CRAN: a Hybrid CNN-RNN Attention-based Model for Text Classification

Long Guo[1], Dongxiang Zhang[2], Lei Wang[2], Han Wang[2], and Bin Cui[1]

[1] Peking University , Beijing, China
{guolong, bin.cui}@pku.edu.cn
[2] University of Electronic Science and Technology of China, Chengdu, China
{zhangdo,201621060133,2015200101009}@.uestc.edu.cn

Abstract. Text classification is one of the fundamental tasks in the field of natural language processing. The CNN-based approaches and RNN-based approaches have shown different capabilities in representing a piece of text. In this paper, we propose a hybrid CNN-RNN attention-based neural network, named CRAN, which combines the convolutional neural network and recurrent neural network effectively with the help of the attention mechanism. We validate the proposed model on several large-scale datasets (i.e., eight multi-class text classification and five multi-label text classification tasks), and compare with the state-of-the-art models. Experimental results show that CRAN can achieve the state-of-the-art performance on most of the datasets. In particular, CRAN yields better performance with much fewer parameters compared with a very deep convolutional networks with 29 layers, which proves its effectiveness and efficiency.

Keywords: Text Classification · Convolutional neural network · Recurrent neural network · Attention mechanism.

1 Introduction

Text classification is one of the fundamental tasks in the field of natural language processing where one needs to assign a single or multiple predefined labels to a sequence of text. It has attracted significant attention from both academic and industry communities due to its broad applications, such as topic modeling [21], sentiment classification [16], and spam detection [2]. Traditional approaches of text classification generally consist of a feature extraction stage, where some sparse lexical features, such as n-grams, are extracted to represent a document, followed by a classification stage, where the features are sent to a linear or kernel classifier. More recent approaches start using deep neural networks, such as convolutional neural networks [12, 11] and recurrent neural networks based on long short-term memory (LSTM) [9] or gated recurrent unit (GRU) [5], to jointly perform feature extraction and classification for document classification.

Although neural-network-based approaches to text classification have been proved quite effective, the CNN-based and the RNN-based approaches have shown different capabilities in representing a piece of text. On the one hand, the CNN-based approaches utilize layers with convolutional filters that are applied to local features, and attempt to extract effective text representation by identifying the most influential n-grams of

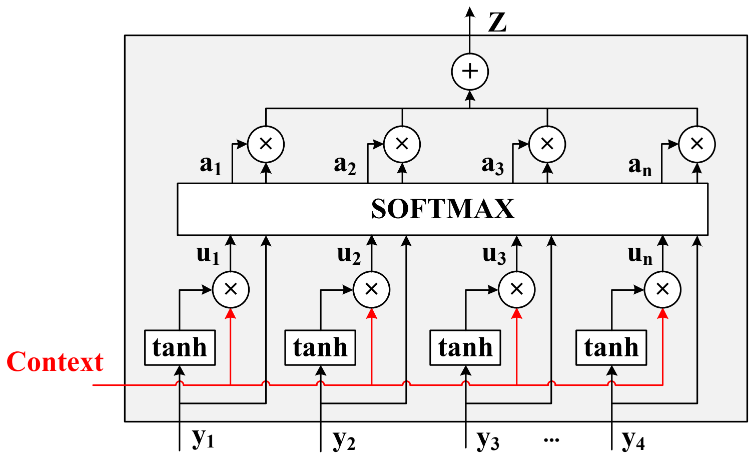

Fig. 1: Attention Model

different semantic aspects. However, these methods can only learn local patterns, and it is challenging to deal with the order-sensitive and long distance dependency patterns contained in the sentences. On the other hand, the RNN-based approaches are particularly good at modeling sequential data, and capable of building effective text representation by learning temporal features and long-term dependencies between nominal pairs. However, these methods treat each word in the sentences equally and thus cannot distinguish between the key words that contribute more to the classification and the common words. While the convolutional neural networks and the recurrent neural networks can complement each other for text classification, existing methods [20, 22] make a straightforward attempt to combine CNN and RNN with a serial structure by directly applying RNN on top of CNN. This kind of serial structure would suffer from some feature loss since the top module cannot be applied directly on the original text. To exploit the full advantages of these two kinds of models, new effective combination method needs to be designed.

Recently, attention-based neural networks have demonstrated great success in a wide range of tasks ranging from image captioning, speech recognition and question answering to machine translations [23, 6, 8, 1, 19]. In terms of natural language classification, Att-BLSTM [28] and HAN [24] can achieve the state-of-the-art performance for relation classification and multiple sentences classification, respectively. Both models apply an attention model on top of a bidirectional RNN model. The basic architecture of the attention model is shown in Figure 1, which takes n arguments y_1, ...,y_n and a context and returns a vector Z which is supposed to be the summary of the input y. The intuition underlying the attention mechanism is that not all parts of a document are equally relevant for answering a query. Therefore, the vector Z is a weighed arithmetic mean of the input y, where the weights are chosen according to the relevance of each y_i given the context. However, in Att-BLSTM and HAN, the context is initialized randomly, where the role of the context is weakened significantly.

Inspired by the role of the context in the attention model, we propose a hybrid parallel CNN-RNN framework, named CNN-RNN attention-based neural network (CRAN), to model the local patterns and long-distance dependency patterns in the text simultaneously. In our model, CNN and RNN are first applied to the original text in parallel. Then, the attention model takes the output of the RNN as the input and the output of the CNN as the context. Therefore, the returned vector Z can pick the useful local features from the sequences generated by the RNN according to the context generated by the CNN.

The key difference to previous work on attention mechanism is that our model uses the output of the CNN as the $context$ to discover when a sequence of token is relevant rather than simply ignoring the context or initializing the context randomly. Experimental results show that CRAN can achieve the state-of-the-art performance on most of the datasets, which demonstrates that the attention mechanism is an effective framework to unify CNN and RNN, and the CNN layer serves as a meaningful context for the attention mechanism. In addition, CRAN yields better performance with much fewer parameters compared with a very deep convolutional networks (VD-CNN) with 29 layers, which proves its effectiveness and efficiency.

2 Model

In this section we describe our proposed CRAN model in detail. As shown in Figure 2, the model proposed in this paper contains the following components.

2.1 Input Layer

In our model, word embedding is used to map words from a vocabulary to a corresponding vector of real values. In more detail, given a document consisting of n words $T = \{w_1, w_2, ..., w_n\}$, every word w_i is converted into a real-valued vector e_i. For each word in T, we first look up the embedding matrix $\mathbf{W} \in \mathbb{R}^{d\times|V|}$, where $|V|$ is a fixed-sized vocabulary and d is the dimension of the word embedding vector. We transform a word w_i into its word embedding e_i by using the matrix-vector product:

$$e_i = \mathbf{W}v_i, \tag{1}$$

where v_i is a vector of size $|V|$ with the element corresponding to e_i set to 1 and other elements set to 0. In this way, the document can be represented as real-valued vectors $\mathsf{e} = \{e_1, e_2, ..., e_n\}$ and fed into the next layer, i.e., the CNN layer and RNN layer.

2.2 CNN Layer

The CNN layer is used to extract the most influential n-grams of different semantic aspects from the text. CNN performs a discrete convolution on the input embedding matrix with a set of different filters. In this paper, we adopt a CNN structure similar with the model proposed in [12]. Th convolutional layer consists of two stages. The first stage is called "convolution", where a set of k filters are applied to the input sequence e. Each filter $\mathbf{F} \in \mathbb{R}^{h\times d}$ with size h is applied to a window of h embedding words $\mathsf{e}_{i:i+h-1}$ to produce a new feature c_i as follows:

$$c_i = f(\mathbf{F}\mathsf{e}_{i:i+h-1} + b), \tag{2}$$

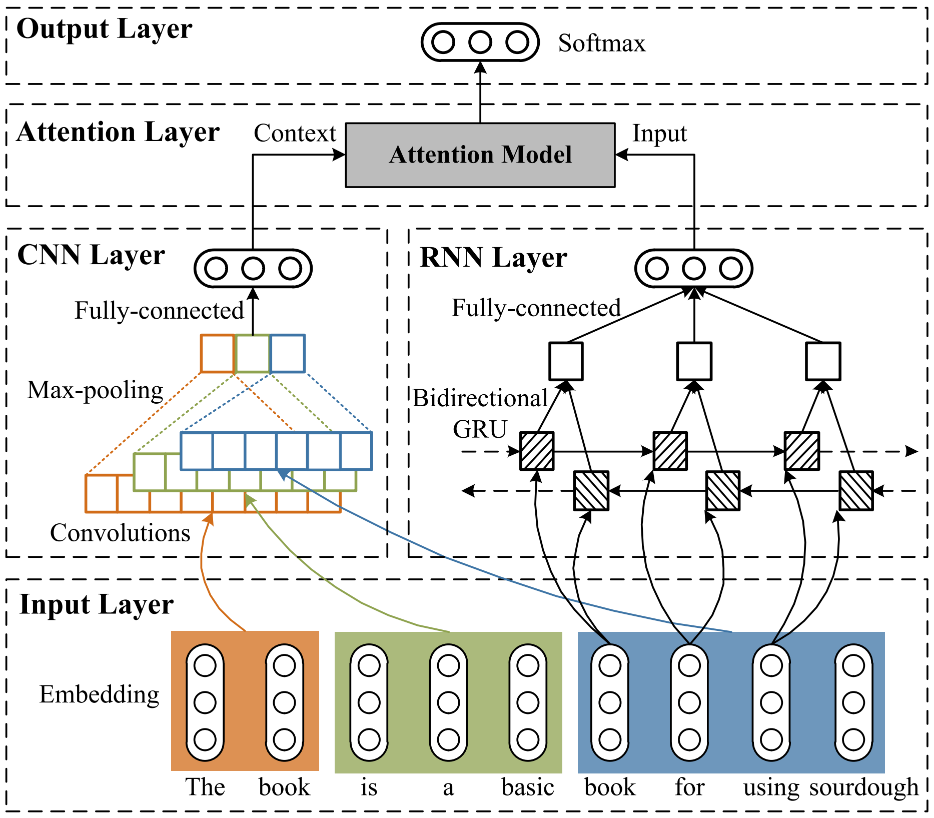

Fig. 2: The hybrid CNN-RNN attention-based model. The shaded rectangles represent the filters in the CNN layer with different sizes. Please refer to Figure 1 for more details of the attention model.

where f is a non-linear function such as tanh or a rectifier. The filter **F** is applied to each possible window of words in the embedding sequence to produce a feature map $\mathsf{c} = [c_1, c_2, ..., c_{n-h+1}]$. With k filters, the first stage produces k feature maps. The second stage is called "pooling", where a max-over-time pooling operation is applied to each feature map and the maximum value $\hat{c} = \max\{\mathsf{c}\}$ is taken as the feature corresponding to **F**. As a result, k features are extracted from the feature maps to form the penultimate layer $\mathsf{p} = \{\hat{c}_1, \hat{c}_2, ..., \hat{c}_k\}$ and are passed to a fully connected layer to reshape the output of the CNN layer, i.e., $\mathsf{p} \in \mathbb{R}^{d_a}$.

2.3 RNN Layer

The RNN layer is used to extract the temporal features and long-term dependencies from the text sequences. RNN is a class of neural network that maintains internal hidden states to model the dynamic temporal behaviour and long-distance patterns of sequences through directed cyclic connections between its units. Take the standard RNN as example. Given the word embedding vectors $\{e_1, e_2, ..., e_n\}$, the word vectors are put into the recurrent layer step by step. For each step t, the network accepts the word vector e_t and the output of the previous step h_{t-1} as the input and produces the current output h_t by a linear transformation followed by a non-linear activation function (e.g., tanh), as follows:

$$h_t = \tanh(\mathbf{W}e_t + \mathbf{U}h_{t-1} + b). \tag{3}$$

To avoid the vanishing gradient problem suffered by the standard RNN, GRU is proposed to control the update of the information via two types of gates: the reset gate r_t and the update gate z_t, which is defined by the following equations:

$$\begin{aligned} r_t &= \sigma(\mathbf{W}_{er}e_t + \mathbf{W}_{hr}h_{t-1} + b_r) \\ z_t &= \sigma(\mathbf{W}_{ez}e_t + \mathbf{W}_{hz}h_{t-1} + b_z) \\ \tilde{h}_t &= \tanh(\mathbf{W}_{eh}e_t + \mathbf{W}_{hh}(r_t \odot h_{t-1}) + b_h) \\ h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t. \end{aligned} \tag{4}$$

One property of the one-directional recurrent layer is that there is imbalance in the amount of information seen by the hidden states at different time steps. This can be easily alleviated by having a bidirectional recurrent layer which is composed of two GRU layers working in opposite directions, as shown in Fig. 2. We obtain the representation of the word w_t by concatenating the forward hidden state $\overrightarrow{h}_t$ and the backward hidden state $\overleftarrow{h}_t$, i.e., h_t = [$\overrightarrow{h}_t$, $\overleftarrow{h}_t$]. In this way, the document can be represented as $\mathtt{h} = \{h_1, h_2, ..., h_n\}$. Before feeding $\mathtt{h}$ into the next attention layer, we use a fully-connected layer to reshape the dimension of $\mathtt{h}$, in order to be aligned with the output of the CNN layer, i.e., $\mathtt{h} \in \mathbb{R}^{n \times d_a}$.

2.4 Attention Layer

The key component of CRAN is the attention layer which combines the output of the CNN layer and RNN layer effectively. In CRAN, the attention mechanism is first applied on top of the output of RNN, i.e., $\mathtt{h}$. For each time step t, we first feed h_t through a fully-connected network to get u_t as a hidden representation of h_t, and then measure the importance of the word w_t as the similarity of u_t with a context vector u_w, followed by a softmax function to get a normalized importance weight a_t. After that, we compute the text vector $\mathtt{d}$ as a weighed arithmetic mean of $\mathtt{h}$ based on the weights $\mathtt{a} = \{a_1, a_2, ..., a_n\}$. The above procedure is defined as :

$$\begin{aligned} u_t &= \tanh(\mathbf{W}_w h_t + b_w) \\ a_t &= \frac{\exp(u_t^T u_w)}{\sum_t \exp(u_t^T u_w)} \\ \mathtt{d} &= \sum_t a_t h_t. \end{aligned} \tag{5}$$

The context vector u_w contains useful information to guide the attention model to locate the informative word from the input sequences, and thus plays an important role in the attention mechanism. However, previous works either ignore the context vector u_w or initialize u_w randomly, which weakens the role of the context significantly. Inspired by the observation and the capability of the CNN model which is effective at exploring regional syntax of words, we propose to use the output of the CNN layer as the context of the attention model. As a result, the new attention layer can be defined as follows:

$$\begin{aligned} u_t &= \tanh(\mathbf{W}_w h_t + b_w) \\ a_t &= \frac{\exp(u_t^T \mathtt{p})}{\sum_t \exp(u_t^T \mathtt{p})} \\ \mathtt{d} &= \sum_t a_t h_t = \mathtt{ah}. \end{aligned} \tag{6}$$

where $\mathtt{p} \in \mathbb{R}^{d_a}$ is the output of the CNN layer and $\mathtt{h} \in \mathbb{R}^{n \times d_a}$ is the output of the RNN layer.

Note that different from traditional attention mechanism where the context vector is the same to all the samples, each sample in CRAN has a unique context vector (i.e., the output of CNN), which provides more flexibility and potential to achieve better performance. By merging the CNN layer and RNN layer using the attention mechanism, the unified model can pick the useful local features from the sequences generated by the RNN model according to the context generated by the CNN model, and thus reserve the merits of both models.

2.5 Output Layer

Our model is designed to support both the multi-class text classification and multi-label text classification.

Multi-class classification. When used for multi-class classification, we use a softmax classifier to predict a single label y from a discrete set of classes Y for a text T. The classifier takes the output of the attention layer as the input and computes the predictive probabilities for all the labels. Then the label with the highest probability is chosen as the label of T.

$$\begin{aligned} &p(y|T) = softmax(\mathbf{W}_{as}\mathtt{d} + b_s) \\ &y = \arg\max_y p(y|T) \end{aligned} \tag{7}$$

The cost function is the categorical cross-entropy loss function:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \bar{y}_i \log(y_i) + \lambda||\theta||_2^F \tag{8}$$

where $\bar{y}_i$ is the ground truth of text t_i, y_i is the estimated probability for each class, m is the size of the dataset, and λ is an L2 regularization hyper-parameter.

Multi-label classification. When used for multi-label classification, we add a fully connected output layer with a sigmoid activation function to predict k labels $\mathtt{y}$ from Y for a text T, which produces a probability for each of the labels. Then the k labels with

(a) Multi-class classification

Dataset	Train	Test	Classes	Classification Task
AG's news	120k	7.6k	4	English news categorization
Sogou news	450k	60k	5	Chinese news categorization
DBPedia	560k	70k	14	Ontology classification
Yelp Review Polarity	560k	38k	2	Sentiment analysis
Yelp Review Full	650k	50k	5	Sentiment analysis
Yahoo! Answers	1400k	60k	10	Topic classification
Amazon Review Full	3000k	650k	5	Sentiment analysis
Amazon Review Polarity	3600k	400k	2	Sentiment analysis

(b) Multi-label classification

Subject	Train	Test	Classes	#Labels
English	16k	1.8k	236	1.25
Chinese	22k	2.4k	72	1.10
Chemistry	320k	35k	536	1.47
Physics	380k	40k	430	1.49
Mathematics	480k	53k	782	1.35

Table 1: Large-scale text classification data sets. #Labels represents the average number of labels per question.

the highest probabilities are chosen as the labels of T.

$$\begin{aligned} &p(y|T) = sigmoid(\mathbf{W}_{as}\mathtt{d} + b_s) \\ &\mathtt{y} = \arg \operatorname*{sort}_{0:k} p(y|T) \end{aligned} \tag{9}$$

The cost function is the binary cross-entropy loss function:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{K}(\bar{y}_{ij}\log(y_{ij}) + (1-\bar{y}_{ij})\log(1-y_{ij})) + \lambda||\theta||_2^F \tag{10}$$

where $\bar{y}_{ij}$ is the ground truth for label l_j of text t_i, y_{ij} is the estimated probability for l_j, and K is the number of classes.

3 Experiments

3.1 Datasets

We conduct extensive experiments on several datasets for both the multi-class classification and multi-label classification, which are introduced as follows.

Multi-class classification. We evaluate the effectiveness of our model on eight freely available large-scale datasets introduced by [26] which cover several classification tasks such as news categorization, sentiment analysis and topic classification. A summary of the statistics for each dataset is listed in Table 1(a). Please refer to [26] for more detailed information of the datasets.

Multi-label classification. Since most datasets for multi-label classification are quite small and contain small number of classes, we build several real-world datasets to fully evaluate the effectiveness of our model, which contain large number of questions collected from the exercises and exams of senior high school in China. Table 1(b) shows the statistics for each dataset. The datasets cover several subjects such as mathematics and physics, and the classes represent the knowledge points corresponding to each subject.

3.2 Baselines

We compare CRAN with several baseline methods, which cover most of the existing CNN-based approaches and RNN-based approaches [3]. For fairness, we do not compare with the models which utilize the hierachical structure of the documents [20, 24], where our model can be adopted as a component in their models. In our experiments, all the models take the whole document as a single sequence. Since the baselines are designed for multi-class classification task, we modify their output layer to support multi-label classification and conduct new experiments on the multi-label classification datasets. In the following, we introduce the baseline methods briefly.

- **CNN-word** represents the word-based CNN model [12], where a CNN model with one convolutional layer is applied to the word embeddings of the documents.
- **CNN-char** represents the character-based CNN model [26], where a CNN model with six convolutional layers is applied to the character representation of the documents.
- **LSTM** treats the whole document as a single sequence and the average of the hidden states of all words are used as feature for classification.
- **Att-BLSTM** represents the attention-based bidirectional LSTM [28], where the attention model is adopted on top of a bidirectional LSTM layer. Att-BLSTM is similar with our CRAN model without the CNN layer.
- **CNN-RNN** represents a hybrid model that processes the input sequence of characters with a number of convolutional layers followed by a single recurrent layer [22].
- **VD-CNN** represents a very deep character-based CNN model [7], where a CNN model with up to 29 convolutional layers is applied to the character representation of the documents.

3.3 Model Settings and Training

The word embeddings can be pretrained or directly trained together with the model. In our experiments, we adopt the pretrained word embeddings. We obtain the word embeddings by training an unsupervised word2vec [17] model on the training and testing datasets. We set the wording embedding dimension to 300.

Since there is no official validation set, we randomly select 10% of the training samples as the validation set. The hyper parameters of the models are tuned on the

[3] We ignore some relevant models because of the unavailability of their codes

validation set. In terms of the CNN layer, we use filters of $3, 4, 5$ with 256 feature maps each and reshape the dimension of penultimate vector p to 200. In terms of the RNN layer, we set the GRU dimension to 100 and thus the bidrectional GRU layer gives us the representation vector h_t with 200 dimensions. Dropout is an effective way to regularize deep neural networks. We apply dropout after the CNN layer as well as after the RNN layer, where the dropout rate is set as 0.3. Other parameters in our model are initialized randomly.

Our model was trained using Adam [13] with a learning rate of 0.001 and a mini-batch of size 256. The input text is padded to a fixed size of 300, where longer text is truncated and masks are generated to help identify the padded region. The implementation is done using Keras with Theano as the backend. All experiments are performed on a single GeForce GTX TITAN X GPU.

3.4 Experimental Results and Analysis

The experimental results of all the models on the multi-class classification datasets and multi-label classification datasets are shown in Table 2 and Table 3, respectively. In the following, we offer some empirical analysis of our CRAN model with respect to different baseline models.

CRAN is an effective hybrid framework. Experimental results show that the attention mechanism is an effective framework to unify CNN and RNN. As shown in Table 2 and Table 3, CRAN performs much better than the standalone CNN and RNN. In particular, CNN-RNN is a rather coarse combination manner for unifying CNN and RNN. Although it can improve performance compared with CNN or RNN, it performs much worse compared with CRAN.

The CNN layer serves as a meaningful context for the attention mechanism. Although Att-BLSTM performs as a rather strong model which can beat most of the existing methods, our CRAN model can still perform better than Att-BLSTM. Remember that Att-BLSTM is similar to our CRAN model except that the context of Att-BLSTM is ignored. This fact proves the effectiveness of the CNN layer as a context of the attention model. The CNN layer can provide useful information for CRAN to pick the important words from the sequences generated by the RNN layer.

CRAN can achieve better performance than the state-of-the-art model VD-CNN while with much fewer parameters. Specifically, CRAN performs much better than CNN-char with 6 convolutional layers and the VD-CNN with 17 convolutional layers. Even though VD-CNN increases its number of layers to 29, CRAN can still beat VD-CNN on most of the datasets for multi-class classification and all the datasets for multi-label classification. In particular, CRAN performs much better than VD-CNN on multi-label classification which is a significantly harder task than multi-class classification. In addition, Table 4 shows the number of parameters for the models. As shown, CRAN has significantly fewer parameters compared with CNN-char and VD-CNN.

3.5 Visualization of Attention

To further show the effectiveness of our model in selecting informative words in a document, we visualize the attention layers of Att-BLSTM and CRAN in Figures 3 and 4.

Method	Depth	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
CNN-word	1	9.92	4.39	1.42	4.60	40.16	31.97	44.40	5.88
CNN-char	6	9.85	4.88	1.66	5.25	38.40	29.55	40.53	5.50
LSTM	-	13.94	4.82	1.45	5.26	41.38	29.16	40.57	6.10
Att-BLSTM	-	8.98	4.10	1.33	4.98	36.77	27.25	38.05	4.95
CNN-RNN	2-3	8.64	4.83	1.43	5.51	38.18	28.26	40.77	5.87
VD-CNN	9	9.17	3.58	1.35	4.88	36.73	27.60	37.95	4.70
VD-CNN	17	8.88	3.54	1.40	4.50	36.07	27.35	37.50	4.41
VD-CNN	29	8.67	3.18	1.29	4.28	35.28	26.57	**37.00**	**4.28**
CRAN	1	**8.30**	**3.15**	**1.05**	**4.00**	**34.69**	**26.28**	37.28	4.35

Table 2: Testing errors of all the models for multi-class text classification, where numbers are in percentage. For VD-CNN, we report the results of the model with different number of layers.

	English			Chinese			Chemistry			Physics			Mathematics		
	k=1	k=2	k=3	k=1	k=2	k=3	k=1	k=2	k=3	k=1	k=2	k=3	k=1	k=2	k=3
CNN-word	54.42	45.41	38.54	79.22	62.94	48.66	55.41	49.27	43.24	61.94	57.80	49.87	54.56	50.52	43.31
CNN-char	42.46	37.23	33.64	78.07	61.98	48.83	50.41	46.52	40.58	57.68	54.74	48.11	49.11	46.63	40.81
LSTM	53.24	46.27	37.23	81.41	62.76	47.66	54.24	48.49	42.51	60.24	57.75	50.01	53.28	49.79	42.27
Att-BLSTM	58.21	49.02	42.61	85.34	63.76	49.74	58.74	52.11	45.72	63.96	60.23	51.42	56.66	53.18	46.09
CNN-RNN	59.89	47.18	41.61	84.44	63.30	49.08	56.91	51.22	45.67	63.67	59.28	52.44	56.51	52.95	45.83
VD-CNN	53.09	43.50	37.74	82.30	63.00	48.89	57.05	51.90	44.92	63.79	59.33	51.57	56.91	53.31	46.10
CRAN	**62.05**	**51.79**	**44.49**	**86.19**	**64.15**	**50.33**	**60.54**	**54.81**	**47.17**	**65.66**	**61.16**	**52.82**	**57.02**	**53.32**	**46.29**

Table 3: F1 scores of all the models for multi-label text classification, where numbers are in percentage. For VD-CNN, we only report the best result when the number of layers is 29.

Method	Depth	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
CNN-char	6	27M	27M	27M	27M	27M	27M	2.7M	2.7M
VD-CNN	9	2.2M	2.2M	2.2M	2.2M	2.2M	2.2M	2.2M	2.2M
VD-CNN	17	4.3M	4.3M	4.3M	4.3M	4.3M	4.3M	4.3M	4.3M
VD-CNN	29	4.6M	4.6M	4.6M	4.6M	4.6M	4.6M	4.6M	4.6M
CRAN	1	**1.2M**	**1.2M**	**1.2M**	**1.2M**	**1.2M**	**1.2M**	**1.2M**	**1.2M**

Table 4: Number of parameters per model. Note that CNN-char adopts the convolutional layer with more filters than that of VD-CNN and thus has more parameters than VD-CNN.

We pick two typical examples from the AG's news and Yahoo! Answers, respectively. We group the words into three categories according to their weights, which is denoted as the blue rectangles with different color depth.

Figure 3 shows a document picked from the AG's news corpus. As shown, CRAN predicts the document to be 4, which represents "Sci/Tech", while Att-BLSTM labels the document with 3, which represents "Business". The reason that CRAN can predict the correct label is that it selects the words carrying strong information, i.e., "information technology", which are highly related to "Sci/Tech". On the other hand, Att-

Ground Truth: 4

Att-BSLTM: Prediction 3
does nick carr matter strategy business concludes that a controversial new book on the strategic value of information technology is flawed but correct

CRAN: Prediction 4
does nick carr matter strategy business concludes that a controversial new book on the strategic value of information technology is flawed but correct

Fig. 3: Documents from AG's news. Label 3 means Business while label 4 means Sci/Tech.

Ground Truth: 8

Att-BSLTM: Prediction 5
h ow many holes does a movie theater screen have every movie theater screen i ve ever seen has had a bunch of holes in it i think its so they can have speakers behind the screen is there a standard for the size and spacing of the holes is there a certain horizontal and vertical count

CRAN: Prediction 8
h ow many holes does a movie theater screen have every movie theater screen i ve ever seen has had a bunch of holes in it i think its so they can have speakers behind the screen is there a standard for the size and spacing of the holes is there a certain horizontal and vertical count

Fig. 4: Documents from Yahoo! Answers. Label 5 means Computers & Internet while label 8 means Entertainment & Music.

BLSTM focuses on the words "strategy business" which are more related to "Business" and mislead the judgement. A similar phenomenon can be observed in Figure 4. The document in Figure 4 is more complicated than that in Figure 3. Att-BLSTM distributes more weights on the words "screen" where "screen" is a rather common words in the field of "Computers". "screen" also obtains a rather high weight in CRAN. However, CRAN also gives high weights for the words "movie" and "theater", which helps the model to recognize that "screen" represents the screen of movie theater instead of computer.

From the above examples, we can see CNN indeed serves as a meaningful context for the attention mechanism. CNN can help the attention mechanism to focus on more correct informative words to help the model make the right decision.

4 Related Work

In this section, we review the existing solvers for text classification according to the evolving of their two primary components: feature representation and classifier design. The early methods [10, 18, 21] on text classification used bag-of-words or n-grams to construct features and adopted traditional classifiers.

In recent years, various deep learning based models have been proposed and achieved state-of-the-art performance. We group them into the following categories and conduct a brief review in the following.

Convolution neural networks. Since 2014, the CNN-based and RNN-based approaches have emerged and become the trend for text classification. Kim in [12] proposed convolutional neural networks (CNN) for sentence classification. By word embedding, a sentence with length n is converted into $n \times k$ matrix representation. Then, a convolution operation with a filter window of length h words, together with a max-over-time pooling layer is adopted. In DCNN proposed in [11], Kalchbrenner et al. applied dynamic k-max pooling over time to generalize the original max pooling in traditional CNN. There are also several convolutional networks proposed to work at character level [26, 7] and very deep models (up to 29 layers) can be designed with small convolutions and pooling operations.

Recurrent neural networks. Carrier and Cho in [4] gave a tutorial on using a recurrent neural network for sentiment analysis which is one type of text classification. Their model adopts a single long short-term memory layer followed by a mean pooling over time. Lai et al. in [14] introduced a bidirectional recurrent neural network for text classification, which adopts a bidirectional long short-term memory layer followed by a max pooling over time. A similar structure is proposed in [25] for relation classification. The goal of the recurrent structure is to capture contextual information as far as possible when learning word representations.

Hybrid CNN-RNN models. Tang et al. in [20] proposed a hierachical neural network model named Conv-GRNN for hierarchical processing of a document. Their model first learns sentence representation with convolutional neural network or long short-term memory. Then the representation of the sentences is sent to a gated recurrent neural network. Xiao and Cho [22] proposed a hybrid model named ConvRec that processes an input sequence of characters with a number of convolutional layers followed by a single long short-term memory layer. Other CNN-RNN models including [3, 15, 27] apply a similar serial structure with ConvRec. The major difference between Conv-GRNN and ConvRec lies in the purpose of combining the convolutional network and recurrent network. In conv-GRNN, the convolutional network is constrained to model each sentence, and the recurrent network to model inter-sentence structures. While in ConvRec, the recurrent layer is applied to the output of the convolutional layers to capture the long-term dependencies, where the whole document is viewed as a single sentence.

Different from ConvRec, our proposed CRAN model combines the convolutional network and the recurrent network with the help of the attention mechanism. In addition, CRAN is orthogonal to the hierarchical model Conv-GRNN and can be plugged in Conv-GRNN as a sentence feature extraction module.

Attention-based models In [28], Att-BLSTM is proposed to capture the most import semantic information in a sentence for relation classification, where the attention model is applied on top of a bidirectional LSTM layer. In [24], bidirectional GRU with two levels of attention mechanisms named HAN were proposed for text classification. The former is used to encode the sentences and the latter to capture importance of the sentences. Att-BLSTM can be viewed as an component in HAN, where Att-BLSTM is used to capture the important words in the word attention encoder and the important sentences in the sentence attention encoder of HAN.

The difference between CRAN and Att-BLSTM is that Att-BLSTM adopts a randomly initialized vector as the context of the attention mechanism while CRAN utilizes the output of the CNN model as the context which can provide more instructive information. Again, CRAN is orthogonal to HAN and can be adopted as a component in HAN similar to Att-BLSTM.

5 Conclusion

In this paper, we propose a hybrid CNN-RNN attention-based neural network, named CRAN, which combines the convolutional neural network and recurrent neural network effectively with the help of the attention mechanism. The proposed model is able to pick the useful local features from the sequences generated by the RNN layer according to the context generated by the CNN layer, and thus reserve the merits of both models in representing a piece of text. We validate the proposed model on several large scale datasets for both multi-class and multi-label text classification. Experimental results show that CRAN can achieve state-of-the-art performance on most of the datasets. The proposed model is a general hybrid architecture that is not limited to text classification or natural language inputs. It will be interesting to see future research on applying the architecture on other applications.

6 Acknowledge

This work is supported in part by the National Natural Science Foundation of China under Grant No. 61702016, 61602087, 61632007, 61572039, the China Postdoctoral Science Foundation under Grant No. 2017M610019, 973 program under No. 2014CB340405, Shenzhen Gov Research Project JCYJ20151014093505032, the Fundamental Research Funds for the Central Universities under grants No. ZYGX2016J080, ZYGX2014Z007.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR **abs/1409.0473** (2014)
2. Blanzieri, E., Bryl, A.: A survey of learning-based techniques of email spam filtering. Artif. Intell. Rev. **29**(1), 63–92 (Mar 2008)
3. Cai, R., Zhang, X., Wang, H.: Bidirectional recurrent convolutional neural network for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers (2016), http://aclweb.org/anthology/P/P16/P16-1072.pdf
4. Carrier, P.L., Cho, K.: Lstm networks for sentiment analysis. Deep Learning Tutorials (2014)
5. Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. CoRR **abs/1406.1078** (2014), http://arxiv.org/abs/1406.1078
6. Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., Bengio, Y.: Attention-based models for speech recognition. In: Advances in Neural Information Processing Systems. pp. 577–585 (2015)
7. Conneau, A., Schwenk, H., Barrault, L., LeCun, Y.: Very deep convolutional networks for natural language processing. CoRR **abs/1606.01781** (2016)
8. Hermann, K.M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P.: Teaching machines to read and comprehend. CoRR **abs/1506.03340** (2015)

9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–80 (1997)
10. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: ECML. vol. 1398, pp. 137–142. Springer (1998)
11. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: ACL (1). pp. 655–665. The Association for Computer Linguistics (2014)
12. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP. pp. 1746–1751 (2014)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014), http://dblp.uni-trier.de/db/journals/corr/corr1412.html/KingmaB14
14. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: AAAI. pp. 2267–2273. AAAI Press (2015)
15. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. pp. 2267–2273. AAAI'15, AAAI Press (2015), http://dl.acm.org/citation.cfm?id=2886521.2886636
16. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: HLT-NAACL. pp. 142–150 (2011)
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS. pp. 3111–3119 (2013)
18. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? sentiment classification using machine learning techniques. In: EMNLP. pp. 79–86 (July 2002)
19. Song, J., Guo, Z., Gao, L., Liu, W., Zhang, D., Shen, H.T.: Hierarchical LSTM with adjusted temporal attention for video captioning. CoRR **abs/1706.01231** (2017), http://arxiv.org/abs/1706.01231
20. Tang, D., Qin, B., Liu, T.: Document modeling with gated recurrent neural network for sentiment classification. In: EMNLP. pp. 1422–1432 (2015)
21. Wang, S.I., Manning, C.D.: Baselines and bigrams: Simple, good sentiment and topic classification. In: ACL (2). pp. 90–94. The Association for Computer Linguistics (2012)
22. Xiao, Y., Cho, K.: Efficient character-level document classification by combining convolution and recurrent layers. CoRR **abs/1602.00367** (2016)
23. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. Computer Science pp. 2048–2057 (2015)
24. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A.J., Hovy, E.H.: Hierarchical attention networks for document classification. In: HLT-NAACL. pp. 1480–1489 (2016)
25. Zhang, D., Wang, D.: Relation classification via recurrent neural network. CoRR **abs/1508.01006** (2015)
26. Zhang, X., Zhao, J.J., LeCun, Y.: Character-level convolutional networks for text classification. In: NIPS. pp. 649–657 (2015)
27. Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., Xu, B.: Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. In: Calzolari, N., Matsumoto, Y., Prasad, R. (eds.) COLING. pp. 3485–3495. ACL (2016), http://dblp.uni-trier.de/db/conf/coling/coling2016.html//ZhouQZXBX16
28. Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H., Xu, B.: Attention-based bidirectional long short-term memory networks for relation classification. In: ACL (2016)