% hamming loss on AAPD dataset compared with the state-of-the-art work. And obtained good performance on RCV1-V2 dataset.

Index Terms—Multi-label, Nerual Networks, Global-and-Local Extractor

I. INTRODUCTION

Multi-label text classification (MLTC) is an important task in natural language processing (NLP), which assigns multiple labels to each text in the dataset. MLTC enables a broad range of applications, and one common real-world scene is the news’ labels classification. In order to accurately and effectively recommend the news to its right users, we need to classify the news. Usually, the content of news involves multiple topics (labels), and this is where multi-label classification comes in.

To tackle this task, many efficient methods had been proposed. Binary Relevance (BR) [33] transforms the MLTC into multiple single-label classification problem. Specifically, BR procedure works in an independent manner, where the binary classifier for each class label is learned by ignoring the existence of other class labels [22]. Hence, many correlation-enabling extensions to binary relevance have been proposed [19], [23], [24], [35]. Recently, neural network models have made a remarkable achievement in NLP. Inspired by the tremendous success of the sequence-to-sequence (seq2seq) model, there are many innovations [31], [41], [43] based on it, such as novel decoder structure and various attention mechanisms, and achieve new state-of-the-art performance.

Typical approach (BR), unlike seq2seq model, dose not produce cumulative error and the prediction of a single label will not be disturbed by all labels. But on the other hand, this independence causes it to ignore correlations between labels, which might weaken the performance of task.

Seq2seq model, however, suffers from several problems. First, labels are predicted by RNN-based decoder relying on a predefined ordering of labels. Previous studies [1], [2], [30] show that label ordering in dataset greatly affects the model performance. Moreover, setting a strict order to acquire the perfect label order is too costly. Second, seq2seq model is trained with the maximum likelihood estimation (MLE) method and the cross-entropy loss function. Labels with more occurrences are more likely to be predicted, and the correlations among labels cannot be captured by the model without label interaction. Third, at one timestep, the probability of all labels is converted only from the low-dimension hidden state of decoder, which fails to make full use of the text. Moreover, as the timestep increases, the error of label prediction will accumulate when a previous timestep cannot correctly predict a label.

To address these issues, we propose a novel framework that combines the advantages of BR and neural networks. In other words, some relatively independent classifiers are learned by BR, and the fitting ability of neural network is also utilized to extract some global and local information. With MLTC problem been decomposed into binary classification for each label, recurrent neural network (RNN) and convolutional neural networks (CNN) is selected to fully extract the information from the text. As shown in Figure 1, Global and local extractors using neural networks have extracted the information that they care about respectively, which will greatly help to improve the performance of MLTC task. In this method, since no generative method is used to get the label sequence, the relevance of labels is captured through Extractor without cumulative error.

In addition, we observe that there are category relationships and hierarchical relationships among labels. Intuitively, in
MLTC task, the quantity of label is large and often artificially pre-specified. There will be an objective hierarchical relationship between labels, or labels can be divided into multiple categories. Labels in the same category share some knowledge. Therefore, we assign a specific local extractor to each category, so that this extractor can accurately extract the information that the category cares about from the sentence. Ideally, as shown in Figure 1, when computer science-related information appears in the text, the corresponding local extractor can discover it.

Inspired by these, we propose the Global-Locally Encoding (GLEN) Model. GLEN utilizes global extractors to learn knowledge among the text that plays a fundamental role in the results of classification. Then some local extractors (category-specific) are used to extract different information of categories.

The experimental results on real-world datasets including AAPD [31], RCV1-V2 [6] show that our model achieves the state-of-art performance compared to all baseline methods. GLEN obtains 0.720 micro-F1 and 0.0236 hamming loss on AAPD dataset, outperforming prior state-of-the-art work by 3.0% and 6.0%. On RCV1-V2 dataset, our model achieves 0.870 micro-F1 and 0.0080 hamming loss.

The contributions of this paper are listed as follows:

- Unlike recent efforts to improve the seq2seq model, we propose a new framework which combines the merits of neural network and BRs methods. It captures the relevance of labels and will not produce cumulative error.
- To the best of our knowledge, we are the first to take the category of labels into account in MLTC task. We use global and local extractors to obtain knowledge shared by all categories and knowledge within categories. And the trick of label statement is used to integrate the hierarchical relationship among labels.
- Experimental results demonstrate that our model outperforms all baseline models and achieves the state-of-the-art performance on the dataset AAPD and RCV1-V2.

II. RELATED WORK

Multi-label text classification is one of the most important task in NLP. Many efforts have been invested in it. Early work on exploring the MLTC task focuses on machine learning algorithms, mainly including problem transformation methods and algorithm adaptation methods [2]. Problem transformation methods, such as Binary Relevance (BR) [33], Label Powerset (LP) [34], and Classifier Chains (CC) [35], map the MLTC task into multiple single-label learning task. Algorithm adaptation method extend specific learning algorithm to handle multi-label data directly. The corresponding representative work is ML-DT [27], Rank-SVM [26], ML-KNN [16] and so on.

Typical method -BR is one of the simplest and most popular transformation methods. Its main drawback is that it does not consider dependencies between labels. However, it has been shown that in many cases, BR can yield predictive performance as good as more complex methods depending on the characteristics of the data [20], [21].

Recent years, the research turned to the application of neural networks. Zhang [36] proposed a neural network algorithm named BP-MLL, which is the multi-label version of backpropagation. Kurata [37] proposed a neural network initialization method to treat some of the neurons in the final hidden layer as dedicated neurons.

Since RNN-based approach (e.g. seq2seq) works well on MLTC task, more and more researchers focus on it. Chen [39] proposed an ensemble application of CNN and RNN to capture both the global and the local textual semantics. The model belongs to encoder-decoder pattern, in which CNN acts as encoder and RNN is used for decoding. Yang [2] and Li [42] proposed to view the multi-label classification task as a sequence generation problem. A seq2seq model with a novel decoder (global embedding) structure is proposed by Yang [2]. Li [42] proposed a label distributed seq2seq model with a novel soft loss function to solve the problem.

To further improve the seq2seq model, some methods of attention mechanism have been proposed. Lin [41] proposed a semantic-unit-based dilated convolution model based on seq2seq. A corresponding hybrid attention mechanism and multi-level dilated convolution are implemented to extract both the information at the word-level and the level of the semantic unit. Wang [44] proposed ranking-based AutoEncoder with a word-vector-based self-attention.

However, some problems inherent in seq2seq are difficult to solve. Training a RNN decoder requires a predefined order of label. Although Nam [29] and Yang [31] compare several ordering strategies and suggest ordering positive labels by frequency directly in descending order (from frequent to rare labels), it is unnatural to impose a strict order on labels, which may break down label correlations in a chain [30]. Yang [2] proposed a novel sequence-to-set framework utilizing deep reinforcement learning, which reduce the dependence on the label order. Tsai [30] proposed a framework based on optimal completion distillation and multitask learning to solve this. Chen [3] proposed the order-free RNN to dynamically decide
In Section 3.4. Finally, we present the **Scoring Module** in Section 3.5.

Formally, the source sentence \( x = \{x_1, \ldots, x_i, \ldots, x_n\} \) and the target label sequence \( t = \{t_1, \ldots, t_i, \ldots, t_m\} \) are both given in the dataset, where \( n, m \) are the corresponding lengths. \( x_i \) is a word in the sentence, and \( t_i \) is a label. Given the label space with \( L \) labels, we divide all labels into \( k \) categories. For the convenience of explanation, we set \( k \) to 4, that is, we assume that there are four categories, and they are recorded as Category \( A \) to Category \( D \). The number of elements (labels) in these four categories is \( l_A, l_B, l_C \) and \( l_D \). The purpose is to correctly predict the involved label based on the sentence.

### A. Overview

Our model is shown in Figure 3. Our model consist of three components: **Global Encoding Module**, **Local Encoding Module** and **Scoring Module**.

Given a sentence \( x \), we first compute the probability of \( l_A \) labels under category \( A \), and then calculate the probability of other labels under category \( B \) to category \( D \).

Under category \( A \), we use **Global Encoding Module** shared by all categories and **Local Encoding Module** specific to category \( A \) to encode the sentence \( x \). As for labels, we first expand \( l_A \) labels in category \( A \) to label statements, and pass through the **Global Encoding Module** and **Local Encoding Module**, and further simplified by the self-attention layer. Then, the encoding of labels seperately read the encoding of sentence, and fuse the results to get the score of each label in current category by **Scoring Module**. Next, we select the labels that receive a score above a threshold.

After dealing with all categories, we integrate the labels selected by different categories as the final output.

### B. Assign a Category to Each Label

We assign a category to each label. In general, the labels are often pre-specified and each label have practical meaning. And when the quantity of labels is large, in order to improve the work efficiency, these labels will be artificially divided into certain categories.

In the AAPD and RCV1 datasets used in our experiments, these categories of information can be explicitly extracted, which will be described in detail in the **Experimental Settings** in Chapter 4. However, previous studies have ignored these categories of information and treated all labels at the same level.

Then, in our model, we regard categories (e.g. label "category \( A \)"") as first-level labels and actual labels (e.g. label "\( a_1 \)"") as second-level labels.

To make full use of the first and second level labels, rather than simply using a trainable embedding matrix to encode label, we expand the label into an assignment statement so that more information could be obtained by time sequence encoding.

We use the trick of the label statement here. As shown in Figure 3, label \( a_1 \) is expanded to statement,

\[
<s>A = a_1 < /s>
\] (1)
Fig. 3. The overview of our proposed model. Assume that all labels are grouped into 4 categories and they are recorded as Category A to Category D. Our model consists of three components: Global Encoding Module, Local Encoding Module and Scoring Module.

where A is a category with the same dimension as the label embedding. We treat these categories as special labels. Specifically, we treat the labels as hierarchical structures, the category labels are on the first level, and the actual labels are on the second level. In this way, our model can learn not only the knowledge within the category, but also the hierarchy of labels through both global and local fashion.

C. Global Encoding Module

Global Encoding Module is shared by all categories. This module consists of three components: a global sentence CNN (convolutional neural network), a global sentence LSTM (bidirectional Long Short-Term Memory) and a global label LSTM. In order to avoid repeated calculation, we only run the global module once for encoding a sentence $x$ when dealing with different categories.

Consider the global sentence LSTM first. It encodes the source sentence from both directions and generates the hidden states for each word,

$$h^g_i = \text{LSTM}^g \left( h^g_{i-1}, x_i \right), \quad h^g_i = \text{LSTM}^g \left( h^g_{i+1}, x_i \right) \quad (2)$$

where the superscript $g$ represents global LSTM. Then the hidden state from both directions at each time step are concatenated ($h^g_i = [h^g_i, h^g_i]$). The global label LSTM has the same structure.

As for the global sentence CNN, unlike CNN-RNN model [39] in which CNN is used as an encoder, in our model we apply CNN to extract text features and structure information. The way to use CNN is similar to the one used in [9]. Each word in sentence $x$ will look up the randomly initialized word embedding matrix. The sentence is then a concatenation of word vectors $w_i$. Thus, a sentence $x$ of length $n$ is represented as:

$$S = w_{1:n} = w_1 \oplus w_2 \oplus \ldots \oplus w_n \quad (3)$$

where $\oplus$ is the vector concatenation operator. So, we can treat the sentence as a "image", and perform convolution on it via linear filters. In natural language, several adjacent words are usually used together to form a phrase. Thus, filters with different window sizes are used to extract this information, which may be helpful for classification. We use $w_{i;i+h-1}$ to represent the sub-sentence. A convolution operation involves a filter $W_{CNN}$ and a bias term.

$$o_i = W_{CNN}^T \cdot S_{i:i+h-1} + b \quad (4)$$

where $i = 1 \ldots n - h + 1$, and $\cdot$ is the dot product between the filter vector and the word vector. This filter is applied repeatedly to each possible window of $h$ words in the sentence to produce a feature map $o = [o_1, o_2, \ldots, o_{n-h+1}]$. Then, we apply a $1 - \text{max-pooling}$ to each feature map to induce a fixed-length vector. This process is repeated by different filters with different window sizes.

Next, concatenate the output vector of filters and project it into a lower dimensional vector $h^c$ by a full-connect layer, where superscript $c$ indicates CNN, as shown in Figure 3.

D. Local Encoding Module

Local Encoding Module consists of two components: a local sentence LSTM and a local label LSTM. Each category has its own unique LSTM network to extract the information it cares about. The local sentence LSTM and the local label LSTM have the same structure as the global LSTM.

Consider the local sentence LSTM first under category $A$,

$$h^A_i = \text{LSTM}^A \left( h^A_{i-1}, x_i \right), \quad h^A_i = \text{LSTM}^A \left( h^A_{i+1}, x_i \right) \quad (5)$$
where the superscript \( A \) represents local sentence LSTM. Then the hidden state from both directions at each time step are concatenated \((h_t^A = \left[ h_t^A; \bar{h}_t^A \right])\).

The outputs of the two (global and local) LSTMs for sentence are combined through a category-specific scalar parameter \( \alpha^A \) to yield a global-local encoding \( h^A \) of sentence \( x \), where superscript \( s \) indicates sentence and \( \alpha^A \) is a trainable parameter. Now \( h^s \) denotes that both universal information for classification and unique information for category \( A \) are obtained.

\[
h^s = \alpha^A h^0 + (1 - \alpha^A) h^A \tag{6}
\]

Next, consider the encoding of labels. Similar to the process of sentence encoding, \( h^l \) is calculated by passing the label statement to global label LSTM and local label LSTM, where superscript \( l \) means label. The global label LSTM can process all the labels in different categories and learn the general knowledge applicable to classification contained in the label. And the local label LSTM (for category \( A \)) is only used to extract the information of the label under the current category.

Then, different from sentence encoding, we compute a self-attention context \( c^l \) over \( h^l \) aiming to transform \( h^l \) into a vector. Self-attention or intra-attention, is a special case of attention mechanism that only requires a single sequence to compute its representation, which has been applied to many tasks [10]–[13]. It also provides a more flexible way to select, represent and synthesize the information of the inputs [14]. In our case, for \( i \)th element in \( h^l \),

\[
a^l_i = W_{aa} h^l_i + b \tag{7}
\]

\[
p^l = \text{softmax} \left( a^l \right) \tag{8}
\]

where the subscript of \( W_{aa} \) indicates self-attention.

The self-attention context \( c^l \) is then the sum of each element \( h^l_i \), weighted by the corresponding normalized self-attention score \( p^l_i \).

\[
c^l = \sum_i p^l_i h^l_i \tag{9}
\]

### E. Scoring Module

Intuitively, we can determine whether the sentence expressed the label under one category by examining two input source, \( h^c \) and \( h^s \). The first source is the timing encoding of the sentence \( x \). Labels are interested in words at certain positions in the sentence.

\[
a^t_i = (h^s_i)^\top c^l \tag{10}
\]

\[
p^l = \text{softmax} \left( a^t \right) \tag{11}
\]

\[
g^l = \sum_i p^l_i h^s_i \tag{12}
\]

\[
y^t_i = W_1 g^l_i + b \tag{13}
\]

where \( t \) indicates timing information. The score \( y^t \) indicates the degree to which the label was expressed by the sentence \( x \) under certain category.

The second source is the output of structure information encoding module \( h^c \). This source studies the association between label and adjacent words collocations. All labels under the current category \( c^l \) get the probability of their classification by reading the structural information \( h^c \) of the sentence .

\[
y^c = (c^l)^\top h^c \tag{14}
\]

The final score \( y \) is then a weighted sum between two source \( y^t \) and \( y^c \), normalized by the sigmoid function \( \sigma \).

\[
y = \sigma \left( y^t + uy^c \right) \tag{15}
\]

where the weight \( u \) is a adjustable parameter. The dimensions of \( y^t, y^c \), and \( y \) are all \( k \), which is the number of labels under one category.

Finally, we use binary cross-entropy loss to estimate the loss of prediction. It is a sigmoid activation plus a cross-entropy loss. Unlike softmax loss and cross-entropy loss used in generative model [2], it is independent for each label (class), meaning that the loss computed for every class is not affected by other class. That’s why it is used in our model, were the insight of an element belonging to a certain class should not influence the decision for another class. Below is the loss calculation between the score of label \( y \) under one category and groundtruth \( t \) of sentence \( x \).

\[
loss = -\frac{1}{K} \sum_{i=1}^{k} t_i \cdot \log(y_i) + (1 - t_i) \cdot \log(1 - y_i) \tag{16}
\]

### IV. Experiment

In the following, we evaluate our proposed model on two datasets. We first introduce the dataset, evaluation metrics, all baselines, experimental settings and results. Finally, we perform ablation experiments and discussion.

#### A. Datasets

**Arxiv Academic Paper Dataset (AAPD):** This dataset is provided by Yang [31]. It consists of the abstract and corresponding subjects of 55,840 papers in arxiv. An academic paper may have multiple subjects and there are 54 subjects in total, such as \( \text{cs.IT}, \text{math.CO}, \text{math.IT}, \text{quant-ph} \). To be specific, the training set contains 53840 samples, while the validation set and test set contain 1000 samples respectively. The statistic information of the two datasets is shown in Table II.

<table>
<thead>
<tr>
<th>Label ( A )</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C11</td>
<td>Strategy, new companies, joint ventures, consortia, diversifications, investment.</td>
</tr>
<tr>
<td>E12</td>
<td>Monetary/economic policy and intervention, interest rates.</td>
</tr>
<tr>
<td>G15</td>
<td>All European Community affairs.</td>
</tr>
<tr>
<td>M11</td>
<td>Stock exchanges, performance of equities.</td>
</tr>
</tbody>
</table>

**Reuters Corpus Volume I (RCV1-V2)**: This dataset is provided by Lewis [6]. It consists of over 800,000 manually

\[1\]http://www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm
categorized newswire stories made available by Reuters Ltd for research purposes. Multiple topics can be assigned to each newswire story and there are 103 topics in total. To be specific, the training set contains 802414 samples, while the validation set and test set contain 1000 samples respectively. The topics in the dataset are desensitized, such as C11, C12, E521, ECAT. However, the actual meaning of each topic (text interpretation) as shown in Table I and the tree structure of the topics are additionally given in the data set. This means that there is a hierarchical relationship between the topics.

B. Evaluation Metrics

Following the previous studies [16], [39], we adopt hamming loss and micro-F1 score to evaluate the performance of our models. For reference, the micro-precision as well as micro-recall are also reported.

Hamming-loss [31], [38], [41] evaluates the fraction of misclassified instance-label pairs, where a relevant label is missed or an irrelevant is predicted.

\[
HL = \frac{1}{L} \sum \mathbb{1}(y \neq \hat{y})
\]  

Micro-F1 [31], [40] can be interpreted as a weighted average of the precision and recall. It is calculated globally by counting the total true positives, false negatives, and false positives.

\[
\text{microF}_1 = \frac{\sum_{j=1}^{L} 2tp_j}{\sum_{j=1}^{L} 2tp_j + fp_j + fn_j}
\]

C. Baselines

In the following, we introduce the baseline models with which our model compares.

- **Binary Relevance (BR) [33]** transforms the MLC task into multiple single-label classification problems by ignoring the correlations between labels.
- **Classifier Chains (CC) [35]** transforms the MLC task into a chain of binary classification problems and takes high-order label correlations into consideration.
- **Label Powerset (LP) [34]** transforms a multi-label problem to a multi-class problem with one multi-class classifier trained on all unique label combinations.
- **CNN [9]** uses multiple convolution kernels to extract text features, which are then inputted to the linear transformation layer followed by a sigmoid function to output the probability distribution over the label space.
- **CNN-RNN [39]** proposes a CNN and RNN based method that is capable of efficiently representing textual features and modeling higher-order label correlation.
- **Seq2Seq [31]** applies the sequence-to-sequence model to perform multi-label text classification.

D. Experimental Settings

We implement our experiments in PyTorch on an NVIDIA 1080Ti GPU. The size of the vocabulary is 50000 and out-of-vocabulary (OOV) words are replaced with unk for both datasets. We use the Adam [8] optimization method to minimize the binary cross-entropy loss over the training data. Follow [31], for the hyper-parameters of the Adam optimizer, we set the learning rate \( \alpha = 0.001 \), two momentum parameters \( \beta_1 = 0.9 \) and \( \beta_2 = 0.999 \) respectively, and \( \epsilon = 1 \times 10^{-8} \). We set dropout [7] to 0.2.

For the RCV1-V2 dataset, each document is truncated at the length of 500. We set the (randomly intialized) word embedding size to 256. As for the number of categories, no specific information is displayed in the label, such as C11, E12, G15 and M11. However, as shown in Table I, each label has an actual text interpretation, and all labels are tree-shaped. We can use this information to assign categories to labels. We use the first letter of label to distinguish categories. Therefore, it can be determined that there are four categories of labels: C, E, G, M. All four categories are in the first layer of topic hierarchy tree which is shown in official website. For example, topic ECAT belongs to category E.

Therefore, for the encoding of sentence, one global LSTM and four local LSTM for four categories are used. For the encoding of labels, the same number of LSTMs are configured. The hidden size of the bi-LSTM is 128. It should be noted that the dimension of word embedding must be twice the dimension of hidden size of bi-LSTM due to the inner product structure information encoding module. we use 10 filters of each the four window-size (4,5,6,7), which result in a 400-dimension output after concatenation. And this output be projected into a 256-dimension vector \( h^c \) by a full-connect layer. We set the parameter \( u \) in equ.15 to 0.5.

For the AAPD dataset, we use 100 filters of each the four window-size (2,3,4,5). As for the number of categories, we use the prefix of the label (subject) to determine the category. We divided the subjects into three categories: cs, math, and the rest. For example, subject cs.cv starts with cs., so it is assigned to category cs. Subject physics.soc-ph is classified as category the rest. To be specific, among the total of 54 subjects, there are 33 subjects in category cs and 8 subjects in category math, while category the rest contains 13 subjects.

The size of (randomly initialized) label embedding is 256. Hence, one global LSTM and three local LSTM are configured for the sentence encoding and label encoding. The rest of the parameters are the same as that in the RCV1-V2 dataset.

E. Results

The experiment results of our model and the baselines on both datasets are shown in Table IV and Table III.
analyze the effectiveness of different components of our model. We perform some ablation experiments on the AAPD dataset to understand the impact of each component on the model's performance. Our model still achieves the prior best work in terms of micro-F1 score over the state-of-the-art performance. Similar to the results on RCV1-V2 dataset, our model still achieves the prior best work with less parameters on RCV1-V2 dataset.

V. Ablation Test on AAPD Dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>HL(-)</th>
<th>P(+)</th>
<th>R(+)</th>
<th>F1(+)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR</td>
<td>0.0216</td>
<td>0.644</td>
<td>0.648</td>
<td>0.646</td>
</tr>
<tr>
<td>CC</td>
<td>0.0306</td>
<td>0.657</td>
<td>0.651</td>
<td>0.654</td>
</tr>
<tr>
<td>LP</td>
<td>0.0312</td>
<td>0.662</td>
<td>0.608</td>
<td>0.634</td>
</tr>
<tr>
<td>CNN</td>
<td>0.0256</td>
<td>0.849</td>
<td>0.545</td>
<td>0.664</td>
</tr>
<tr>
<td>CNN-RNN</td>
<td>0.0278</td>
<td>0.718</td>
<td>0.618</td>
<td>0.664</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>0.0251</td>
<td>0.746</td>
<td>0.659</td>
<td>0.699</td>
</tr>
<tr>
<td>Our model</td>
<td>0.0236</td>
<td>0.770</td>
<td>0.677</td>
<td>0.720</td>
</tr>
</tbody>
</table>

Table III presents the experimental results on AAPD test set. It indicates that our model achieves a reduction of 6.0% in hamming loss and an improvement of 3.0% in micro-F1 score over the state-of-the-art performance. Similar to the results on the AAPD test set, our model still achieves the prior best work with less parameters on RCV1-V2 dataset.

F. Ablation Test and Discussion

Considering the feasibility of computing resources, we perform some ablation experiments on the AAPD dataset to analyze the effectiveness of different components of our GLEN model. The results of these experiments are shown in Table V.

In the table, we mark the process of global sentence CNN as text global-cnn, the global and local sentence LSTM as text global-local-rnn and the process of label statement encoding as label global-local-rnn, and . And - text global-cnn in the table means that we do not implement text global-cnn module in our model, and the rest of our model are working properly.

Table V Ablation Test on AAPD dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>HL(-)</th>
<th>P(+)</th>
<th>R(+)</th>
<th>F1(+)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our model</td>
<td>0.0236</td>
<td>0.770</td>
<td>0.677</td>
<td>0.720</td>
</tr>
<tr>
<td>- text global-cnn</td>
<td>0.0024</td>
<td>0.781</td>
<td>0.645</td>
<td>0.706</td>
</tr>
<tr>
<td>- text global-local-rnn</td>
<td>0.0253</td>
<td>0.803</td>
<td>0.578</td>
<td>0.672</td>
</tr>
<tr>
<td>- label global-local-rnn</td>
<td>0.0238</td>
<td>0.779</td>
<td>0.654</td>
<td>0.711</td>
</tr>
</tbody>
</table>

Decomposing multi-label classification task into multiple binary classification tasks can effectively avoid the accumulation of errors caused by the generative method (seq2seq). The way to avoid the disadvantage of seq2seq is to apply the traditional classification loss function. Therefore, we use binary cross-entropy loss (BCE) in our model, which leads to an improvement in micro-precision (P) metric. This is due to the fact that some labels with low probability will be trained to tend to 0, the most directly manifestation of which is that the model will not output its own uncertain results. At the beginning of training stage, the prediction is usually empty.

Temporal order is important than structure information. As shown in Table V, the performance of the model without text global-local-rnn is weaker than the model without text global-cnn, which suggest that capturing temporal dependencies is helpful for understanding phrases for classification. Because of the cooperation between two parts, text global-cnn and text global-local-rnn, our model can achieve the desired results.

Label encoding with label statement trick and self-attention works better than ordinary label embedding. We observe that there is a significant decrease in performance when remove the label global-local-rnn. This stems from the flexibility in the attention context computation afforded by the self-attention mechanism, which allows the model to focus on selecting sentence history relevant to the current label. Moreover, label statement trick is integrated into the primary and secondary labels, which makes the label encoding process more efficient.

V. CONCLUSION

In this paper, we propose a simple and effective novel framework, named Global- Locally Encoding (GLEN). Experimental results show that GLEN achieves the state-of-the-art performance. Our model learns information within categories by encoding sentences and labels in both global and local fashion. In addition, through the method of label statement, we integrated the hierarchical relationship of labels into the model for learning. And our model does a good job of avoiding the problems that might arise with a generative approach, such as error accumulation and unclear label ordering.

VI. ACKNOWLEDGEMENT

This paper is supported by the Natural Science Foundation of China (Grant No: U1811462), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

REFERENCES
