Co-attention Network with Label Embedding for Text Classification

Minqian Liu[a], Lizhao Liu[b], Junyi Cao[b], Qing Du[b]

[a]*School of Computer Science and Engineering, South China University of Technology, China*
[b]*School of Software Engineering, South China University of Technology, China*

Abstract

Most existing methods for text classification focus on extracting a highly discriminative text representation, which, however, is typically computationally inefficient. To alleviate this issue, label embedding frameworks are proposed to adopt the label-to-text attention that directly uses label information to construct the text representation for more efficient text classification. Although these label embedding methods have achieved promising results, there is still much space for exploring how to use the label information more effectively. In this paper, we seek to exploit the label information by further constructing the text-attended label representation with text-to-label attention. To this end, we propose a Co-attention Network with Label Embedding (CNLE) that jointly encodes the text and labels into their mutually attended representations. In this way, the model is able to attend to the relevant parts of both. Experiments show that our approach achieves competitive results compared with previous state-of-the-art methods on 7 multi-class classification benchmarks and 2 multi-label classification benchmarks.

Keywords: Text-label co-attention, Label embedding, Text classification, Natural language processing, Deep learning

1. Introduction

Text classification is one of the fundamental tasks in Natural Language Processing (NLP), which has been widely applied in sentiment analysis [1], question answering [2], and so on. Given an input text, the goal of text classification is to assign it with one or multiple labels from the predefined label set.

Existing methods for text classification mainly focus on encoding the input text and conducting classification with the encoded text representation. Traditionally, Brown *et al*. [3] and Wallach [4] apply feature engineering techniques and use hand-crafted features to represent the text. In recent years, deep neural networks have been applied to encode the text, including Convolutional Neural Networks (CNNs) [5, 6] and Recurrent Neural Networks (RNNs) [7, 8]. By employing deep models, previous methods are able to extract a more discriminative text representation to improve the performance of text classification. However, these deep models typically consume a large number of computational resources.

*Corresponding author: Qing Du

Email addresses: csmqliu@mail.scut.edu.cn (Minqian Liu), selizhaoliu@mail.scut.edu.cn (Lizhao Liu), sejaycao@mail.scut.edu.cn (Junyi Cao), duqing@scut.edu.cn (Qing Du)

To alleviate this issue, Wang *et al*. [9] and Du *et al*. [10] adopt label embedding frameworks for more efficient text classification. They notice that the impact of label information on learning the text representation is indirect [9]. Hence, they propose to embed all the labels in the predefined label set into the same space as the word embedding and then adopt label-to-text attention to produce a label-attended text representation. By incorporating the label information in a more direct way, these label embedding approaches achieve promising results with fewer parameters and less computation. Thus, a natural question to ask is how to use the information from labels more effectively.

When exploiting the labels, we argue that it is helpful to leverage the information from the text. Intuitively, in a fine-grained sentiment classification task with five classes, humans can first easily exclude some obviously wrong answers (*e.g.*, very negative and negative) by briefly glancing at the text. Then, they can pay more attention to the remaining categories (*e.g.*, neutral, positive, and very positive). Motivated by this, we further incorporate the text-to-label attention into text classification. In this way, we can focus on finding the label(s) that more closely match(es) the text during encoding the label embedding.

In this paper, we propose a text-label co-attention mechanism to obtain the text-attended label representation and the label-attended text representation. Inspired by the co-attention adopted in question answering [11, 12, 13], we introduce a Co-attention Network with Label Embedding (CNLE), where we jointly encode the text and label into their mutually attended representations for more effective text classification. Our model consists of two modules, namely the Text-Label Co-attentive Encoder (TLCE) and the Adaptive Label Decoder (ALD). In particular, the TLCE produces the mutually attended representations to focus on the relevant parts of both, while the ALD leverages these representations to cope with both multi-class classification and multi-label classification without any modification to the model. The extensive experiments on several benchmark datasets demonstrate the effectiveness of the proposed method.

The main contributions of this paper are summarized as follows.

- For more effective text classification, we propose a text-label co-attention mechanism to obtain both the label-attended text representation and text-attended label representation. This allows the model to focus on the relevant parts of both the text and labels to benefit text classification.
- We devise a Co-attention Network with Label Embedding (CNLE) consisting of a Text-Label Co-attentive Encoder (TLCE) and an Adaptive Label Decoder (ALD). The TLCE aims to obtain the mutually attended representations of text and labels, and the ALD leverages these representations to tackle both the multi-class and the multi-label classification without modifying the model.
- We evaluate our method on 7 benchmarks of multi-class classification and 2 benchmarks of multi-label classification. The experiments show that our method achieves competitive results compared with previous state-of-the-art methods.

2. Related Work

Text Classification. Traditional methods for text classification [14, 15, 16] use feature engineering techniques such as N-grams [3] and bag-of-words (BoW) [4] as the feature extractor, and then apply Support Vector Machine (SVM) [16] as the classifier. In recent years, models that are based on neural networks [17, 18, 19, 20, 21, 22], such as Convolutional Neural Networks

(CNNs) [23, 24, 25, 5, 6] and Recurrent Neural Networks (RNNs) [26, 7, 8, 27], have been widely applied in text classification to extract more informative features from the text. Moreover, Yang *et al*. [28] propose a Sequence Generation Model (SGM) to apply a sequence-to-sequence generative approach for the multi-label classification. During the decoding stage, SGM is able to capture the correlations among the labels since it produces the next label given the labels predicted at previous steps. In this work, we also adopt the encoder-decoder architecture to generate the target label(s). In this way, our model is able to cope with both the multi-class classification and multi-label classification without modifying the model. Compared with traditional approaches (*e.g.*, SVM), deep models [8, 5, 7, 20, 21] have achieved substantial improvement in terms of performance thanks to their powerful encoding ability. However, these methods typically rely on a highly discriminative text representation that may require abundant computation resources to obtain. In this work, in addition to exploiting the text representation, we also seek to leverage the label information for more effective and efficient text classification.

Label Embedding. Label embedding for text classification has been studied in multitask learning [29] to tackle the potential loss of label information. Wang *et al*. [9] and Du *et al*. [10] propose to view text classification as a word-label matching problem. Wang *et al*. [9] notice that the use of label information only occurs in learning the classifier on top of the model and its impact on learning the text representation is indirect. To address this issue, they directly incorporate label information from the bottom of the model by introducing the label embedding with a label-to-text attention mechanism. These methods show that the label embedding is informative for the downstream classification task. With a simpler architecture with fewer parameters, they still yield very promising results. In this work, we further incorporate the text-to-label attention and produce the text-label co-attended representation for text classification. Compared with the previous label embedding methods, our method leverages the feedback from the text representation to encode more information into the labels.

Co-attention Mechanism. Co-attention [30, 31, 32] is widely applied in the multi-modal learning between images and language (*e.g.*, visual question answering). Lu *et al*. [30] propose a co-attention model that jointly reasons about the image and question information. Recently, Lu *et al*. [32] incorporate a co-attention transformer architecture to learn the joint representations of the image content and natural language. Moreover, the co-attention has also been applied in question answering [11, 12, 13]. Seo *et al*. [11] adopt context-to-query attention and query-to-context attention to reduce the information loss. Xiong *et al*. [12] adopt the co-attention to attend to important parts in the document and the question. Co-attention [33, 30, 12, 13, 34, 32] in the previous methods typically fuse the information from two separated sources (*e.g.*, image-language or question-text) to produce the attended source representations for each source respectively. Different from the classic co-attention mechanism, we modify the popular self-attention [35] into a co-attention between the source and target (*i.e.*, the text sequence and label sequence) to jointly produce the mutually attended representations for both the text and labels. In this way, our model is able to focus on the relevant parts of both to benefit text classification.

3. Co-attention Network with Label Embedding

We propose a Co-attention Network with Label Embedding (CNLE) to jointly encode the input text sequence and the label sequence, and then use their co-attended representations to generate the target label(s) for text classification. The overall scheme of CNLE is shown in Figure 1.

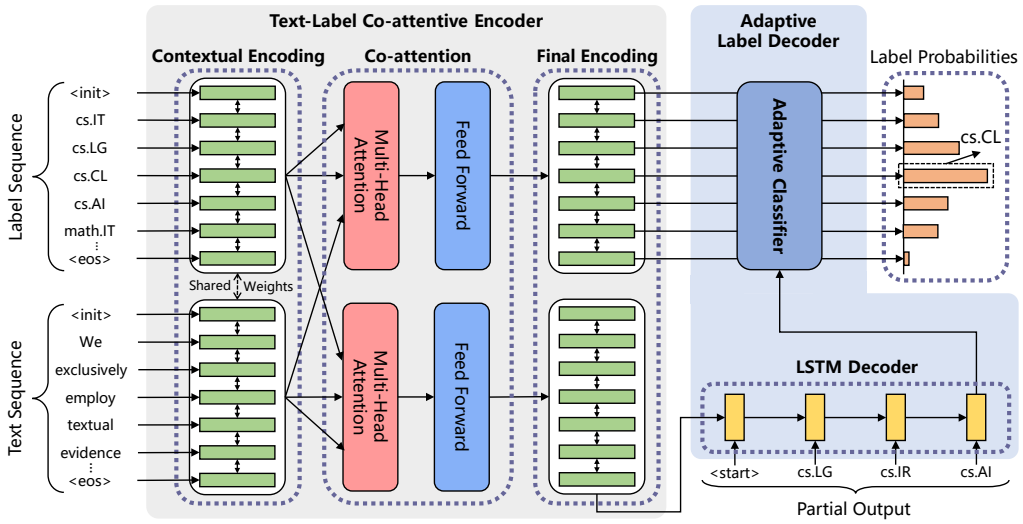

Figure 1: The overall architecture of the Co-attention Network with Label Embedding (CNLE). The illustration shows the encoding process and one timestep of the decoding process on the multi-label classification task. The Text-Label Co-attentive Encoder (TLCE) takes a text sequence and a label sequence as inputs to produce their mutually attended representations. Then, the Adaptive Label Decoder (ALD) uses both the text and label representation to compute the probability for each class in the current timestep. Best viewed in color.

Notations. Throughout the paper, we use the following notations. We denote the text sequence as $\mathbf{x} = [x_1, ..., x_m]$ and the label set as $\mathcal{C} = \{l_1, ..., l_c\}$, where m denotes the number of words in $\mathbf{x}$ and c denotes the number of classes (*e.g.*, for binary classification, $c = 2$). In this work, we consider text classification as a generation problem, where we generate one label at each decoding timestep. Specifically, given the text $\mathbf{x}$, we aim to produce the probabilities $\hat{\mathbf{y}}_t$ for all classes to predict the target label at t-th decoding step. For the multi-class classification, only a single step of decoding is required, while the multi-label classification requires multiple decoding steps.

3.1. General Scheme

In this paper, we focus on incorporating text-to-label attention based on previous label embedding methods [9, 10] to further exploit the information from labels. To this end, we propose a text-label co-attentive architecture to obtain the text-attended label representation and label-attended text representation during the encoding process and then leverage both representations to generate a target label sequence. The intuition behind our proposed co-attention mechanism is that the information of text and labels can be mutually fused with each other. In this way, the model is able to focus on the relevant parts in the text sequence and label sequence.

Formally, we consider a text sequence $\mathbf{x}$ containing m word tokens in the document and a label sequence $\mathbf{l}$ containing c label tokens in the predefined label set $\mathcal{C}$. It is worth noting that the label sequence is converted from the label set $\mathcal{C}$ (see Section 4.1.3 for details) and it is identical for each sample during both training and inference. We aim to incorporate the label information and text information to obtain a label-attended text representation $\mathbf{z}_{x|l}$ and a text-attended label

representation $\mathbf{z}_{l|x}$:

$$\mathbf{z}_{x|l}, \mathbf{z}_{l|x} = f_{enc}(\mathbf{x}, \mathbf{l}), \tag{1}$$

where f_{enc} is any mapping function to be learned. After acquiring these two representations, we generate a sequence of probabilities $\hat{\mathbf{y}}$ for each class with a decoder:

$$\hat{\mathbf{y}} = f_{dec}(\mathbf{z}_{x|l}, \mathbf{z}_{l|x}), \tag{2}$$

where f_{dec} is any mapping function to be learned. In this way, our decoder is able to leverage the mutually attended representations to facilitate the label prediction.

In the following, we will present the architecture design of our model. Particularly, we detail the Text-Label Co-attentive Encoder (TLCE) in Section 3.2 and introduce the Adaptive Label Decoder (ALD) in Section 3.3.

3.2. Text-Label Co-attentive Encoder

In this subsection, we introduce our proposed Text-Label Co-attentive Encoder (TLCE) in detail. The TLCE aims to jointly encode the text sequence and the label sequence into mutually attended text and label representations. Specifically, the TLCE consists of a contextual encoding layer, a co-attention layer, and a final encoding layer.

3.2.1. Contextual Encoding Layer

To encode the word tokens and label tokens into meaningful encodings, we adopt a Bidirectional Long Short Term Memory (BiLSTM) [36] to capture the contextual cues from the text and the correlation (which we consider as a special case of contextual information) among the labels.

Given an input word sequence $\mathbf{x} \in \mathbb{R}^m$ and a label sequence $\mathbf{l} \in \mathbb{R}^c$, we first map them into word embeddings $\mathbf{X}_{emb} \in \mathbb{R}^{m \times d_{emb}}$ and label embeddings $\mathbf{L}_{emb} \in \mathbb{R}^{c \times d_{emb}}$ respectively. The word embeddings are initialized with the pre-trained word vectors [37, 38] while the label embeddings are randomly initialized. For computation efficiency, we apply two independent linear projection layer to project $\mathbf{X}_{emb}$ and $\mathbf{L}_{emb}$ into the more compact embeddings with a smaller dimension, *i.e.*, $\mathbf{X}_{proj} \in \mathbb{R}^{m \times d}$ and $\mathbf{L}_{proj} \in \mathbb{R}^{c \times d}$, respectively. Here, $d < d_{emb}$.

To capture the contextual information in the text sequence and the correlation in the label sequence, we apply a BiLSTM on the projected word embeddings $\mathbf{X}_{proj} \in \mathbb{R}^{m \times d}$ and the projected label embeddings $\mathbf{L}_{proj} \in \mathbb{R}^{c \times d}$ respectively:

$$\begin{aligned} \mathbf{X}_{enc} &= \text{BiLSTM}(\mathbf{X}_{proj}), \\ \mathbf{L}_{enc} &= \text{BiLSTM}(\mathbf{L}_{proj}), \end{aligned} \tag{3}$$

where $\mathbf{X}_{enc} \in \mathbb{R}^{m \times d}$ is the text encoding and $\mathbf{L}_{enc} \in \mathbb{R}^{c \times d}$ is the label encoding. To reduce the number of parameters, we share the weights of this BiLSTM.

3.2.2. Co-attention Layer

To acquire the text-attended label representation and the label-attended text representation, we modify the widely applied self-attention module in the Transformer [35] into a co-attention layer. Instead of applying the self-attention, we employ a co-attention between the text encoding and the label encoding to attend to each other. Intuitively, the label-attended text representation can help the model to focus more on the relevant words for the classification task, while the text-attended label representation is able to emphasize the labels that match the text better.

For convenience, we first recap the scaled dot product attention and multi-head attention in Transformer [35]. The scaled dot product attention is defined as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_h}})\mathbf{V}, \tag{4}$$

where $\mathbf{Q} \in \mathbb{R}^{q \times d_k}$, $\mathbf{K} \in \mathbb{R}^{k \times d_k}$, and $\mathbf{V} \in \mathbb{R}^{k \times d_v}$. And the multi-head attention is defined as:

$$\begin{aligned} \text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \text{Softmax}([\mathbf{H}_1; ...; \mathbf{H}_p])\mathbf{W}^O, \\ \text{where } \mathbf{H}_i &= \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V), \end{aligned} \tag{5}$$

and the projecting parameters are $\mathbf{W}_i^Q \in \mathbb{R}^{d_k \times d_p}$, $\mathbf{W}_i^K \in \mathbb{R}^{d_k \times d_p}$, $\mathbf{W}_i^V \in \mathbb{R}^{d_v \times d_p}$, and $\mathbf{W}_i^O \in \mathbb{R}^{pd_p \times d_h}$. We use $d_k = d_v = d_p$ and $d_p = d_h/p$ is the dimension for each head, where d is the interval model dimension and p is the number of heads. We use the $[\cdot]$ to denote the concatenate operation throughout the paper.

In the multi-head self-attention, the matrices $\mathbf{Q}, \mathbf{K}$, and $\mathbf{V}$ are identical. To mutually refine the text and label, we convert the self-attention into a co-attention by feeding both the text encoding $\mathbf{X}_{enc}$ and the label encoding $\mathbf{L}_{enc}$ into the multi-head attention module:

$$\begin{aligned} \mathbf{X}_{att} &= \text{MultiHead}_X(\mathbf{X}_{enc}, \mathbf{L}_{enc}, \mathbf{L}_{enc}), \\ \mathbf{L}_{att} &= \text{MultiHead}_L(\mathbf{L}_{enc}, \mathbf{X}_{enc}, \mathbf{X}_{enc}), \end{aligned} \tag{6}$$

where $\mathbf{X}_{att} \in \mathbb{R}^{m \times d_h}$ and $\mathbf{L}_{att} \in \mathbb{R}^{c \times d_h}$ are the label-attended text representation and the text-attended label representation respectively.

Similar to the Transformer [35] architecture, we also apply residual connections and two independent Feed Forward Networks (FFN) following Layer Normalization (LN) [39] to obtain their fused encodings $\mathbf{X}_{fuse} \in \mathbb{R}^{m \times d}$ and $\mathbf{L}_{fuse} \in \mathbb{R}^{c \times d}$:

$$\begin{aligned} \mathbf{X}_{fuse} &= \text{LN}_X(\text{FFN}_X(\mathbf{X}_{att}) + \mathbf{X}_{enc}), \\ \mathbf{L}_{fuse} &= \text{LN}_L(\text{FFN}_L(\mathbf{L}_{att}) + \mathbf{L}_{enc}). \end{aligned} \tag{7}$$

The FFN projects the input into dimension d and further propagates the mutually attended information, while the residual connection fuses the attended representations with their original encodings.

3.2.3. Final Encoding Layer

To further leverage the contextual information of the label-attended text encoding and the correlation of the text-attended label encoding, we apply a final encoding layer on top of the co-attention layer.

In the final encoding process, two independent BiLSTMs are applied to propagate the mutually attended information in the fused text encoding and the fused label encoding, respectively. One BiLSTM encodes the fused text encoding $\mathbf{X}_{fuse}$ into the text final representation $\mathbf{X}_{fin}$:

$$\mathbf{X}_{fin} = \text{BiLSTM}_X(\mathbf{X}_{fuse}), \tag{8}$$

where $\mathbf{X}_{fin} \in \mathbb{R}^{m \times d}$. Note that the hidden state $\mathbf{h} \in \mathbb{R}^{d \times 1}$ and the cell state $\mathbf{c} \in \mathbb{R}^{d \times 1}$ inside the BiLSTM_X are preserved for the subsequent decoding process. They are used to initialize the hidden state and cell state in LSTM Decoder respectively (see Section 3.3.1 for details).

Another BiLSTM encodes the fused label encoding to yield the final representation of the label sequence $\mathbf{L}_{fin} \in \mathbb{R}^{c \times d}$:

$$\mathbf{L}_{fin} = \text{BiLSTM}_L(\mathbf{L}_{fuse}). \tag{9}$$

3.3. Adaptive Label Decoder

In this subsection, we elaborate on our Adaptive Label Decoder (ALD) in detail. The ALD leverages both the text representation and the label representation to generate the label(s). For each timestep, the decoding process of ALD is separated into two steps: 1) obtain the hidden state, the cell state, and the recurrent context state with an LSTM Decoder; 2) compute a probability for each class via an adaptive classifier.

3.3.1. LSTM Decoder

To cope with both the multi-class and the multi-label classification without any modification to the model, we adopt an LSTM Decoder to generate the label(s), which is able to further model the correlations between labels [28].

Specifically, we adopt a standard LSTMCell with attention [40] to implement the LSTM Decoder. During training, we first look up the label embedding $\mathbf{e}_{t-1} \in \mathbb{R}^{d\times 1}$ for the ground true label in the $(t-1)$-th decoding step. Note that during inference, we look up the embedding for the predicted label instead. Then, the LSTMCell takes the label embedding $\mathbf{e}_{t-1}$, the recurrent context state $\mathbf{r}_{t-1} \in \mathbb{R}^{d\times 1}$, the hidden state $\mathbf{h}_{t-1} \in \mathbb{R}^{d\times 1}$, and the cell state $\mathbf{c}_{t-1} \in \mathbb{R}^{d\times 1}$ of previous timestep as inputs. It outputs the hidden state $\mathbf{h}_t$ and the cell state $\mathbf{c}_t$ for the current timestep t as follows:

$$\mathbf{h}_t, \mathbf{c}_t = \text{LSTMCell}([\mathbf{e}_{t-1}; \mathbf{r}_{t-1}], \mathbf{h}_{t-1}, \mathbf{c}_{t-1}). \tag{10}$$

Here, we initialize the $\mathbf{h}_0$ and $\mathbf{c}_0$ with $\mathbf{h}$ and $\mathbf{c}$ from the encoding process respectively. Both the $\mathbf{e}_0$ and $\mathbf{r}_0$ are initialized with zero vectors.

After we obtain the hidden state $\mathbf{h}_t$, we compute the attention weights $\mathbf{a}_t \in \mathbb{R}^{m\times 1}$ to help the decoder to focus on the text information relevant to the timestep t:

$$\mathbf{a}_t = \text{Softmax}(\mathbf{X}_{fin}\mathbf{W}_1\mathbf{h}_t), \tag{11}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d\times d}$ is a trainable matrix. Then, we obtain the recurrent context state $\mathbf{r}_t$ for timestep t as follows:

$$\mathbf{r}_t = \text{Tanh}(\mathbf{W}_2[\mathbf{X}_{fin}^T\mathbf{a}_t; \mathbf{h}_t]), \tag{12}$$

where $\mathbf{W}_2 \in \mathbb{R}^{d\times 2d}$ is a trainable matrix. The recurrent context state $\mathbf{r}_t$ will be used in the next timestep to compute the hidden state $\mathbf{h}_{t+1}$ as in Equation 10.

Then, the hidden state $\mathbf{h}_t$ will be passed to the adaptive classifier for subsequent procedures.

3.3.2. Adaptive Classifier

To leverage the informative text-attended label representation, we propose an adaptive classifier. Comparing with the linear projection layer used in most of the existing methods [5, 26, 41, 27, 42], the adaptive classifier is able to attentively leverage the text-attended label representation to directly output a probability for each class.

Given the hidden state $\mathbf{h}_t$ of timestep t, the adaptive classifier takes the final label encoding $\mathbf{L}_{fin}$ and the hidden state $\mathbf{h}_t$ as inputs and produce the probabilities $\hat{\mathbf{y}}_t$ at timestep t:

$$\hat{\mathbf{y}}_t = \text{Sotfmax}(\mathbf{L}_{fin}\mathbf{W}_3\mathbf{h}_t), \tag{13}$$

where $\mathbf{W}_3 \in \mathbb{R}^{d\times d}$ is a trainable matrix and $\hat{\mathbf{y}}_t \in \mathbb{R}^{c\times 1}$ is a vector that contains the probability for each class at timestep t. In this way, we directly incorporate the text-attended label representation to facilitate the classification process.

Table 1: Statistics summary of multi-class and multi-label classification datasets. #Training and #Test refer to the number of samples in the training set and test set respectively.

Type	Dataset	#Classes	#Training	#Test	Task
Multi-class	Yelp Full	5	650k	50k	sentiment
	Yelp Polarity	2	560k	38k	sentiment
	Amazon Full	5	3,000k	650k	sentiment
	Amazon Polarity	2	3,600k	400k	sentiment
	AG News	4	120k	76k	topic
	DBPedia	14	560k	70k	ontology
	Yahoo! Answers	10	1,400k	60k	topic
Multi-label	AAPD	54	53,840	1,000	topic
	Reuters-21578	90	9,598	3,299	topic

After generating the probabilities for all the T timesteps, we compute the objective function $\mathcal{L}$ to optimize our model as follows:

$$\mathcal{L} = -\sum_{t=1}^{T} \sum_{i \in \mathcal{C}} \mathbb{1}\left[\mathbf{y}_t = i\right] \log(\hat{\mathbf{y}}_t), \tag{14}$$

where $\mathbf{y}_t$ is the ground true label at timestep t. $\mathbb{1}[\cdot]$ is the indicator function, where $\mathbb{1}[A] = 1$ if A is true and $\mathbb{1}[A] = 0$ if A is false. Note that Equation 14 can be applied in both the multi-label classification and multi-class classification task.

4. Experiments

4.1. Experimental Settings

4.1.1. Datasets

We evaluate the models on 7 multi-class and 2 multi-label datasets. The statistics summary of these datasets is presented in Table 1.

For multi-class classification, we follow [24] and use 7 standard benchmark datasets with contents specified as follows. Note that we consider the binary classification as a special case of the multi-class classification.

- Yelp Full Review (Yelp F.): The Yelp Review dataset is obtained from the Yelp Dataset Challenge in 2015. This task aims to predict the sentiment polarity labels, *i.e.*, very negative, negative, neutral, positive, and very positive.
- Yelp Polarity Review (Yelp P.): This dataset contains the same set of the review texts in the Yelp Full Review, but only considers two sentiment polarities, *i.e.*, negative and positive.
- Amazon Full Review (Amz. F.): The Amazon dataset is obtained from the Stanford Network Analysis Project (SNAP) [43]. Similar to the Yelp Review dataset, the Amazon Full Review dataset also has five sentiment stars ranging from 1 to 5.
- Amazon Polarity Review (Amz. P.): Similar to the Yelp Polarity Review, this dataset also considers the star 1 and 2 as negative, and the star 4 and 5 as positive.

- AG News (AG): The AG News dataset is obtained from the Internet news articles [44]. It has four categories for topic classification: world, entertainment, sports, and business.
- DBPedia (DBP.): The DBPedia is constructed from Wikipedia [45]. It has 14 non-overlapping ontology classes from DBpedia 2014.
- Yahoo! Answers (Yah. A.): The Yahoo! Answers dataset is constructed from Yahoo! Answers Comprehensive Questions and Answers version 1.0 dataset in the Yahoo! Webscope program.

For multi-label classification, we evaluate our method on 2 popular datasets, namely AAPD [28] and Reuters-21578 [46].

- Arxiv Academic Paper Dataset (AAPD): The AAPD dataset is built by collecting the abstract and the corresponding subjects of 55,840 papers from the computer science academic website.[1]
- Reuters-21578: Reuters is a dataset collected from Reuters news articles with 90 categories.

4.1.2. Evaluation Metrics

We use accuracy as the metric for multi-class classification. For multi-label classification, we adopt the micro-averaged F1 score [47], which comes from the class-weighted harmonic mean of precision and recall. Denote the true positive, false positive, and false negative counts of the i-th class as TP_i, FP_i, and FN_i respectively. The micro-F1 score is computed as follows:

$$\text{micro-F}_1 = \frac{\sum_{i=1}^{c} 2TP_i}{\sum_{i=1}^{c} 2TP_i + FP_i + FN_i}. \tag{15}$$

4.1.3. Implementation Details

To convert the predefined label set into our input label sequence, we randomly choose the order of labels (which is fixed once chosen) and concatenate all the labels to obtain the label sequence. The label sequence contains all the labels in the label set and it is identical for every sample in the dataset. For word embeddings in the text sequence, we concatenate the 300-dimension Glove [37] and the 100-dimension character n-gram embedding [38] to produce 400-dimension embeddings, *i.e.*, $d_{emb} = 400$, which are frozen during training. We initialize Out-Of-Vocabulary (OOV) words and label embeddings from a uniform distribution with the range $[-0.01, 0.01]$ (also with 400 dimensions). The internal model dimension d is set to 256. For the BiLSTM, we set the output dimension to $d/2$ and concatenate the outputs of two directions to obtain the final outputs. The hidden dimension in the co-attention layer, *i.e.*, d_h is set to 150 and the number of attention heads p is set to 3. We use a greedy search decoding strategy for the decoder since labels are usually short in length. Our model is trained with an Adam [48] optimizer along with the warmup schedule as in [35]. β_1, β_2 and ϵ in the Adam optimizer are set to 0.9, 0.98 and 10^{-9} respectively.

[1] https://arxiv.org/

Table 2: Test accuracy on 7 multi-class benchmarks. The best results are in bold.

Model	Yelp P.	Yelp F.	Amz. P.	Amz. F.	AG	DBP.	Yah. A.
Bags-of-words [24]	92.24	57.99	90.40	54.64	88.81	96.61	68.89
LSTM [24]	94.74	58.17	93.90	59.43	86.06	98.55	70.84
VDCNN [5]	95.72	64.26	95.69	63.00	91.27	98.71	73.43
D. LSTM [27]	92.60	59.60	-	-	92.10	98.70	73.70
Bi-BloSAN [18]	94.56	62.13	-	-	92.45	98.77	76.28
LEAM (Linear) [9]	93.43	61.03	-	-	91.75	98.32	75.22
LEAM [9]	95.31	64.09	-	-	92.45	99.02	**77.42**
EXAM [10]	-	-	95.50	61.90	93.00	99.00	74.80
E1-E2 [42]	96.70	67.00	96.00	63.10	93.20	99.00	75.00
CNLE (Ours)	**97.13**	**68.15**	**96.23**	**64.18**	**94.00**	**99.17**	75.78

Table 3: Test micro-F1 on 2 multi-label benchmarks. The best results are in bold.

Model	AAPD	Reuters
SVM [41]	69.1	86.1
TextCNN [23]	51.4	80.8
HAN [8]	68.0	85.2
XML-CNN [6]	68.7	86.2
SGM [28]	71.0	78.8
LSTM$_{base}$ [41]	69.6	84.9
LSTM$_{reg}$ [41]	70.5	87.0
CNLE (Ours)	**71.7**	**89.9**

4.2. Quantitative Results

In this part, we compare the performance of different methods on 9 benchmarks in total. We present the results of multi-class classification in Table 2 and the results of multi-label classification in Table 3. Due to space constraints, we use abbreviations for the names of the datasets that are introduced in Section 4.1 in Table 2.

Multi-class Classification. As shown in Table 2, our CNLE generally achieves higher accuracy compared with the previous baseline methods on the 6 multi-class datasets, which validates the effectiveness of our method. Label embedding methods (*i.e.*, LEAM [9], EXAM [10]) achieve promising performance compared with the methods with more complex architecture, such as VDCNN [5] and Encoder1-Encoder2 [42] (abbreviated as E1-E2 in Table 2). Different from them, our model uses both the text-attended label presentation and label-attended text presentation. The results show that our model achieves considerable improvement in performance comparing with previous state-of-the-art methods. This agrees well with our intuition that further exploiting the information from labels improves the text classification task more effectively.

Multi-label Classification. We also compare our model with previous state-of-the-arts in multi-label classification. From Table 3, our model outperforms the prior works, often by a large margin. Significantly, our model surpasses previous methods by 2.9% on the Reuters dataset. The improvement indicates that our model is able to better cope with multi-label classification.

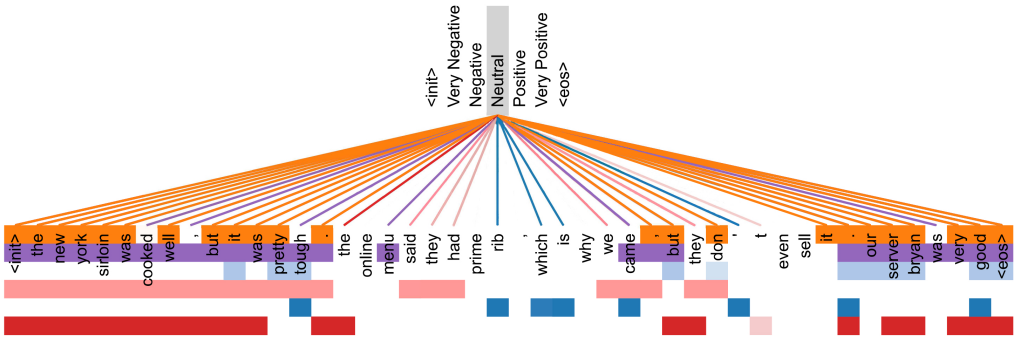

Figure 2: Visualization of the attention scores in the multi-head co-attention layer. We visualize them from a sample with label *Neutral* in the test set of the Yelp Full Review dataset. The top three rows show the label-to-text multi-head attention scores, while the three rows at the bottom show the text-to-label attention scores. Best viewed in color.

Table 4: Ablation study of CNLE on 4 multi-class benchmarks. Best results are in bold.

Model	Yelp P.	Yelp F.	AG	DBP.
CNLE w/o Label Embedding	96.96	68.07	93.74	99.15
CNLE w/o Text-to-label Attention	96.66	68.10	93.93	**99.17**
CNLE	**97.13**	**68.15**	**94.00**	**99.17**

4.3. Qualitative Results

We visualize both the label-to-text attention and the text-to-label attention in the co-attention layer in Figure 2. As depicted in the figure, the important parts for classification, such as *the new york sirloin cooked well, but it was pretty tough*, have higher attention scores on both sides. It shows that our co-attention layer captures the classification-relevant parts in both the text sequence and the label sequence, which verifies the effectiveness of the proposed text-label co-attention mechanism.

5. Further Experiments

5.1. Ablation Studies

To further validate the effectiveness of our CNLE, we conduct two ablation studies on both multi-class and multi-label benchmarks. The results are shown in Table 4 and Table 5 respectively. In the first ablation study (CNLE w/o Label Embedding), we do not use the label embedding (*i.e.*, label sequence) at all. In another word, only the text sequence is fed into the model as input, which yields a model similar to most existing methods. In the second ablation (CNLE w/o Text-to-label Attention), we retain the label embedding and label-to-text attention. However, we remove the text-to-label attention such that the label embedding is not attended by the text. Ablation results indicate that with the text-attended label representation, our method generally yields a considerable improvement in terms of performance. This shows the effectiveness of our text-label co-attentive architecture.

Table 5: Ablation study of CNLE on 2 multi-label benchmarks. Best results are in bold.

Model	AAPD	Reuters
CNLE w/o Label Embedding	71.6	88.0
CNLE w/o Text-to-label Attention	70.6	89.4
CNLE	**71.7**	**89.9**

Table 6: Comparison with large pre-trained models in terms of the test accuracy, number of parameters and the average performance gain (Δ). Best results are in bold.

Model	Yelp P.	Yelp F.	Amz. P.	Amz. F.	DBP.	Params.(M)	Δ (%)
CNLE (Ours)	97.13	68.15	96.23	64.18	99.17	3	-
BERT [20]	98.11	70.68	97.37	65.83	99.36	340	0.015
XLNet [21]	**98.45**	**72.20**	**97.60**	**67.74**	**99.38**	340	**0.025**

5.2. Comparison with Large Pre-trained Models.

Recently, several large pre-trained models such as BERT [20] and XLNet [21] use large scale unsupervised corpus and obtain a powerful language representation ability. These models indeed achieve better performance than ours, but we argue that they are computationally inefficient and might not be practical in some real-life settings. In Table 6, we present the comparison between our CNLE and two large pre-trained models in terms of performance and the number of parameters. We clearly see that they have more than 100 times parameters comparing with CNLE, but only with less than 0.03% performance gain. This is a big disadvantage in the real-life scenarios with strict time and space constraints.

6. Conclusion

In this paper, we introduce a text-label co-attention mechanism for more effective text classification. We seek to obtain the mutually attended representations for both the text sequence and the label sequence. In this way, we are able to focus on the relevant parts of the text and labels to facilitate text classification. Moreover, with the generative decoder, we are able to cope with both multi-class classification and multi-label classification without modifying the model. The extensive experiments on 7 multi-class classification benchmarks and 2 multi-label benchmarks demonstrate the superiority of our method over the considered baseline methods. In future work, the idea of the co-attention mechanism can be further applied to other NLP tasks, such as few-shot text classification and machine reading comprehension.

References

[1] T. Sahni, C. Chandak, N. R. Chedeti, M. Singh, Efficient twitter sentiment classification using subjective distant supervision, in: 2017 9th International Conference on Communication Systems and Networks (COMSNETS), IEEE, 2017, pp. 548–553.

[2] D. Zhang, W. S. Lee, Question classification using support vector machines, in: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, 2003, pp. 26–32.

[3] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, J. C. Lai, Class-based n-gram models of natural language, Computational linguistics 18 (4) (1992) 467–479.

[4] H. M. Wallach, Topic modeling: beyond bag-of-words, in: Proceedings of the 23rd international conference on Machine learning, 2006, pp. 977–984.
[5] A. Conneau, H. Schwenk, L. Barrault, Y. Lecun, Very deep convolutional networks for text classification, in: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 1107–1116.
[6] J. Liu, W.-C. Chang, Y. Wu, Y. Yang, Deep learning for extreme multi-label text classification, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 115–124.
[7] P. Liu, X. Qiu, X. Huang, Recurrent neural network for text classification with multi-task learning, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016, pp. 2873–2879.
[8] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical attention networks for document classification, in: Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies, 2016, pp. 1480–1489.
[9] G. Wang, C. Li, W. Wang, Y. Zhang, D. Shen, X. Zhang, R. Henao, L. Carin, Joint embedding of words and labels for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, pp. 2321–2331.
[10] C. Du, Z. Chen, F. Feng, L. Zhu, T. Gan, L. Nie, Explicit interaction model towards text classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 6359–6366.
[11] M. Seo, A. Kembhavi, A. Farhadi, H. Hajishirzi, Bidirectional attention flow for machine comprehension, in: International Conference on Learning Representations, 2017.
[12] C. Xiong, V. Zhong, R. Socher, Dynamic coattention networks for question answering, in: International Conference on Learning Representations, 2017.
[13] B. McCann, N. S. Keskar, C. Xiong, R. Socher, The natural language decathlon: Multitask learning as question answering, arXiv preprint arXiv:1806.08730.
[14] A. McCallum, K. Nigam, et al., A comparison of event models for naive bayes text classification, in: AAAI-98 workshop on learning for text categorization, Vol. 752, Citeseer, 1998, pp. 41–48.
[15] H.-J. Kim, J.-U. Kim, Y.-G. Ra, Boosting naı¨ ve bayes text classification using uncertainty-based selective sampling, Neurocomputing 67 (2005) 403–410.
[16] S. R. Gunn, et al., Support vector machines for classification and regression (1998).
[17] S. Zhou, Q. Chen, X. Wang, Active deep learning method for semi-supervised sentiment classification, Neurocomputing 120 (2013) 536–546.
[18] T. Shen, T. Zhou, G. Long, J. Jiang, C. Zhang, Bi-directional block self-attention for fast and memory-efficient sequence modeling, in: International Conference on Learning Representations, 2018.
[19] G. Liu, J. Guo, Bidirectional lstm with attention mechanism and convolutional layer for text classification, Neurocomputing 337 (2019) 325–338.
[20] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.
[21] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, in: Advances in neural information processing systems, 2019, pp. 5754–5764.
[22] J. Kim, S. Jang, E. Park, S. Choi, Text classification using capsules, Neurocomputing 376 (2020) 214–221.
[23] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1746–1751.
[24] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, in: Advances in neural information processing systems, 2015, pp. 649–657.
[25] P. Wang, B. Xu, J. Xu, G. Tian, C.-L. Liu, H. Hao, Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification, Neurocomputing 174 (2016) 806–814.
[26] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.
[27] D. Yogatama, C. Dyer, W. Ling, P. Blunsom, Generative and discriminative text classification with recurrent neural networks, arXiv preprint arXiv:1703.01898.
[28] P. Yang, X. Sun, W. Li, S. Ma, W. Wu, H. Wang, Sgm: sequence generation model for multi-label classification, in: Proceedings of the 27th International Conference on Computational Linguistics, 2018, pp. 3915–3926.
[29] H. Zhang, L. Xiao, W. Chen, Y. Wang, Y. Jin, Multi-task label embedding for text classification, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018, pp. 4545–4553.
[30] J. Lu, J. Yang, D. Batra, D. Parikh, Hierarchical question-image co-attention for visual question answering, in: Advances in neural information processing systems, 2016, pp. 289–297.
[31] Z. Yu, J. Yu, J. Fan, D. Tao, Multi-modal factorized bilinear pooling with co-attention learning for visual question answering, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 1821–1830.

[32] J. Lu, D. Batra, D. Parikh, S. Lee, Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks, in: Advances in Neural Information Processing Systems, 2019, pp. 13–23.
[33] Y. Li, T. Liu, J. Hu, J. Jiang, Topical co-attention networks for hashtag recommendation on microblogs, Neurocomputing 331 (2019) 356–365.
[34] L. Zhang, Z. Guan, A. Hauptmann, The co-attention model for tiny activity analysis, Neurocomputing 105 (2013) 51–60.
[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.
[36] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional lstm and other neural network architectures, Neural networks 18 (5-6) (2005) 602–610.
[37] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.
[38] K. Hashimoto, C. Xiong, Y. Tsuruoka, R. Socher, A joint many-task model: Growing a neural network for multiple nlp tasks, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2017, pp. 1923–1933.
[39] J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450.
[40] T. Sahni, C. Chandak, N. R. Chedeti, M. Singh, Efficient twitter sentiment classification using subjective distant supervision, in: 2017 9th International Conference on Communication Systems and Networks (COMSNETS), IEEE, 2017, pp. 548–553.
[41] A. Adhikari, A. Ram, R. Tang, J. Lin, Rethinking complex neural network architectures for document classification, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4046–4051.
[42] G. Niu, H. Xu, B. He, X. Xiao, H. Wu, G. Sheng, Enhancing local feature extraction with global representation for neural text classification, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 496–506.
[43] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: Proceedings of the 7th ACM conference on Recommender systems, 2013, pp. 165–172.
[44] G. M. Del Corso, A. Gulli, F. Romani, Ranking a stream of news, in: Proceedings of the 14th international conference on World Wide Web, 2005, pp. 97–106.
[45] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al., Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia, Semantic Web 6 (2) (2015) 167–195.
[46] C. Apté, F. Damerau, S. M. Weiss, Automated learning of decision rules for text categorization, ACM Transactions on Information Systems (TOIS) 12 (3) (1994) 233–251.
[47] C. D. Manning, P. Raghavan, H. Schütze, Introduction to information retrieval, Cambridge university press, 2008.
[48] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: International Conference on Learning Representations, 2015.