# Enhanced-RCNN: An Efficient Method for Learning Sentence Similarity

Shuang Peng
Ant Financial Services Group
HangZhou, China
jianfeng.ps@antfin.com

Hengbin Cui
Ant Financial Services Group
HangZhou, China
alexcui.chb@antfin.com

Niantao Xie
Peking University
BeiJing, China
xieniantao@pku.edu.cn

Sujian Li
Peking University
BeiJing, China
lisujian@pku.edu.cn

Jiaxing Zhang
Ant Financial Services Group
HangZhou, China
jiaxing.zjx@antfin.com

Xiaolong Li
Ant Financial Services Group
HangZhou, China
xl.li@antfin.com

## ABSTRACT

Learning sentence similarity is a fundamental research topic and has been explored using various deep learning methods recently. In this paper, we further propose an enhanced recurrent convolutional neural network (Enhanced-RCNN) model for learning sentence similarity. Compared to the state-of-the-art BERT model, the architecture of our proposed model is far less complex. Experimental results show that our similarity learning method outperforms the baselines and achieves the competitive performance on two real-world paraphrase identification datasets.

## CCS CONCEPTS

• **Information systems → Information retrieval**; • **Computing methodologies → Natural language processing**.

## KEYWORDS

Sentence Similarity; Deep Learning; Recurrent Convolutional Neural Network

**ACM Reference Format:**
Shuang Peng, Hengbin Cui, Niantao Xie, Sujian Li, Jiaxing Zhang, and Xiaolong Li. 2020. Enhanced-RCNN: An Efficient Method for Learning Sentence Similarity. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366423.3379998

## 1 INTRODUCTION

Learning sentence similarity is a fundamental and important natural language process (NLP) task which may be greatly enhanced by modeling the underlying semantic representations of compared sentences [18]. For example, in most intelligent customer service systems, determining whether the user's question has the same meanings as standard questions in the knowledge system can help accurately match the user's question and provide solutions in time. In particular, a model should not be susceptible to variations of wording or syntax used to express the same idea. Moreover, a good model should also have the capacity to learn sentence similarity regardless of the length of the text and also needs to be efficient when applied to real-world applications. Learning such sentence similarity by high-efficiency methods has attracted many research interests [30, 33]. However, this remains a hard problem.

With the renaissance of neural network models and the development of large annotated data, many deep learning methods are introduced in learning sentence similarity. Compared with traditional NLP methods, deep learning methods take better into account both semantics and structure of sentences and therefore are more suitable for text representation [9]. Recent progress in learning sentence similarity has demonstrated the effectiveness of three types of frameworks. The first one is the Siamese network where two symmetric networks such as the convolutional neural network (CNN) and recurrent neural network (RNN) with sharing parameters are used to model the compared sentences [2, 18, 20, 22]. This framework focuses on modeling essential information in each sentence and ignores the interaction between the two sentences in the encoding process. The second framework is called "matching-aggregation" which strengthens capturing the interactive representations between two sentences [5, 6, 27, 33]. For example, the Enhanced Sequential Inference Model (ESIM) employs attention-based LSTM to extract high-order interactions between two sentences and achieve excellent performance. This kind of model is always computationally complex to care for both the sentence features and sentence interaction. The last framework is called "BERT-based" methods which fine-tune the pre-trained deep language models directly with minimal modifications to perform downstream tasks, including learning sentence similarity [9, 12, 31]. However, the parameter size and inference time cost of "BERT-based" methods are at least a dozen times than "Siamese Network" and "matching-aggregation" based methods.

Inspired by this recent progress, in this paper, we combine the advantages of the first two kinds of sentence similarity learning frameworks and introduce an enhanced recurrent convolutional neural network called Enhanced-RCNN. In the Enhanced-RCNN model, we employ the Siamese multi-layer CNNs to extract key information from the two sentences and adopt the attention-based RNNs to capture the interactive effects between two sentences. Compared to the traditional sequential encoding, the incorporation of CNNs can reduce the computational complexity and capture more fine-grained features [4]. With the combination of CNN


*WWW '20, April 20–24, 2020, Taipei, Taiwan*

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3379998

and RNN, we can better grasp the similarity and difference between two sentences, and we further design the fusion layer to combine two sentence representations for computing the final similarity score. Through carefully designing the *Encoder* and *Fusion* layer, we achieve obvious improvements in learning sentence similarity on two benchmark datasets for paraphrase identification task. Besides, compared with the recently introduced BERT model, Enhanced-RCNN also has competitive performance and with far less complexity.

## 2 RELATED WORK

The deep learning applications in learning sentence similarity have recently received much attention [9, 18, 27], starting from the availability of high-quality semantic word representations [9, 17] and the seminal papers introduce creative frameworks for learning sentence similarity. In the past few years, many deep learning models based on Siamese network or "matching-aggregation" framework have made progress in learning sentence similarity [5, 6, 18, 27, 30, 32, 33].

### 2.1 Siamese Network Framework

For Siamese network framework [2, 18], two input sentences are applied by the same neural network encoder (e.g., a CNN or an RNN) individually, so that both of the two sentences are encoded into vectors in the same embedding space. Then, a matching interaction is made solely based on the two-sentence vectors. The advantage of this framework is that sharing parameters makes the model smaller and easier to train, and the sentence vectors can be used directly for measuring similarity based on cosine distance or another type of distance. However, a disadvantage is that there is no explicit interaction between the two sentences during the encoding procedure, which may lose some vital information.

To deal with this problem, some prior work also incorporates attention mechanisms into Siamese network framework [5, 32] to address the problem of insufficient interactions. Different from learning each sentence's representation separately, the attention-based models consider the mutual influence between the two sentences. One of the famous models is ABCNN [32], which is an attention-based CNN for modeling sentence similarity.

### 2.2 Matching-Aggregation Framework

For "matching-aggregation" framework [6, 27], smaller units (such as words or characters) of the two sentences are firstly matched (Prior work usually uses bidirectional RNN to encode variable-length sentences into a fixed-length vector), and then the matching results are aggregated (e.g., typically uses RNN) into a vector to make the final decision. This framework captures more interactive features between the two sentences. Therefore it acquires significant improvements. However, this framework still has some disadvantages. The main problem is the time-consuming matching operations in capturing interactive features.

One famous model based on "matching-aggregation" framework is called ESIM [6], which employs attention-based LSTM to capture high-order interactive information between two sentences. Motivated by the idea of ESIM, there are also some models introducing many complex matching operations to enhance the model's ability to extract mutual information between two sentences.

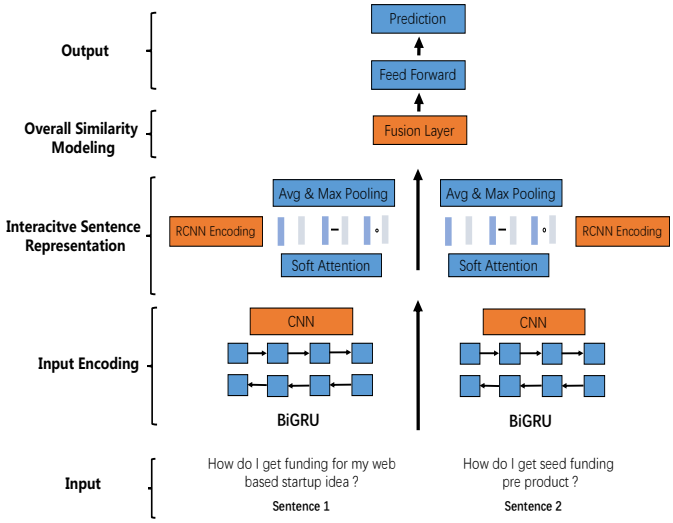



**Figure 1: The architecture of Enhanced-RCNN model**

### 2.3 Pre-trained Language Model

Recently pre-trained language models have attracted many research attentions. As a typical representative, BERT is one of the critical innovations in the recent progress of contextualized representation learning [9]. The idea behind the progress is that even though the word embedding [17] layer (in a typical neural network for NLP) is trained from large-scale corpora, training a wide variety of neural architectures that encode contextual representations only from the limited supervised data on end tasks is insufficient. BERT adopts a fine-tuning approach that requires almost no specific architecture for each end task. This is desired as an intelligent agent should minimize the use of prior human knowledge in the model design. Instead, it should learn such knowledge from data. Based on this design, BERT achieves state-of-the-art performance in many NLP tasks, including paraphrase identification [25].

As introduced before, our work is also based on "matching-aggregation" framework and uses attention-based encoder in learning sentence similarity [6]. The primary improvements of our work are that we design a way to incorporate CNN to extract key information in the sentence. Moreover, we utilize a fusion layer to better combine different sentence representations for overall similarity modeling.

## 3 ENHANCED-RCNN MODEL

Our proposed Enhanced-RCNN model is composed of three components: input encoding, interactive sentence representation, and similarity modeling. A high-level overview of Enhanced-RCNN model is shown in Figure 1. In the following subsections, we will detailedly introduce the three components.

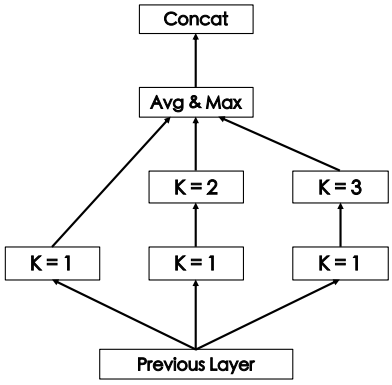



**Figure 2: Structure of our proposed CNN-based Encoding Layers. We implement 1-dimensional convolution with a structure similar to the Inception [23] over the outputs of the RNN encoder, where k refers to the kernel size.**

## 3.1 Input Encoding

Input encoding consists of three parts: bidirectional GRU (BiGRU) [7], CNN, and Pooling. Each part extracts different features from a multi-dimensional perspective. BiGRU extracts the sequence and context features of sentences and can memorize long-distance information. Compared with BiGRU, CNN is better at capturing "keywords and phrases information" in sentences. This characteristic has been used in summarization research [4]. Moreover, the pooling layer in our model processes the CNN output and can help reduce the model parameters and enhance the extensiveness and robustness. We explain the detailed structure of these three parts below.

Assume we have two sentences $\mathbf{a} = (a_1, ..., a_{l_a})$ and $\mathbf{b} = (b_1, ..., b_{l_b})$, where $a_i$ or $b_j \in \mathbb{R}^k$ is an embedding of $k$-dimensional vector, which can be initialized with some pre-trained word (or character) embeddings. The goal is to predict a label $\mathbf{y}$ that indicates the similarity between $\mathbf{a}$ and $\mathbf{b}$.

*3.1.1 RNN Encoder.* We employ BiGRU to encode two input sentences (Equation (1) and (2)), because GRU is computationally more efficient than LSTM and its performance is on par with LSTM [8]. Here BiGRU learns to represent a word (or character) (e.g., $a_i$) and its context. We denote $\bar{\boldsymbol{a}}_i$ as the hidden (output) state generated by the BiGRU at time $i$ over the input sequence $\boldsymbol{a}$. The same is applied to $\bar{\boldsymbol{b}}_j$:

$$\bar{\boldsymbol{a}}_i = \boldsymbol{BiGRU}(\boldsymbol{a}, i), \forall i \in [1, ..., l_a]. \tag{1}$$

$$\bar{\boldsymbol{b}}_j = \boldsymbol{BiGRU}(\boldsymbol{b}, j), \forall j \in [1, ..., l_b]. \tag{2}$$

BiGRU contains two GRUs starting from the left and the right end, respectively. The hidden states generated by these two GRUs at each time step are concatenated to represent that time step and its context.

*3.1.2 CNN Encoder.* Next with the BiGRU results, we apply the improved convolutional neural networks (ImprovedCNN) to capture the important information implicit in the input sequence and generate a fixed-length vector $\widetilde{\boldsymbol{p}}$:

$$\widetilde{\boldsymbol{p}} = ImprovedCNN(\bar{\boldsymbol{p}}) \tag{3}$$

when designing the CNN layer, we refer the idea of *"Network In Network"* (NIN) [11, 14]. Figure 2 shows the detailed structure. The first CNN layer helps reduce the dimension of the output feature map and increase non-linearity. The second CNN layer further captures robust and abstract features.

Due to the specialty of the kernel in CNN, it is better at capturing "keywords and phrases information". The result of each convolution will fire when a particular pattern is detected. By varying the size of the kernels and concatenating their outputs, we can identify patterns of multiples sizes (2, 3, or 5 adjacent words). Patterns like "I hate", "very good" could be expressions and therefore CNNs can identify them in the sentence regardless of their position.

The activation function here is Relu [19]. Specifically, for each CNN component, the calculation formula is as follows:

$$\widetilde{\boldsymbol{p}}_i = Relu(\boldsymbol{W} \cdot \bar{\boldsymbol{p}}_i + bias) \tag{4}$$

where $\boldsymbol{W}$ be the convolution weights and $\bar{\boldsymbol{p}}_i$ be the output of BiGRU at time $i$.

At last, we employ column-wise average and max-pooling to extract features from convolution output and concatenate all these vectors to form the final vector.

## 3.2 Interactive Sentence Representation

The module of interactive sentence representation aims to obtain an appropriate representation for each sentence with consideration of the interactive effects from the other sentence.

*3.2.1 Soft-attention Alignment.* First, we employ the soft attention alignment [1, 15] to associate the relevant parts between two sentences with Equation (5).

$$\boldsymbol{e}_{ij} = \bar{\boldsymbol{p}}_{a_i}^T \cdot \bar{\boldsymbol{p}}_{b_j} \tag{5}$$

where $\boldsymbol{e}_{ij}$ is the attention weight.

Specifically, soft attention is only applied on the BiGRU output $\bar{\boldsymbol{p}}$, because BiGRU captures sequence information in the sentences which can be naturally integrated with the attention-based alignment scheme.

For the element of one sentence encoding, i.e., $\bar{\boldsymbol{p}}_{a_i}$, the relevant semantics in another sentence is measured by $\boldsymbol{e}_{ij}$, more specifically with Equation (6) and Equation (7).

$$\hat{\boldsymbol{p}}_{a_i} = \sum_{j=1}^{l_b} \frac{exp(e_{ij})}{\sum_{k=1}^{l_b} exp(e_{ik})} \bar{\boldsymbol{p}}_{b_j}, \forall i \in [1, ..., l_a], \tag{6}$$

$$\hat{\boldsymbol{p}}_{b_j} = \sum_{i=1}^{l_a} \frac{exp(e_{ij})}{\sum_{k=1}^{l_a} exp(e_{kj})} \bar{\boldsymbol{p}}_{a_i}, \forall j \in [1, ..., l_b], \tag{7}$$

where $\hat{\boldsymbol{p}}_{a_i}$ is a weighted summation of $\left\{\bar{\boldsymbol{p}}_{b_j}\right\}_{j=1}^{l_b}$. Intuitively, the content in $\left\{\bar{\boldsymbol{p}}_{b_j}\right\}_{j=1}^{l_b}$ that is relevant to $\bar{\boldsymbol{p}}_{a_i}$ will be selected and represented as $\hat{\boldsymbol{p}}_{a_i}$.

*3.2.2 Interaction Modeling.* First, we compute the difference and the element-wise product for the tuple $< \bar{\boldsymbol{p}}_a, \hat{\boldsymbol{p}}_a >$ as well as $< \bar{\boldsymbol{p}}_b, \hat{\boldsymbol{p}}_b >$. Then, we employ both average and max-pooling to process the obtained vectors. We expect that such operations could help capture interactive information. Finally, we concatenate all these vectors to perform the interaction modeling. Here we also include the previous CNN output $\widetilde{\boldsymbol{p}}$. The calculation process is as follows:

$$\boldsymbol{v}_a = [\bar{\boldsymbol{p}}_a; \hat{\boldsymbol{p}}_a; \bar{\boldsymbol{p}}_a - \hat{\boldsymbol{p}}_a; \bar{\boldsymbol{p}}_a \odot \hat{\boldsymbol{p}}_a] \tag{8}$$

$$\boldsymbol{v}_b = [\bar{\boldsymbol{p}}_b; \hat{\boldsymbol{p}}_b; \bar{\boldsymbol{p}}_b - \hat{\boldsymbol{p}}_b; \bar{\boldsymbol{p}}_b \odot \hat{\boldsymbol{p}}_b] \tag{9}$$

$$\boldsymbol{v}_{a,ave} = \sum_{i=1}^{l_a} \frac{\boldsymbol{v}_{a_i}}{l_a}, \ \boldsymbol{v}_{a,max} = \max_{i=1}^{l_a} \boldsymbol{v}_{a_i} \tag{10}$$

$$\boldsymbol{v}_{b,ave} = \sum_{j=1}^{l_b} \frac{\boldsymbol{v}_{b_j}}{l_b}, \ \boldsymbol{v}_{b,max} = \max_{j=1}^{l_b} \boldsymbol{v}_{b_j} \tag{11}$$

$$\boldsymbol{o}_a = [\boldsymbol{v}_{a,ave}; \widetilde{\boldsymbol{p}}_a; \boldsymbol{v}_{a,max}] \tag{12}$$

$$\boldsymbol{o}_b = [\boldsymbol{v}_{b,ave}; \widetilde{\boldsymbol{p}}_b; \boldsymbol{v}_{b,max}] \tag{13}$$

Different from the current "matching-aggregation" frameworks, we add input encoding with both RNN and CNN into interactive sentence representation. By incorporating both RNN and CNN, we can capture more fine-grained features (sequence and keywords information) during input encoding. Moreover, due to its parameter sharing scheme, CNN can help reduce the number of parameters in the model.

## 3.3 Similarity Modeling

With the interactive sentence representation $\boldsymbol{o}_a$ and $\boldsymbol{o}_b$ derived, a particular fusion layer has been designed to combine the two sentence representation separately for overall similarity modeling.

*3.3.1 Fusion Layer.* Here we refer the [26] and apply a heuristic matching trick with a difference and element-wise product to combine two sentence representations:

$$m(P, Q) = tanh(W_f[P; Q; P \circ Q; P - Q] + b_f) \tag{14}$$

where $P$ and $Q$ denote two input sentence representations, $\circ$ denotes the element-wise product, and $W_f, b_f$ are trainable parameters. The output dimension is projected back to the same size as the original representation $P$ or $Q$ via the projected matrix $W_f$.

Since we find that the original interactive sentence representations are important in reflecting the semantics at a more global level, we also introduce different levels of gating mechanism to incorporate the projected representations $m(\cdot, \cdot)$ with the original interactive sentence representations. As a result, the fused representations of two sentences can be formulated as:

$$\boldsymbol{o}'_a = g(\boldsymbol{o}_a, \boldsymbol{o}_b) \cdot m(\boldsymbol{o}_a, \boldsymbol{o}_b) + (1 - g(\boldsymbol{o}_a, \boldsymbol{o}_b)) \cdot \boldsymbol{o}_a \tag{15}$$

$$\boldsymbol{o}'_b = g(\boldsymbol{o}_b, \boldsymbol{o}_a) \cdot m(\boldsymbol{o}_b, \boldsymbol{o}_a) + (1 - g(\boldsymbol{o}_b, \boldsymbol{o}_a)) \cdot \boldsymbol{o}_b \tag{16}$$

This process could be regarded as a special case of modeling some high-order interaction between the tuple elements. And the final output of *Fusion Layer* concatenates the output of the first process:

$$\boldsymbol{m}_{out} = [\boldsymbol{o}'_a, \boldsymbol{o}'_b] \tag{17}$$

**Table 1: The descriptions of two paraphrase identification datasets.**

| | Quora | Ant Financial |
|---|---|---|
| Total numbers of sentence pair | 384K | 492K |
| Average length of sentences | 12.56 | 13.01 |
| Ratio of positive and negative examles | 0.59 | 0.62 |

**Table 2: Ablation study on development sets of the corresponding datasets.**

| | Quora | Ant Financial |
|---|---|---|
| Model | Accuracy (%) | |
| **Complete Model** | **88.55** | **77.09** |
| w/o BiGRU | 83.38 | 74.47 |
| w/o CNN | 87.47 | 75.66 |
| w/o Attention | 87.92 | 76.73 |

*3.3.2 Label Prediction.* In the final prediction layer, we put $\boldsymbol{m}_{out}$ into a multi-layer perceptron for label prediction (a binary label). The entire model is trained end-to-end and uses cross-entropy as the loss function.

# 4 EXPERIMENTS

In this section, we evaluate Enhanced-RCNN model on two benchmark datasets. We will first introduce the general setting of Enhanced-RCNN model in subsection 4.1. Then, we demonstrate the properties of Enhanced-RCNN model through some ablation studies in subsection 4.2. Next, we compare Enhanced-RCNN model with current sentence similarity learning models on some standard benchmark datasets in subsection 4.3. Finally, we compare Enhanced-RCNN model with the state-of-the-art BERT model in subsection 4.4 and conduct detailed case analysis on Ant Financial Dataset in subsection 4.5.

## 4.1 Experiment Settings

The experimental datasets come from the "Quora Question Pairs"[1] and "Ant Financial"[2]. These two datasets consist of many question pairs annotated with a binary value indicating whether the two questions are a paraphrase of each other. We split the two datasets into three parts: training pairs, development pairs and testing pairs. For two datasets, the number of development and testing pairs are both 10K[3]. The detailed descriptions of the two datasets are shown in Table 1.

In order to get the best performance, we have tuned the hyper-parameters on the development set. Their values are illustrated as follows:

**Quora Question Pairs:** We use only word-based embeddings for Quora dataset, without char-based embeddings or syntactic features. We initialize word embeddings in the word representation layer with the 300-dimensional GloVe word vectors pre-trained

[1] https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs
[2] https://dc.cloud.alipay.com/index#/topic/data?id=3
[3] In development and testing sets, the ratio between positive and negative examples is 0.5 for Quora dataset and 0.76 for Ant Financial dataset.

**Table 3: Performance for paraphrase identification on two standard benchmark datasets.**

| | Quora | Ant Financial |
|---|---|---|
| Model | Acc / F1 (%) | |
| (1) Siamese-CNN[32] | 79.60 / – | 68.40 / – |
| (2) Multi-Perspective-CNN | 81.38 / – | 69.96 / – |
| (3) Siamese-LSTM[18] | 82.58 / – | 70.11 / – |
| (4) Multi-Perspective-LSTM | 83.21 / – | 71.02 / – |
| (5) L.D.C.[29] | 85.55 / 85.23 | 73.28 / 72.16 |
| (6) BiLSTM-Generalized-Pooling[5] | 86.82 / 85.90 | 74.42 / 72.21 |
| (7) ESIM[6] | 87.50 / 87.44 | 74.91 / 71.27 |
| (8) BiMPM[27] | 88.17 / 87.96 | 74.33 / 72.49 |
| (9) DINN[10] | 89.06 / 89.01 | 74.56 / 72.74 |
| **Enhanced-RCNN** | **89.30 / 89.47** | **75.51 / 73.39** |

**Table 4: Performance comparison with BERT-Base model on two standard benchmark datasets.**

| | Quora | Ant Financial |
|---|---|---|
| Model | Acc / F1 (%) | |
| (1) BERT-Base | 90.09 / 90.20 | 75.77 / 73.40 |
| (2) Enhanced-RCNN | 89.30 / 89.47 | 75.51 / 73.39 |
| **Enhanced-RCNN (ensemble)** | **90.28 / 90.35** | **76.85 / 74.20** |

**Table 5: Parameter size and inference time on Ant Financial dataset between Enhanced-RCNN and BERT-Base model.**

| Model | parameter size | time (s/batch) |
|---|---|---|
| BERT-Base | 102.2M | 0.23 ± 0.20 |
| **Enhanced-RCNN** | **7.7M** | **0.02 ± 0.01** |

from the 840B Common Crawl corpus [21]. We set the hidden size as 192 for all BiGRU layers.

**Ant Financial:** Recent researches find that char-based models consistently outperform word-based models in many Chinese NLP tasks [16]. As a result, for this Chinese paraphrase identification dataset, we do not use word segmentation and use char-based model. We initialize char embeddings in the char representation layer with the 100-dimensional cw2vec [3] pre-trained from the provided question pairs. We set the hidden size as 128 for all BiGRU layers.

For two benchmark datasets, we both initialize the out-of-vocabulary (OOV) embeddings randomly and apply dropout to every layer and set the dropout ratio as 0.2. To train the model, we minimize the cross-entropy of the training set, and use the Adam optimizer [13] to update parameters. We set the learning rate at 0.0005. During training, we still update the pre-trained word embeddings. For all the experiments, accuracy and f1-score are used as the evaluation metric, and we pick the model which obtains the best accuracy on the development set and then evaluate it on the test set.

## 4.2 Ablation Studies

To demonstrate the properties of our model, we perform some ablation studies in this subsection. Table 2 shows the performance on the development set.

First, we study the influence of CNN or RNN (BiGRU) encoder. We build three ablation models different in the encoder layer: 1) Only use CNN in the encoder layer and without the BiGRU; 2) Only use BiGRU in the encoder layer and without the CNN. Comparing the first two ablation models with the "Complete Model", we can observe that removing the BiGRU hurts the performance for about 5% and 2.6%, and removing the CNN hurts the performance for about 1% and 1.4%. Obviously, adding CNN in the encoder is supplemented for extracting more fine-grained features.

Second, we evaluate the effectiveness of attention mechanisms. To this end, we construct the ablation model by removing the attention mechanism. From the experimental results, we can see that the attention mechanism does not have an obvious influence on the performance. This finding implies from other sides that the current complex attention mechanism in learning sentence similarity may be redundant.

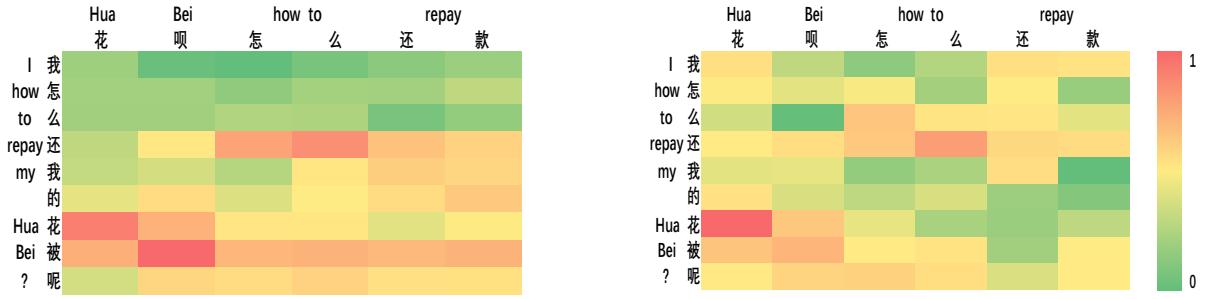
## 4.3 Experiments on Paraphrase Identification

In this subsection, we compare the effect of our model with two types of models ("Siamese" and "Matching-aggregation" frameworks) in learning sentence similarity. We still conduct the experiment on the "Quora Question Pairs" and "Ant Financial technical competition" datasets.

First, under the Siamese framework, we compare our model with two baseline models: Siamese-CNN and Siamese-LSTM. Both of the two models encode two input sentences into sentence vectors with a neural network encoder and make a decision based on the cosine similarity between the two sentence vectors. However, they implement the sentence encoder with a CNN and an LSTM, respectively. We design the CNN and the LSTM model according to the architectures in [28].

Second, based on the two baseline models, we refer three more baseline models Multi-Perspective-CNN, Multi-Perspective-LSTM, and Bilateral-Multi-Perspective-LSTM. In these three models, we change the cosine similarity calculation layer with multi-perspective cosine matching function and apply a fully-connected layer (with sigmoid function on the top) to make the prediction [27].

Third, we further compare the L.D.C. [29], ESIM [6], BiLSTM-Generalized-Pooling [5] and DINN [10] models, which are typical models under the "matching-aggregation" framework.




(a) Enhanced-RCNN (b) Enhanced-RCNN without CNN encoder

**Figure 3: A case study of the paraphrase identification task. The premise is "How to repay Huabei", and the hypothesis is "How do I repay my Huabei ?" (but has some misspellings).**

Table 3 shows the performances of all baseline models and our Enhanced-RCNN model. We can see that RNN-based models (Siamese-LSTM or Multi-Perspective-LSTM) perform much better than CNN-based models (Siamese-CNN or Multi-Perspective-LSTM), which indicates that RNN is more effective for encoding the sentence. From the performance results, we can also see that Multi-Perspective-based models work better than Siamese-based models, which indicates that the multi-perspective matching mechanism is more effective than the single Siamese representations. Our Enhanced-RCNN model outperforms the L.D.C. and ESIM models and achieves similar but better performance than BiMPM and DINN models using complex attention mechanism, which indicates that our designed simple architecture can better match sentences from multiple levels of granularity.

## 4.4 Experiments with BERT

The biggest improvement in recent researches belongs to BERT [9]. Here we also compare our model with BERT-Base model, which has 12 layers, 768 hidden dimensions and 12 attention heads (in transformer [24]) with total parameters of 102M. We train the BERT model (Whole Word Masking) using the official training code and commands released by the authors[4] on Nvidia P100 GPU, which uses a linear layer on top of the pooled `[CLS]` representation from BERT encoding. The experimental results are shown in Table 4. Our single model is highly competitive with BERT-Base model. Moreover, after the model ensemble (with 5-fold), Enhanced-RCNN achieves better performance than BERT-Base model. Here we do not compare ensemble BERT because its parameter size is too large and ensemble does not make much sense in real applications.

To show the efficiency of Enhanced-RCNN model, we also compare the model complexity (parameter size) and inference time with the BERT-Base model. Table 5 shows the comparison results. BERT-Base and Enhanced-RCNN models are required to make predictions for a batch of only one pair of sentences on a MacBook Pro with Intel Core i7 CPUs. The reported statistics are the average and the standard deviation of processing 500 batches. The comparison results show that our method has a very high CPU inference speed. Compared with BERT-Base model, Enhanced-RCNN is 10 times faster and has far less model complexity.

In summary, the experimental results show that our proposed Enhanced-RCNN model achieves performance on par with the state-of-the-art models on two paraphrase identification datasets with only a few parameters and fast inference speed.

## 4.5 Case Study and Error Analysis

In this subsection, we further analyze the attention matching weights in a typical case. We choose Enhanced-RCNN model without CNN encoder as a comparison.

Figure 3 tangibly shows why the Enhanced-RCNN model outperforms other models without CNN encoder. The heatmap denotes the attention matching weights between tokens of two sentences, computed by two models. The input two sentences are (1) 花呗怎么还款 (How to repay Huabei ?) and (2) 我怎么还我的花被呢 (How do I repay my Huabei ?). Specifically, the second input sentence has some misspellings (花被 is wrongly written and should be 花呗). If we remove CNN encoder in the Enhanced-RCNN, 被 fails to be mapped to 呗. But this is not the case with the original Enhanced-RCNN. We infer the incorporation of CNN features in our designed model make the semantically related characters in two sentences to be easily mapped even though it has some misspellings. As a result, Enhanced-RCNN model without CNN encoder fails to determine the similarity of two sentences.

## 5 CONCLUSIONS AND FUTURE WORK

In this paper, we introduce a new model named Enhanced-RCNN in learning sentence similarity. Enhanced-RCNN outperforms in learning sentence similarity on two benchmark datasets, and experimental results demonstrate that our improvements based on the traditional "Matching-aggregation" framework, including the incorporation of CNN in interactive sentence representations and fusion layer in similarity modeling, help extract more elaborated information with lower complexity.

In future work, we plan to extend the Enhanced-RCNN model to other NLP tasks like answer selection and natural language inference.

[4] https://github.com/google-research/bert

## REFERENCES


[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*. 737–744.

[3] Shaosheng Cao, Wei Lu, Jun Zhou, and Xiaolong Li. 2018. cw2vec: Learning Chinese Word Embeddings with stroke n-grams. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*. AAAI.

[4] Ziqiang Cao, Furu Wei, Sujian Li, Wenjie Li, Ming Zhou, and WANG Houfeng. 2015. Learning summary prior representation for extractive summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. 829–833.

[5] Qian Chen, Zhen-Hua Ling, and Xiaodan Zhu. 2018. Enhancing Sentence Embedding with Generalized Pooling. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1815–1826.

[6] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for Natural Language Inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1657–1668.

[7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. [n.d.]. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. ([n. d.]).

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*. 4171–4186.

[10] Yichen Gong, Heng Luo, and Jian Zhang. 2018. Natural Language Inference over Interaction Space. (2018).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[12] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *arXiv preprint arXiv:1907.10529* (2019).

[13] Diedirik P Kingma and Ba J Adam. 2015. a method for stochastic optimization. 2014. *arXiv preprint arXiv:1412.6980* (2015).

[14] Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400* (2013).

[15] Yang Liu, Matt Gardner, and Mirella Lapata. 2018. Structured Alignment Networks for Matching Sentences. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 1554–1564.

[16] Yuxian Meng, Xiaoya Li, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is Word Segmentation Necessary for Deep Learning of Chinese Representations?

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[18] Jonas Mueller and Aditya Thyagarajan. 2016. Siamese Recurrent Architectures for Learning Sentence Similarity.. In *AAAI*, Vol. 16. 2786–2792.

[19] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

[20] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*. 148–157.

[21] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[22] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[23] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2818–2826.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[25] Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* (2018).

[26] Wei Wang, Ming Yan, and Chen Wu. 2018. Multi-Granularity Hierarchical Attention Fusion Networks for Reading Comprehension and Question Answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1705–1714.

[27] Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 4144–4150.

[28] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Semi-supervised Clustering for Short Text via Deep Representation Learning. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 31–39.

[29] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Sentence Similarity Learning by Lexical Decomposition and Composition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 1340–1349.

[30] Runqi Yang, Jianhai Zhang, Xing Gao, Feng Ji, and Haiqing Chen. 2019. Simple and Effective Text Matching with Richer Alignment Features. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 4699–4709.

[31] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv preprint arXiv:1906.08237* (2019).

[32] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *Transactions of the Association of Computational Linguistics* 4, 1 (2016), 259–272.

[33] Kun Zhang, Guangyi Lv, Linyuan Wang, Le Wu, Enhong Chen, Fangzhao Wu, and Xing Xie. 2019. DRr-Net: Dynamic Re-read Network for Sentence Semantic Matching. (2019).