Article
Multilabel Text Classification Algorithm Based on Fusion of Two-Stream Transformer

Lihua Duan 1, Qi You 2, Xinke Wu 2 and Jun Sun 2,*

1 School of IoT, Wuxi Institute of Technology, 1600 Gaolang West Road, Wuxi 214121, China; duanlihua@wxit.edu.cn
2 School of Artificial Intelligence and Computer Science, Jiangnan University, Lihu Avenue, Wuxi 214122, China; 7171905005@stu.jiangnan.edu.cn (Q.Y.); 619194012@stu.jiangnan.edu.cn (X.W.)
* Correspondence: junsun@jiangnan.edu.cn

Abstract: Existing multilabel text classification methods rely on a complex manual design to mine label correlation, which has the risk of overfitting and ignores the relationship between text and labels. To solve the above problems, this paper proposes a multilabel text classification algorithm based on a transformer encoder–decoder, which can adaptively extract the dependency relationship between different labels and text. First, text representation learning is carried out through word embedding and a bidirectional long short-term memory network. Second, the global relationship of the text is modeled by the transformer encoder, and then the multilabel query is adaptively learned by the transformer decoder. Last, a weighted fusion strategy under the supervision of multiple loss functions is proposed to further improve the classification performance. The experimental results on the AAPD and RCV1-V2 datasets show that compared with the existing methods, the proposed algorithm achieves better classification results. The optimal micro-F1 reaches 73.4% and 87.8%, respectively, demonstrating the effectiveness of the proposed algorithm.

Keywords: multilabel text classification; transformer; attention mechanism

1. Introduction

Text classification is one of the fundamental tasks in the field of natural language processing, aiming at the correct classification management of large amounts of texts from different sources. In the traditional single-label text classification task, each text corresponds to only one category label, and the labels are independent of each other, and the classification granularity is relatively coarse. The corresponding algorithm research and application have become increasingly mature. In contrast, the multilabel text classification (MLTC) task is more difficult to assign two or more class labels to text, but it is also closer to the actual scene. It has a wide range of applications in the fields of information retrieval [1,2], web mining, question answering systems, and sentiment analysis. Due to the wide variety of labels, complex correlations, and imbalanced sample distribution, it poses a huge challenge to build simple and effective multilabel text classifiers.

Traditional machine learning multilabel text classification algorithms are mainly divided into two categories: problem transformation and algorithm adaptation. The former transforms the multilabel classification problem into a series of single-label classification problems, and the latter improves the existing single-label algorithm to make it suitable for multilabel data. Since traditional methods rely on feature engineering and are easily affected by noise, there are still shortcomings in the prediction effect. In recent years, with the rapid development of deep learning, multilabel text classification algorithms based on deep neural networks have received extensive attention. Techniques [3–6], such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer, mine the internal relationship of text to obtain robust text representation to improve multilabel classification effect and generalization. However, these methods often ignore the semantic
information of labels, and recent sequence-to-sequence (Seq2Seq) methods to model label correlations further promote the development of multilabel classification tasks. Among them, Yang et al. [7] proposed the sequence generation model (SGM) to apply the idea of sequence generation to label prediction. On this basis, Yang et al. [8] further exploited the disorder of the set to reduce the risk of accumulating false predictions. However, such methods are prone to overfitting the label combination distribution in the training set, and there is an exposure bias phenomenon [9]. There are also some methods that are not based on the Seq2Seq structure and have achieved good results by utilizing the label information, such as joint embedding of text and labels [10], using label attention [11], label-based pretraining [12], and introducing iterative inference mechanisms [13].

While existing multilabel text classification algorithms have made some progress towards modeling internal textual relationships and capturing label correlations, the exploitation of the relationship between text and labels is less mature. In response to this problem, this paper proposes a simple and effective method that uses the cross-attention mechanism in the transformer decoder to adaptively extract the independent dependencies between text and each type of label. This not only avoids the risk of overfitting the label order, but also enables the ability to learn label embeddings end to end, greatly improving the potential of the model. Specifically, this paper regards each type of label as a label query (LQ) in the transformer decoder, and calculates cross-attention with the input text representation to collect the discriminative features related to the category and then predict in the follow-up. Query the possibility of existence of each type of label.

In order to further enhance the modeling of the global relationship within the text, this paper uses the self-attention mechanism in the transformer encoder to capture the long-distance dependencies of characters in the text, and introduces position encoding (PE) to transfer position information. The robust text representation is then fed into the above decoder. Finally, this paper proposes a novel distribution-balanced loss function, which weights and fuses the classification results of the encoder and decoder from the decision level with low computational cost under the supervision of multiple loss functions.

This paper conducts experiments on the AAPD and RCV1-V2 datasets, showing that the proposed method achieves superior performance with a small number of computational resources and demonstrating the effectiveness of each component of the method. The main contributions of this paper are listed as follows:

1. A simple and effective method based on a transformer decoder is proposed, which effectively extracts the connection between text and class labels through the cross-attention mechanism for multilabel text classification, avoiding the risk of label combination overfitting.
2. A robust text representation is obtained through the self-attention mechanism, and positional encoding of the transformer encoder is used to model the internal relationship of the text. A weighted fusion strategy is proposed under the supervision of multiple loss functions.
3. Experimental analysis on two multilabel text classification datasets demonstrates the superiority and effectiveness of the proposed method.

2. Related Work

The current mainstream multilabel text classification methods can be divided into two categories, namely, traditional machine-learning-based methods and deep-learning-based methods. In particular, the traditional machine learning methods can further be classified into the problem transformation method and the algorithm adaptation method according to different solving strategies. The problem transformation method converts multilabel classification into multiple single-label classification tasks, such as the binary relevance (BR) [14] algorithm, which directly performs a separate binary classification for each label to achieve multilabel classification. However, the performance of this method is not ideal because it ignores label correlations. On this basis, a classifier chain (CC) [15] was proposed in 2011. It treats labels as a sequence structure and transforms the problem
into a chain of binary classifiers, where the input of each binary classifier depends on the results of the previous classifiers. A label powerset (LP) [16] considers label correlations, taking each possible label combination as a new class and transforming the problem into a multiclassification problem. The algorithm adaptation is to improve single-label classification algorithms to adapt to multilabel classification. For example, a ranking support vector machine (Rank-SVM) [17] uses ranking loss to optimize linear classifiers to improve support vector machines to adapt to multilabel data. A multilabel decision tree (ML-DT) [18] improves decision tree algorithms for multilabel classification. The multilabel K-nearest neighbor (ML-KNN) [19] is based on the K-nearest neighbor algorithm to obtain the adjacent category labels, and maximizes the posterior probability to obtain the label set.

With the continuous development of deep learning, many multilabel text classification algorithms based on deep neural networks have been proposed. For example, Kim [3] used CNN for text classification for the first time. Lai et al. [4] further integrated the advantages of RNN. Chang et al. [6] fine-tuned the transformer model for extreme multilabel text. Among them, the method based on Seq2Seq modeling label correlation has achieved good results. The sequence generation model, proposed by Yang et al. [7], leads the development of subsequent research. After that, Yang et al. [8] introduced the idea of reinforcement learning to reduce the influence of incorrect labels. Qin et al. [20] proposed an adaptive RNN to find the optimal label order. Although this type of method improves the classification performance, it is also susceptible to overfitting of the label order. Some subsequent methods, which are not based on the Seq2Seq structure, have obtained good performance by using text and label joint embedding, label attention, label pretraining, iterative reasoning mechanism, and other strategies [10–13,21]. However, none of the above methods can adaptively extract the specific relationship between text and labels end to end. Therefore, this paper proposes a method of learning label query through cross-attention to aggregate category-specific discriminative features for multilabel classification.

3. Model

The overall framework of the multilabel text classification algorithm based on the fusion of a two-stream transformer (JE-FTT) proposed in this paper is shown in Figure 1. The text and labels first go through the joint embedding module. The encoder models the text embedding to obtain the global features of the text. The decoder models the label embedding as a label query to obtain the label-specific text local features and, finally, sends them to the classifier respectively. The final classification result is obtained through a weighted fusion strategy supervised by multiple loss functions.

![Figure 1. The overall framework of the JE-FTT model.](image-url)
3.1. Text Representation

Text representation includes two parts: word embedding and contextual semantic embedding. For each original text in a given text set $T$, the input text sequence $t = \{w_1, \cdots, w_i, \cdots, w_m\}$ is obtained through preprocessing operations, such as word segmentation and removal of stop words, where $w_i$ refers to the $i$-th word in the text sequence, and $m$ represents the total length of the text. Then the pretrained Word2vec [22] is used to obtain the word embedding $X \in \mathbb{R}^{m \times d}$. We define the embedding matrix as $E \in \mathbb{R}^{d \times |V|}$, where $d$ is the dimension of the word embedding, and $|V|$ is the size of the vocabulary. The specific process is shown in Equation (1).

$$X = \{x_1, \cdots, x_i, \cdots, x_m\} = E(w_1, \cdots, w_i, \cdots, w_m)$$

The contextual semantic embedding part is implemented using bidirectional long short-term memory (Bi-LSTM), which avoids the problem of RNN gradient disappearance and better captures bidirectional semantic dependencies. For the input word embedding $X = \{x_1, \cdots, x_i, \cdots, x_m\}$, the forward and backward hidden layer states of each word vector $x_i$ are obtained as shown in Equations (2) and (3).

$$\overset{\to}{h}_i = \text{LSTM}(\overset{\to}{h}_{i-1}, x_i)$$

$$\overset{\leftarrow}{h}_i = \text{LSTM}(\overset{\leftarrow}{h}_{i-1}, x_i)$$

The final hidden layer state of each word vector is spliced from its two directions to obtain the entire text representation $H \in \mathbb{R}^{m \times 2d_h}$, where $d_h$ represents the dimension of the hidden layer state.

$$h_i = [\overset{\to}{h}_i, \overset{\leftarrow}{h}_i]$$

$$H = \{h_1, \cdots, h_i, \cdots, h_m\}$$

3.2. Encoder

In this paper, contextual semantic information is extracted from the source text through text representation learning, but the modeling of long-term dependencies is lost due to the mechanism of sequential computation. Therefore, the self-attention mechanism of the transformer encoder is further used to establish a connection for any position of the text, retain long-distance information, and fully model the global relationship within the text.

The overall structure of the encoder and decoder is shown in Figure 2, which is composed of $N$ identical layers stacked respectively. Each layer contains two main components, including the multihead attention mechanism (MHA) and the feed forward network (FFN). Each component is followed by a residual connection (RC) and layer norm (LN).

The attention mechanism maps a query ($Q$) and a set of key ($K$)-value ($V$) pairs into a weighted sum of values, where the weight of each value is determined by the correlation between the query and the corresponding key calculated by the function. The multihead attention mechanism is an extension of the attention mechanism. The query, key, and value are projected into multiple different subspaces to calculate the attention in parallel so that different information can be paid attention to. Finally, the multihead attention is obtained by splicing and projection. This paper uses scaled dot-product attention and then extends it to multiheads. The specific calculation process is shown in Equations (6)–(8).

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_j)W^O$$
where $d_k$ is the dimension of the input query and key, and $head_i$ represents the result of the $i$-th attention head. We define the corresponding projection parameter matrix as $W^Q_i, W^K_i, W^K_i' \in \mathbb{R}^{d_{model} \times d_k}$, where $d_{model}$ is the attention dimension, and $j$ represents the number of attention heads, and the final projection matrix is $W^K_i' \in \mathbb{R}^{d_{model} \times d_{model}}$.

$$HE_i' = \text{MultiHead}(HE_{i-1}, HE_{i-1}, HE_{i-1})$$  \hspace{1cm} (10)  

$$HE_i = \text{FFN}(HE_i')$$  \hspace{1cm} (11)  

where $HE_i$ and $HE_i'$ represent the output and intermediate result of the encoder layer of the $i$-th layer, respectively, and the value of $i$ ranges from 0 to $N$. For simplicity, the residual connections and layer normalization behind multihead and FFN are omitted in the process. In particular, $HE_0 = H + PE(H)$ denotes the addition result of the input text representation and the corresponding position encoding as the input of the first encoder layer.

### 3.3. Decoder

To efficiently extract the independent dependencies between text and each class of labels, this paper further uses a transformer–decoder structure based on a cross-attention mechanism to adaptively learn label queries.

Similar to the encoder structure used in the previous section, the decoder is also composed of $N$ layers. The only difference is that the multihead attention used by the encoder is based on the self-attention mechanism, while the decoder is based on a cross-attention mechanism. The query, key, and value for calculating attention come from different source inputs. The label embedding $LQ \in \mathbb{R}^{L \times d_{model}}$ is used as the query, and the

Figure 2. The overall structure of the encoder and decoder.
encoder to finally output $\text{HE}_N$ is used as key and value. The detailed calculation process can be expressed as follows:

$$LQ_i' = \text{MultiHead}(LQ_{i-1}, \text{HE}_N, \text{HE}_N)$$ (12)

$$LQ_i = \text{FFN}(LQ_i')$$ (13)

where $L$ represents the number of label categories, and $LQ_i$ and $LQ_i'$ represent the output and intermediate results of the $i$-th decoder layer, respectively. Unlike the original transformer that needs to perform autoregressive prediction and uses mask attention, each type of label query in this paper can be decoded in parallel, which improves the computational speed.

More specifically, inspired by [23–25], this paper uses a learnable initial label embedding $LQ_0$, which can learn label correlations from the data end to end to derive more appropriate initial tag embedding. In addition, this paper removes the self-attention mechanism of the original transformer decoder because the update of the label query in the design of this paper relies on the linear change in the calculation of the cross-attention process, resulting in the update of the label query by the self-attention somewhat being redundant. Therefore, the self-attention in the decoder can be removed to reduce the computational cost while maintaining sufficient representation power without affecting the classification performance.

Through the design of this paper, the label query can adaptively learn the independent dependencies between the text and each type of label through the cross-attention mechanism, and update the label query in each iteration. Finally, after $N$ iterations, the final label query is output as $LQ_N \in \mathbb{R}^{L \times \text{dim}_\text{model}}$. It contains rich information related to the label category in the corresponding text, which can be used for subsequent multilabel text classification prediction.

### 3.4. Weighted Fusion Strategy Supervised by Multiple Loss Functions

According to the encoder–decoder structure, two output results can be obtained, each containing two different kinds of information useful for multilabel classification. The $\text{HE}_N$ output by the encoder contains the global relationship information inside the text, and the $LQ_N$ output by the decoder contains the specific relationship between the text and the label category information. It is not difficult to find that the information contained in either output alone enables multilabel text classification predictions. However, it is an obvious intuition to effectively fuse the two kinds of information to obtain better classification results. Although the internal information of the text and the label category information have been implicitly fused to a certain extent at the feature level through the cross-attention mechanism in the decoding process, further fusion is also necessary at the final classification decision level.

This finding motivates the innovation of this paper, so we propose a novel weighted fusion strategy supervised by multiple loss functions. The specific fusion process is shown in Equations (14)–(18):

$$O_E = \text{Sigmoid}(\text{ClsHead}_E(\text{HE}_N))$$ (14)

$$O_D = \text{Sigmoid}(\text{ClsHead}_D(LQ_N))$$ (15)

$$W_E = \frac{O_E}{O_E + O_D}$$ (16)

$$W_D = 1 - W_E$$ (17)

$$O_F = W_E O_E + W_D O_D$$ (18)

where $\text{ClsHead}_E$ and $\text{ClsHead}_D$ represent the linear layer classification heads used for the encoder $O_E$ and decoder $O_D$, respectively. The classification results of the encoder and decoder (i.e., $W_E$ and $W_E$) represent their corresponding fusion weights. $O_F$ represents the final classification result obtained by fusion.
The choice of the weighted fusion strategy in this paper is based on the idea of the Matthew effect. For two classification results that need to be fused, if the predicted classification confidence is greater, it is given a greater fusion weight in the fusion process, because the higher is the classification confidence, the more confident it is in its classification results. Thus, the fusion strategy of adaptively assigning weights is adopted here.

Although the two kinds of information have been effectively fused through the proposed weighted fusion strategy, through further thinking, it can be found that not only the final classification result $O_F$ after constrained fusion is needed, but also the constraint encoder and the learning of the classification results of the decoder (i.e., $O_E$ and $O_D$) are needed to promote the learning of the final classification result. The specific process can be expressed as follows:

$$
Loss_1 = BCE(O_E, Y) \tag{19}
$$

$$
Loss_2 = BCE(O_D, Y) \tag{20}
$$

$$
Loss_3 = BCE(O_F, Y) \tag{21}
$$

$$
Loss = Loss_1 + Loss_2 + 10 \times Loss_3 \tag{22}
$$

where $BCE$ [26] represents the binary cross entropy loss, and $Y = \{y_1, \cdots, y_K\}$ is the true label corresponding to the text. Since the most fundamental purpose is to promote the learning of the final classification performance, it is necessary to pay more attention to the final classification result $O_F$ and give it a larger weight of the loss function. Here, this paper directly sets its weight to 10 to ensure the realization of this purpose. Such a simple setup avoids complex hyperparameter tuning and already achieves superior results.

4. Experiments

In this section, we evaluate the proposed model on two standard benchmark datasets to verify the performance.

4.1. Datasets

This paper conducts experimental analysis on two multilabel text classification datasets, Arxiv Academic Paper Dataset (AAPD) [7] and Reuters Corpus Volume I (RCV1-V2) [23]. The AAPD dataset consists of 55,840 abstracts of papers in related disciplines in the field of computer science, each of which can contain multiple subject topics, with a total of 54 topic categories; the RCV1-V2 dataset contains more than 800,000 news articles collected by Reuters, a total of 103 themes. The specific statistics of the dataset are shown in Table 1, and the division of training set, validation set, and test set follows the settings in the literature [7].

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Total</th>
<th>Average Length of Text</th>
<th>Labels</th>
<th>Average Number of Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAPD</td>
<td>55,840</td>
<td>163.42</td>
<td>54</td>
<td>2.41</td>
</tr>
<tr>
<td>RCV1-V2</td>
<td>804,414</td>
<td>123.94</td>
<td>103</td>
<td>3.24</td>
</tr>
</tbody>
</table>

4.2. Experimental Setup

The relevant parameter settings in the experimental process of this paper are shown in Table 2. The AdamW [24] optimizer was used for training, the batch size was set to 32, and the number of epochs was 20. The Bi-LSTM hidden dimension was set to 256, and the transformer dimension was set to 512 for matching. The classic eight-head attention setting was used, the number of stacked layers was 3, and Dropout [25] with a scale of 0.1 was used to prevent overfitting. Details about these parameters can be found in the relevant references.
Table 2. Experimental parameter settings.

<table>
<thead>
<tr>
<th>Experimental Parameter</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Epoch</td>
<td>20</td>
</tr>
<tr>
<td>Batch size</td>
<td>32</td>
</tr>
<tr>
<td>Optimizer</td>
<td>AdamW</td>
</tr>
<tr>
<td>Learning rate</td>
<td>10⁻⁴</td>
</tr>
<tr>
<td>Betas</td>
<td>0.9, 0.999</td>
</tr>
<tr>
<td>Weight decay</td>
<td>10⁻⁴</td>
</tr>
<tr>
<td>Learning rate decay</td>
<td>0.9</td>
</tr>
<tr>
<td><strong>Model parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Word embedding dimension</td>
<td>256</td>
</tr>
<tr>
<td>Hidden layer dimension</td>
<td>256</td>
</tr>
<tr>
<td>Transformer dimension</td>
<td>512</td>
</tr>
<tr>
<td>Number of attention heads</td>
<td>8</td>
</tr>
<tr>
<td>Encoder–decoder layers</td>
<td>3</td>
</tr>
<tr>
<td>FFN internal dimensions</td>
<td>2048</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.1</td>
</tr>
</tbody>
</table>

4.3. Results

In order to comprehensively evaluate the effectiveness of the proposed algorithm, this paper conducts comparative experiments with a variety of multilabel text classification baseline algorithms, including some classical machine-learning-based methods (i.e., BR [14], CC [15], and LP [16]) and deep-learning-based methods (i.e., CNN [3], CNN-RNN [8], SGM [7], Seq2Set [8], LEAM [10], LSAN [11], and ML-R [13]). All the algorithms mentioned were implemented using PyTorch (version 1.2.0, Meta AI, Menlo Park, CA, USA) on a server with a RTX 1650Ti GPU (NVIDIA, Santa Clara, CA, USA) and an Intel Core i7-9750X CPU (Intel, Santa Clara, CA, USA).

The comparative experimental results on the AAPD dataset are shown in Table 3. (−) means that the lower the value, the better the algorithm effect, while (+) means that the higher the value, the better the algorithm effect. The optimal result of each column in the table is shown in bold. It can be clearly seen from Table 3 that the algorithm proposed in this paper obtained better classification results than the other algorithms. The micro-F1 value of our proposed method, JE-FTT, has a relative improvement of 5.0% compared with the sequence generation model, SGM, while it has a relative improvement of 1.7% compared with the newer iterative inference algorithm, ML-R. The hamming losses (HLs) are down, 7.6% and 6.5%, year over year.

Table 3. Comparative experiments on the AAPD dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>HL (−)</th>
<th>Micro-P (+)</th>
<th>Micro-R (+)</th>
<th>Micro-F1 (+)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR</td>
<td>0.0316</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>SGM</td>
<td>0.0251</td>
<td>0.746</td>
<td>0.659</td>
<td>0.699</td>
</tr>
<tr>
<td>Seq2Set</td>
<td>0.0247</td>
<td>0.739</td>
<td>0.674</td>
<td>0.705</td>
</tr>
<tr>
<td>LEAM</td>
<td>0.0261</td>
<td>0.765</td>
<td>0.596</td>
<td>0.670</td>
</tr>
<tr>
<td>LSAN</td>
<td>0.0242</td>
<td>0.777</td>
<td>0.646</td>
<td>0.706</td>
</tr>
<tr>
<td>ML-R</td>
<td>0.0248</td>
<td>0.726</td>
<td>0.718</td>
<td>0.722</td>
</tr>
<tr>
<td>JE-FTT</td>
<td><strong>0.0232</strong></td>
<td><strong>0.755</strong></td>
<td><strong>0.714</strong></td>
<td><strong>0.734</strong></td>
</tr>
</tbody>
</table>

From the micro-P and micro-R indicators, JE-FTT has achieved the second highest precision, lower than that of CNN, and has obtained the second-best recall rate, which is slightly less than that of ML-R. These two methods focus on one aspect due to their special mechanisms. The CNN method learns obvious positive sample features, resulting in high prediction accuracy for simple samples but low recall rate for overall samples; the
inference prediction results in ML-R iteration retrieve more samples, but there is a risk of accumulating errors and reducing accuracy. The algorithm in this paper does not have such obvious emphasis, so it achieves an effective balance between precision and recall and obtains the best micro-F1 results.

In addition, compared with the BERT + SGM algorithm, JE-FTT still obtained better performance without using a super-large-scale pretrained language model, which verifies the effectiveness of the proposed model structure and the weighted fusion strategy supervised by a multiple loss functions. Compared with LSAN based on label attention, the decoder structure based on label query cross attention obtained better results, which fully proves the rationality of the structure designed in this paper and shows that such a design can capture text and labels more effectively.

Table 4 presents the comparative experimental results on the RCV1-V2 dataset. It can be seen that JE-FTT achieved the best micro-F1 result, which is 1.0% higher than the sequence generation model SGM and is 0.8% higher than the newer iterative inference algorithm ML-R. Similar conclusions to those on the AAPD dataset can also be obtained on the precision and recall metrics. However, relatively speaking, the improvement on the RCV1-V2 dataset is not obvious, mainly because the potential correlation between the labels of this dataset is greater, and the method of sequence generation structure can benefit. The algorithm in this paper only implicitly models the label correlation, and the effect on this kind of dataset needs further research to improve in the future.

**Table 4.** Comparative experiments on the RCV1-V2 dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>HL (−)</th>
<th>Micro-P (+)</th>
<th>Micro-R (+)</th>
<th>Micro-F1 (+)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR</td>
<td>0.0086</td>
<td>0.904</td>
<td>0.816</td>
<td>0.858</td>
</tr>
<tr>
<td>CC</td>
<td>0.0087</td>
<td>0.887</td>
<td>0.828</td>
<td>0.857</td>
</tr>
<tr>
<td>LP</td>
<td>0.0087</td>
<td>0.896</td>
<td>0.824</td>
<td>0.858</td>
</tr>
<tr>
<td>CNN</td>
<td>0.0089</td>
<td><strong>0.922</strong></td>
<td>0.798</td>
<td>0.855</td>
</tr>
<tr>
<td>CNN-RNN</td>
<td>0.0085</td>
<td>0.889</td>
<td>0.825</td>
<td>0.856</td>
</tr>
<tr>
<td>SGM</td>
<td>0.0081</td>
<td>0.887</td>
<td>0.850</td>
<td>0.869</td>
</tr>
<tr>
<td>Seq2Set</td>
<td><strong>0.0073</strong></td>
<td>0.900</td>
<td><strong>0.858</strong></td>
<td>0.879</td>
</tr>
<tr>
<td>LEAM</td>
<td>0.0090</td>
<td>0.871</td>
<td>0.841</td>
<td>0.856</td>
</tr>
<tr>
<td>LSAN</td>
<td>0.0075</td>
<td>0.913</td>
<td>0.841</td>
<td>0.875</td>
</tr>
<tr>
<td>ML-R</td>
<td>0.0079</td>
<td>0.890</td>
<td>0.852</td>
<td>0.871</td>
</tr>
<tr>
<td>JE-FTT</td>
<td>0.0079</td>
<td>0.912</td>
<td>0.847</td>
<td><strong>0.878</strong></td>
</tr>
</tbody>
</table>

### 4.4. Comparison and Analysis of Fusion Strategies

In order to further investigate the effectiveness of the weighted fusion strategy supervised by multiple loss functions proposed in this paper, a comparative experiment was conducted on the AAPD dataset for a variety of common fusion strategies, and the results are shown in Figure 3. Among them, ‘adaptive’ means that only the adaptive weight fusion strategy is used, without the use of multiple loss function supervision; ‘learned’ means that the fusion weight is learned through an additional fully connected layer; ‘cat’ and ‘add’ represent splicing followed by a fully connected layer and direct addition, respectively.

It can be seen that compared with the three common fusion strategies, such as learned, cat, and add, the adaptive weight fusion strategy (i.e., adaptive) based on the idea of the Matthew effect achieved better results. Among them, the add strategy had the worst performance, mainly because the two output results depend on information from two different sources, one is the internal information of the text, and the other is the information related to the label and the text. The basic information for each judgment is different, resulting in the judgment result. The difference is relatively large, and it is difficult to achieve a balance by simply adding the average. The cat strategy encountered a similar problem. After the information of different source domains is spliced, even if it is followed by a fully connected layer for conversion, it may lead to greater difficulty in learning and affect the effect. The learnable weight strategy achieves the effect second only to the adaptive strategy, indicating that compared with the source domain feature learning,
it is less difficult to learn the fusion weights separately, and good fusion results can be achieved. However, the fully connected layer used to learn the weights may bring some extra computation, which is not as flexible as the adaptive weighting strategy.

Figure 3. Comparison results of fusion strategies, including (a) HL results and (b) micro-F1 results.

Finally, the multiloss function supervision proposed in this paper further catalyzes the adaptive weight fusion strategy, which simply and effectively obtained the optimal effect with the least cost through the common constraints and weight adjustment of the three outputs.

4.5. Ablation Experiment

To verify the effectiveness of each component in JE-FTT, a series of ablation experiments were performed on the AAPD dataset. The results of using only the encoder output prediction, using only the decoder output prediction, and using only the adaptive weighted fusion under the supervision of the original single loss function were compared in this section. The specific results are shown in Table 5.

Table 5. Ablation experiment results.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>HL (−)</th>
<th>Micro-P (+)</th>
<th>Micro-R (+)</th>
<th>Micro-F1 (+)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoder prediction</td>
<td>0.0236</td>
<td>0.739</td>
<td>0.692</td>
<td>0.724</td>
</tr>
<tr>
<td>Decoder prediction</td>
<td>0.0229</td>
<td>0.764</td>
<td>0.693</td>
<td>0.726</td>
</tr>
<tr>
<td>Adaptive</td>
<td>0.0225</td>
<td>0.785</td>
<td>0.686</td>
<td>0.732</td>
</tr>
</tbody>
</table>

First, from the internal information of the text, it can be found that only using the encoder output for prediction can obtain a relatively competitive result. This shows that the self-attention mechanism and positional encoding in the encoder can effectively model the global relationship within the text and use it for multilabel classification. This finding is consistent with the basic intuition that effectively extracting text internal information is sufficient for basic multilabel text classification.

Second, considering the information between labels and texts, only using the encoder output for prediction can achieve superior classification performance, with micro-F1 reaching 0.724. This is mainly due to the use of the cross-attention mechanism in the design of this paper to learn the dependencies between each type of label query and text representation, which can integrate specific categories and corresponding text information into each type of label query; thus, the final label query is obtained. It has a strong discriminative ability and completes the adaptive fusion of feature-level text and label features.
Then, considering the decision-level fusion results of the above two kinds of information, it is found that in the basic setting of calculating the loss function separately for the final fusion result micro-F1, the adaptive weight fusion strategy is used for the latter two. This information has been effectively fused to a certain extent. The specific performance is improved in all indicators, which fully verifies its effectiveness.

Finally, in order to further tap the fusion potential of the two kinds of information, the multiloss function supervision strategy proposed based on the previous thinking further improves the final classification performance. It is not difficult to find that the strategy only needs to perform two additional loss function values of the output results during the training process, which brings almost negligible training cost; and in the testing phase, there is no additional computational cost at all, which can be said to be a free performance improvement strategy.

5. Conclusions

This paper proposed a novel multilabel text classification algorithm integrating a dual-stream transformer, which effectively solves the problem of insufficient extraction of dependencies between category labels and text. Through the attention mechanism and the designed label query, the internal information of the text and the information between the label and the text are fully extracted. Finally, a weighted fusion strategy supervised by multiple loss functions is proposed to further improve the classification performance, which effectively fuses the two kinds of information at a negligible cost. In order to evaluate the effectiveness of our proposed algorithm, we conducted a set of comparative experiments with a variety of multilabel text classification baseline algorithms on the datasets AAPD and RCV1-V2. The experimental results show that our algorithm has obvious advantages over its competitors, for it achieved the best micro-F1 values on both datasets (i.e., 73.4% on AAPD and 87.8% on RCV1-V2). Although the risk of overfitting is avoided through the learning of implicit label correlation, it also loses the full use of the relevant information between labels. How to mine such information more reasonably in the dataset with rich information between labels will be a future study.

Author Contributions: All authors contributed to the study conception and design. Conceptualization, methodology, investigation, formal analysis, and writing—original draft preparation, L.D., Q.Y. and X.W.; writing—review and editing and supervision, J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the National Key Research and Development Program of China (grant nos.: 2018YFC1603303, 2018YFC1604004) and the National Science Foundation of China (grant no: 61672263).

Conflicts of Interest: The authors declare no conflict of interest.

References


