Multimedia Tools and Applications (2021) 80:6479–6492
https://doi.org/10.1007/s11042-020-09996-y

Paraphrase detection using LSTM networks and handcrafted features

Hassan Shahmohammadi[1] · MirHossein Dezfoulian[1] · Muharram Mansoorizadeh[1]

Received: 14 March 2020 / Revised: 23 August 2020 / Accepted: 29 September 2020 /
Published online: 16 October 2020

Abstract
Paraphrase detection is one of the fundamental tasks in the area of natural language processing. Paraphrase refers to those sentences or phrases that convey the same meaning but use different wording. It has a lot of applications such as machine translation, text summarization, QA systems, and plagiarism detection. In this research, we propose a new deep-learning based model which can generalize well despite the lack of training data for deep models. After preprocessing, our model can be divided into two separate modules. In the first one, we train a single Bi-LSTM neural network to encode the whole input by leveraging its pretrained GloVe word vectors. In the second module, three sets of handcrafted features are used to measure the similarity between each pair of sentences, some of which are introduced in this research for the first time. Our final model is formed by incorporating the handcrafted features with the output of the Bi-LSTM network. Evaluation results on MSRP and Quora datasets show that it outperforms almost all the previous works in terms of f-measure and accuracy on MSRP and achieves comparable results on Quora. On the Quora-question pair competition launched by Kaggle, our model ranked among the top 24% solutions between more than 3000 teams.

Keywords Paraphrase detection · Short text similarity · Deep learning · Feature engineering · Information fusion

1 Introduction

With the ever increasing textual data on social media platforms such as Twitter and Facebook, measuring the semantic similarity of short texts is becoming more important, and

✉ Muharram Mansoorizadeh
mansoorm@basu.ac.ir

Hassan Shahmohammadi
h.shahmohammadi@eng.basu.ac.ir

MirHossein Dezfoulian
dezfoulian@basu.ac.ir

[1] Bu-Ali Sina University, Hamedan, Iran

hence, related NLP tasks have been gaining a lot of attention. One of such tasks is paraphrase detection which tries to measure the semantic equivalence of two pieces of text. It is a critical task in many NLP applications such as machine translation, text summarization, QA systems, and plagiarism detection. In this research, we propose a new model that achieves a decent performance, despite the lack of sufficient training data for deep-learning based models. Our model can be divided into three parts. Preprocessing step is the first part that prepares the sentences for the next step. In the second part, terms are mapped to their numerical representations using GloVe word embedding [31]. The output of the embedding layer is then fed into a Bi-LSTM neural network [16] to encode the whole sentence by leveraging its word vectors. In the third part, three sets of fine-grained handcrafted features are provided to measure the similarity between each pair of sentences. Some of these features are introduced in this research for the first time. Our final model is formed by combining the handcrafted features and the output of Bi-LSTM network. Two datasets namely, MSRP [9] and Quora[1] are used to evaluate the model. The result of evaluating the model on MSRP shows that it outperforms many of the previous works in terms of f-measure and accuracy. The evaluation results on Quora also show comparable results among other works. On the Quora-question pair competition launched by Kaggle[2], our model ranked in the top 24% solutions among more than 3000 competitors. The results of the evaluation for both datasets show that our model achieves outstanding results on small datasets and it can generalize well despite the lack of adequate training data for deep models. The rest of this paper is organized as follows: We briefly introduce the related works in Section 2. In the next section, we will elaborate on our proposed model. Thereafter, the datasets and evaluation metrics will be discussed. We will then report the results of our experiments and finally, in Section 6, we will make a conclusion and pinpoint the future directions of our research.

2 Related work

Several works have been carried out on paraphrase detection, especially on MSRP, many of which have focused on handcrafted features. These include SVM and logistic regression trained on machine translation based features [11, 25], employing LCS, WordNet similarity features [10, 18, 23], POS tags similarity metrics [40], and statistical approaches such as TF-IDF, BoW, and LSA [19, 26].

Deep learning models, on the other hand, have been gaining a lot of attention due to their promising performance. Hence, there has been a surge of interests to evaluate them on paraphrase detection as well. He et al. [15] proposed a Siamese convolutional neural network and trained it on GloVe word-embeddings. They used cosine similarity, Euclidean distance, and element-wise absolute difference to measure the similarity between each pair. Milajevs et al. [28] trained a skip-gram model on MSRP and represented each sentence by summing up its word vectors. They set a threshold on the cosine similarity of each pair to predict the final output. Wang et al. [43] proposed a CNN model that first decomposes each pair of sentences to their similar and dissimilar parts, and then extracts their respective feature vectors. They used Word2vec embedding to representing each sentence. Logeswar et al. [24] and Sjoblom et al. [37] both proposed RNN models for sentence modeling and predicting the semantic equivalence for pairs of sentences. Instead of using pre-trained embeddings, the

[1]https://www.kaggle.com/c/quora-question-pairs/data

[2]https://kaggle.com/c/quora-question-pairs

word vectors are learned during the training process. Wang et al. [42] trained a RNN model with attention techniques on GloVe [31] and EMLo [32] word-embeddings.They evaluated their model on several natural language understanding tasks and developed a benchmark called GLUE. We will compare the performance of our model to this benchmark. Radford et al. [33] proposed a new model for word representation by leveraging language modeling techniques. They trained the model on BooksCorpus [45]. The similarity between each pair is the output of a softmax layer on top of the model. While this model works well on huge training data, we will see that its performance declines on small datasets. Sjoblom et al. [37] proposed an RNN model with GRU cells for paraphrase detection of non-English movie subtitles and trained their model on Opusparcus [7]. They also evaluated their model on MSRP. Blacoe et al. [2] tackled with this task by examining distributional models of semantics such as simple vector space model and dense representation obtained from the neural network language model. Cheng et al. [5] presented a deep compositional model of meaning based on recursive and recurrent neural network, during the learning process, a dynamic word sense induction for each word is considered.

Some researchers tried to take advantage of both deep learning models and classical approaches, as deep learning based approaches usually require a large amount of training data. Socher et al. [38] proposed a model based on recursive autoencoders. They built the syntactic tree for each sentence by using a parser and then trained an autoencoder to minimize the error of predicted word vectors. The final feature vector for each pair is formed by a dynamic pooling layer. Zhang et al. [44] followed a similar approach and replaced the dynamic pooling layer by a CNN model. Kenter et al. [21] used word2vec and TF-IDF term weighing to create a similarity matrix for the pairs, they trained an SVM classifier on the obtained features. Agarwal et al. [1] tackled this problem by using a CNN followed by an LSTM network. They also incorporated handcrafted features obtained from TF-IDF term weighting, WordNet, and N-gram similarities. Our approach belongs to this category as well, since we propose a deep learning model combined with handcrafted features.

3 The proposed method

This section elaborates on the model architecture which can be divided into three parts. In the first part, a preprocessing module is employed on each pair of sentences to eliminate unnecessary tokens and prepare them for the next step. In the second part, a deep learning model is proposed by using a single layer Bi-LSTM network on top of an embedding layer initialized by GloVe word-vectors. Finally, in the third part, an end to end model is developed by combining three types of convenient handcrafted features with the deep model. The modules are detailed in the following subsections.

3.1 Preprocessing

Preprocessing is a very critical step in all NLP tasks. It facilitates the learning process by cleaning, augmenting, and enrichment of the raw text. For this purpose, we prepared the datasets by applying the fallowing operations to each pair of sentences.

- Lowercase all the words.
- Extend well-known abbreviations and acronyms (based on a pre-defined dictionary).
- Replace symbols with their names (e.g. '%' with *percent*).

- Replace frequent numbers with their equivalent words (e.g. 1000 with thousand). Frequent numbers are those whose alphabetic form could also be used.
- Remove all the non-ASCII characters.
- Tokenize document to its terms (by filtering punctuation marks and splitting the sentence based on spaces).

We did not perform stemming either stop-word removal since it is previously shown that they can help to detect the nuanced semantic details necessary for paraphrase detection [36]. As an example for the preprocessing module, the sentence *"I live in the USA and I have just won 1000 $ from a lottery"* would be converted to *"i live in the united states of america and i have just won thousand dollar from a lottery"*. It is worth mentioning that this step contributes a one-percent improvement in the final model.

3.2 Deep features

After passing each sentence through the preprocessing module, we employed an embedding layer initialized with 300d GloVe vectors trained on Wikipedia 2014 and Gigaword5[3]. For those words that are not in the GloVe's vocabulary, we replaced them with the word *unknown*. We also set the maximum length of each sentence to 30 terms. Longer sentences are truncated and shorter ones are zero-padded. Hence, the output of this layer for each sentence is a matrix of size 30 $\times$ 300. The output of the embedding layer is then fed into a single-layer Bi-directional LSTM [16]. The reason behind using a single LSTM to encode both of the sentences is that the representation becomes symmetric and order-independent. In fact, it yields better results than separate LSTMs for the sentences. Each sentence is represented by extracting the output of the last time-step of Bi-LSTM. The final numerical representation for each pair of sentences is formed by the following equation. It concatenates the element-wise multiplication and the absolute values of their differences.

$$vector = [S_1 \odot S_2; |S_1 - S_2|] \tag{1}$$

Moreover, we conducted experiments on CNN and RNN with GRU cells as well, but the best result was achieved by a single-layer LSTM. So in this research, we only focus on the RNN.

3.3 Hand-crafted features

Even though LSTMs are powerful in terms of learning non-linear boundaries and sequential relationships, they need large amounts of training data to achieve reasonable performance [14]. On the other hand, a dataset such as MSRP [9] is very small. Thus, we extracted the following syntactic and semantic fine-grained features to facilitate the learning process.

3.3.1 Syntactic features

- Jaccard similarity: We computed the Jaccard coefficient [13] for each pair of sentences. Assuming S_i refers to the set of words in the sentence i, it measures the similarity as:

$$similarity = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} \tag{2}$$

[3]https://nlp.stanford.edu/projects/glove/

- Difference in length: This measurement simply computes the absolute difference between the character-length and word-length of the two sentences. We used both of them as two distinct features.
- Longest Common Sub-sequence (LCS): This feature [13] captures the similarity between two sentences based on their longest shared word sequences.
- N-gram similarity: This method [13] measures the similarity based on a sub-sequence of n words from a given sequence of text. The similarity rate is computed by dividing the number of shared n-grams by the union of their n-grams. We used bi-gram and tri-gram features. No improvement was observed by using higher n-grams.
- Fuzzy features: The *fuzzywuzzy* python library[4] extracts four different similarity features based on the Levenshtein distance.[5] We used all of the four features.

3.3.2 New syntactic features

Ratio of stop words We believe that stop words are of important value in conveying semantic information. As an example for their importance, consider the following pair of sentences from MSRP:

- I never organised a youth camp for the diocese of Bendigo.
- I never attended a youth camp organised by that diocese.

The stop words *for* and *by* are obviously critical for catching the semantic differences. The Jaccard similarity for the pair is about 0.4 which grows to 0.55 when the stop words are removed.

By (3), We define two new features to measure the similarity between the sentences, according to their shared stop words. Here, S_1 and S_2 are the sentences (represented as the list of words), SW is the set of stop words, and n_{S_i} indicates the number of stop words in S_i. We set ϵ to a small positive value so that potential division by zeros are omitted.

$$RSW = \frac{|S_1 \cap S_2 \cap SW|}{min(n_{S_1}, n_{S_2}) + \epsilon} \tag{3}$$

Another feature is induced by replacing the *min* with *max* in the equation.

Ratio of non-stop words Much similar to the *ratio of stop words*, this feature measures the similarity of sentences based on their shared *non-stop words*. We consider a word as a non-stop word if it is not listed in the English stop list.[6]

Inverse longest common sub-sequence (ILCS) We introduce this feature to capture the longest phrasal difference between the sentences. To understand the importance of this feature, consider the following pair from MSRP:

- It was a little bit embarrassing the way we played in the first two games, Thomas said.
- We're in the Stanley Cup finals, and it was a little bit embarrassing the way we played in the first two games.

Even though they share many words in common, the sentence *We're in the Stanley Cup finals* makes them semantically unequal and hence, a feature capturing this difference would

[4]https://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/

[5]https://en.wikipedia.org/wiki/Levenshtein_distance

[6]https://gist.github.com/sebleier/554280

be helpful. Algorithm 1 shows how we generate this feature. We slide a window of size at least 10 words over the first sentence and each time, we compute the word-level LCS between the window and the second sentence. Then we use the minimum of all the values as the ILCS feature. The stride to move the window is one word and the size of the window is increased by one after it reaches the end of the sentence. Since the feature is not symmetric, we compute it again by switching the sentences and use it as another feature.

Algorithm 1 generation of ILCS feature.

```
 1: procedure ILCS($words_1$, $words_2$)                    ▷ a list of words for each sentence
 2:     $portion \leftarrow string$
 3:     $ratio, bestratio \leftarrow float$
 4:     $n_1 \leftarrow length(words_1)$
 5:     for <k=10 to $n_1$> do
 6:         for <i=0 to $n_1 - k$> do
 7:             $portion \leftarrow words_1[i : i + k]$              ▷ take a portion of size k
 8:             $ratio \leftarrow LCS(portion, words2)/k$  ▷ normalize the value by portion size
 9:             if $ratio < bestratio$ then                  ▷ find the minimum LCS
10:                 $ratio \leftarrow bestratio$
11:     $output \leftarrow bestratio$
```

3.3.3 Semantic features

Cosine similarity of the embeddings Even though averaging the word vectors is not the best idea to encode a sentence, it can capture useful semantic information and works well in many tasks [20]. Thus, we averaged all the GloVe word-vectors in each sentence and then we computed the cosine similarity between the respective vectors. We repeated the embedding using Word2vec [27] and used it as another feature.

WordNet features WordNet[7] is a lexical database of the English language which clusters the words based on their semantic relationships. It is mainly used for extracting synonyms and measuring the semantic similarities of words. We extracted three different features namely *PATH*, *WUP* and *LCH* [30] for each pair of sentences and used them as distinct features.

3.4 Final model

Figure 1 displays the final model. The handcrafted and deep features are concatenated at the end of the model to constitute the final aggregated feature vector. Among different combination functions (e.g. product and sum) we found that concatenation achieves better results. A fully connected layer followed by a *softmax* function performs the classification task. Activation function for all the fully connected layers is ReLU [34] and batch-normalization [17] is adopted in each of the layers. We also used dropout technique [39] to prevent overfitting and help the model to generalize better. The whole network is trained in an end-to-end manner.

Our evaluation metric to select the best model was accuracy on the validation data with early stopping at five epochs tolerance. The *binary cross-entropy* loss was used to train the

[7]https://wordnet.princeton.edu/

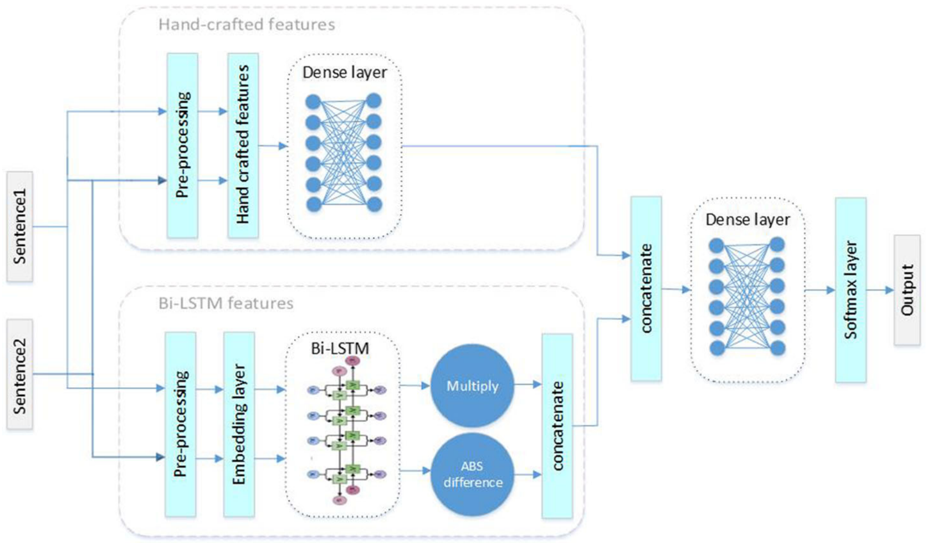

Fig. 1 Model architecture, handcrafted features are augmented with deep learning based features

model with *nAdam* optimizer with the learning rate of *0.001*. Additionally, we employed a two-layer Bi-directional LSTM to see if higher layers could catch the semantic information better.

4 Training and evaluation

In this section, we elaborate on the datasets used in our experiments, and also the way we trained and evaluated our model.

4.1 Datasets

We used two datasets in this research. The first one is the well-known Microsoft Research Paraphrase Corpus [9] and the second one is Quora dataset[8] which has been introduced in Kaggle competitions. The MSRP dataset, which has been extracted from news sources on the web, contains 4076 pairs of sentences for training and 1725 for evaluation, each with a binary label. The distribution of the classes is not equal and 67% of the samples belong to the positive class. The Quora dataset has a similar structure and contains 404301 pairs of sentences for training and more than 2 million for evaluation. It is obtained from questions of different topics asked in Quora website. Samples with negative labels constitute 63% of the training data in this dataset.

4.2 Evaluation metrics

Our evaluation metrics are f-measure and accuracy as they are commonly used for this task [19, 42, 43]. Since the labels of Quora's test set are not publicly available, the only

[8]https://www.kaggle.com/c/quora-question-pairs

metric we could use for the test set was the log-loss reported by Kaggle. Moreover, we made a submission to GLUE benchmark[9] which is a new benchmark for NLU tasks. This benchmark has sampled 360k, 40k, and, 400k of the Quora dataset, respectively for train, validation, and, test sets. for the MSRP, it uses the original test and train set and reports accuracy and f-measures for both datasets.

4.3 Training

We trained our model on both Quora and MSRP. Training on MSRP consists of two parts. In the first one, we trained various classifiers on each set of the handcrafted features to observe the contribution of each of them. We used numerous classifiers namely, MLP [22], AdaBoost [12], XGboost [4], Logistic Regression [29], Random Forest [3], Naive Bayes [35], SVM(rbf kernel) [6], and KNN [22]. For the second part, we divided the training data into 10 folds and trained 10 Bi-LSTM models (with and without handcrafted features combined, see Tables 2 and 4) with the same structure for 15 epochs on nine folds and evaluated it on the remaining fold. Consequently, after the training was finished, we had the 10 best models each of which evaluated on different parts of the training data. Labels of test samples are estimated by averaging the probability predicted by all of the best models.

For the Quora dataset, we trained a single model with and without handcrafted features and made a submission using the best model to both GLUE benchmark and Quora Question-pairs competition. Since this dataset is much bigger compared to MSRP, we ignored WorldNet and ILCS features as they are very time-consuming.

5 Experimental results

Evaluation results of the first part of the training on MSRP are shown in Table 1. The results confirm that adding each set of features improves both accuracy and f-measure in most of the cases. The effect of our features is obvious, especially when using logistic regression which increases the accuracy by absolute two percent. The most important finding is that our features achieved the best relative results with classifiers which have a poor performance when using solely semantic or syntactic features. For example, the accuracy of Naive-Bayes classifier and SVM trained on semantic plus syntactic features is very low but when they are trained with our features, they obtain the best relative accuracy. Therefore, we can conclude that our features convey supportive discriminant information and induce a different perspective on the data.

Table 2 shows the evaluation results of the second part of the training on MSRP. Bi-LSTMSs plus GloVe and the handcrafted features (Bi-LSTM +All) outperformed the other settings. both accuracy and f-measure got worse when we added another Bi-LSTM layer. This model has more parameters and is likely to overfit faster than simpler models, which might be due to the small size of MSRP. Additionally, the results of the best architecture with GRU cells are reported which confirm the superiority of LSTM over GRU on this task. Table 3 shows the evaluation result of our model in comparison with other works on MSRP. Our best model outperforms most of the previous works. The result of GLUE baseline and our model are in bold in this table.

[9]https://gluebenchmark.com

Table 1 Results on MSRP with different types of handcrafted features

Features type	Accuracy							
	Adaboost	MLP	Logistic regression	Random forest	XGboost	Naive bayes	SVM-RBF	Knn
Semantic	70.9	74.2	73.1	73.9	72.2	69.2	72.6	65.2
Syntactic	74.9	77.6	76.2	74.9	74.7	67.7	68.4	68.2
Our features	73.4	75.1	75	74.3	73.8	71.4	74.6	68.2
Syntactic + Semantic	75.2	78	76.6	74.8	76.3	69.1	68.6	68.3
Syntactic + Semantic+ Our features	75.9	78.1	78.6	76.7	76.8	69.2	69	68.3
F-measure								
Semantic	79.7	81.7	81.4	81.9	80.8	76.9	81.8	72.3
Syntactic	82	83.6	83	82.3	82	73	79.1	74.5
Our features	81.1	82.6	82.6	82.3	81.5	78.1	82.8	75.1
Syntactic + Semantic	82	84.1	83.2	82.3	83.1	75	78	74.7
Syntactic + Semantic+ Our features	82.5	84.2	84.8	83.8	83.4	75	77.7	74.7

Table 2 Results on the MSRP test data for different models

	LSTM+ GloVe	Bi-LSTM+ GloVe	Bi-LSTM+ All	Two-layer Bi-LSTM+All	Bi-GRU+ ALL
Accuracy	71.9	73.4	79.2	78.2	79
F-measure	81.8	82.6	85.4	84.2	84.2

All refers to all of the handcrafted features and initialization of words with GloVe

Table 4 shows the evaluation results on Quora dataset. Similar to MSRP, a bi-directional LSTM combined with the handcrafted features outperforms other models. Adding another LSTM layer or employing GRU cells instead of LSTM, both degrade the performance. However, the drop in accuracy (when 2-layer-LSTM is used) is less compared to that on MSRP which indicates the necessity of more training data for deeper models. We believe more training data is required to see an improvement over one-layer LSTM in this case. The effect of the handcrafted features is obvious. They improved both f-measure and accuracy by more than two percent when combined. The test result according to the GLUE benchmark for our best model is also shown. We made a submission to Quora-Question-pairs competition and achieved a log-loss of 0.27 which is shown the table. The performance of our best

Table 3 Comparison of our model with related works on MSRP

Method	Year	Accuracy	F-measure
Sjöblom et al.	2018	69.5	80.6
Mihalcea et al.	2006	70.3	80.3
Islam et al.	2009	72.6	82.3
Milajevs et al.	2014	73	82
Blacoe and Lapata	2012	73	82.3
Agarwal et al.	2018	73	83.3
Fernando and Stevenson	2008	74.1	82.4
Qayyum et al.	2012	74.7	81.8
Finch et al.	2005	75	82
Kenter et al.	2015	76.6	84
Radford et al.	2018	75.7	82.3
Kozareva et al.	2006	76.64	79.57
Socher et al.	2011	76.8	83.6
logeswaran et al.	2018	76.9	84
GLUE baseline	2018	77.3	83.5
Madnani et al.	2012	77.4	84.1
Zhang et al.	2017	77.5	84.5
Wang et al.	2016	78.4	84.7
He et al.	2015	78.6	84.73
Cheng and Kartsaklis	2015	78.6	85.3
Proposed method		79.2	85.4
Ji and Eisenstein	2013	80.4	85.9

Table 4 Results on the Quora for different models

	LSTM + GloVe	Bi-LSTM + GloVe	Bi-LSTM + All	Two-layer Bi-LSTM + All	Bi-GRU+ ALL
Validation Acc, f1	81.94, 75.8	82.5, 76	84.6, 79.6	84.4, 78.6	84.4, 78.9
Test Acc, f1 (GLUE)	-	-	83.7, 59	-	-
Log loss(Kaggle),-	-	-	0.27	-	-

All refers to all the handcrafted features and initialization of words with GloVe.

model in comparison with other works on the Quora dataset, according to the GLUE benchmark, is shown in Table 5. It achieves comparable results among different algorithms. We can observe that a deep model such as Radford et al. [33] perform rather poorly on small corpus such as MSRP but it achieves the best performance on this dataset where a huge amount of training data is available. Our model, on the other hand, shows great performance on small corpora and comparable results on large datasets.

To investigate the performance of our model, we have provided the confusion matrix on the evaluation set for both datasets. Figure 2 shows the confusion matrices. As it is evident, on MSRP, our model suffers from a tendency toward the dominant class, probably due to the small size of training data. In the case of Quora, where the sufficient training data is available, the bias toward the dominant class is significantly reduced despite the unequal distribution of the classes. Moreover, we discovered that 47% of mis-classified samples in MSRP contain numerical symbols. Hence, defining a specific feature to catch the numerical information, would be promising idea. Unlike MSRP, this number is only

Table 5 Comparison of our model with related works on Quora according to the GLUE benchmark

Method	Accuracy	F-measure
BiLSTM+ELMo	78	57.8
CBOW	79.1	51.4
BiLSTM	80.2	59.1
InferSent	81.7	59.1
Single Task BiLSTM	81.7	61.4
BiLSTM+CoVe	82	59.1
Skip-Thought	82.2	56.4
DisSent	82.6	59.5
GenSen	82.9	59.8
Single Task BiLSTM+CoVe	83.3	59.4
GLUE Baseline	83.5	63.3
Single Task BiLSTM+Attn	83.5	62.9
Proposed method	83.7	59
Single Task BiLSTM+CoVe+Attn	84.1	60.1
Samuel Bowman(2018)	84.7	64.8
BiLSTM+ELMo+Attn	85.1	64.6
Single Task BiLSTM+ELMo+Attn	86.5	66.1
Alec Radford et al.(2018)	88.5	70.3

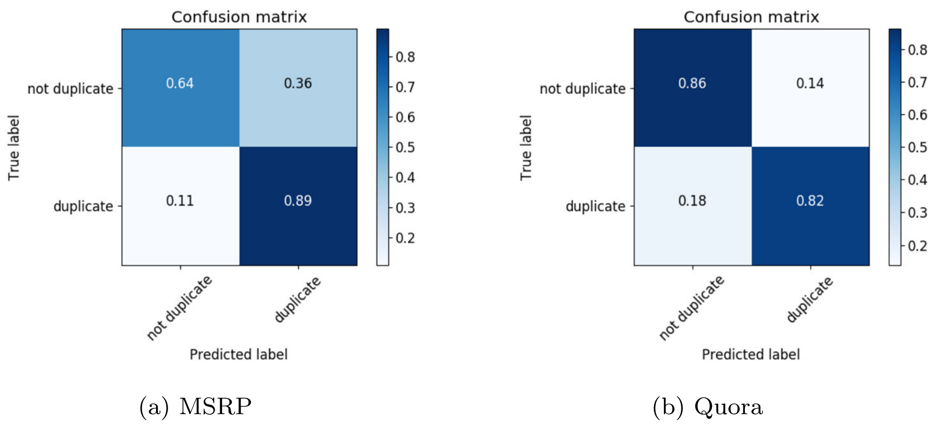

Fig. 2 Confusion matrices of validation data per dataset. **a**:MSRP, **b**:Quora

10% for Quora which indicates that our model can interpret the numerical differences when adequate training data is available.

6 Conclusion and future works

We proposed a new method using bi-directional LSTMs combined with handcrafted features and evaluated our model on MSRP and Quora datasets. Despite the small size of the MSRP corpus, our model achieved a great performance. The effect of handcrafted features is remarkable, especially when used in a small dataset like MSRP. These features also improved the results on the Quora dataset but the improvement is less compared to that on MSRP, since LSTMs can generalize well on large data. Our model also achieved competitive results on the Quora dataset according to the GLUE benchmark and the QQP competition on Kaggle.

In the future we will employ the result of our error analysis and extend this line by analyzing the contribution of handcrafted features to different properties of the sentences (e.g. length and overlap ratio). We will also exploit the contribution of additional word embeddings such as ELMo [32] and use the state of the art models such as BERT [8] and attention techniques [41] which has been gaining attention recently.

Compliance with Ethical Standards

Conflict of interests The authors declare that there is no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Agarwal B, Ramampiaro H, Langseth H, Ruocco M (2018) A deep network model for paraphrase detection in short text messages. Information Processing & Management 54(6):922–937

2. Blacoe W, Lapata M (2012) A comparison of vector-based representations for semantic composition. In: Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning, pp 546–556. Association for Computational Linguistics
3. Breiman L (2001) Random forests. Machine learning 45(1):5–32
4. Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp 785–794. ACM
5. Cheng J, Kartsaklis D (2015) Syntax-aware multi-sense word embeddings for deep compositional models of meaning. arXiv:1508.02354
6. Cortes C, Vapnik V (1995) Support-vector networks. Machine learning 20(3):273–297
7. Creutz M (2018) Open subtitles paraphrase corpus for six languages. arXiv:1809.06142
8. Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805
9. Dolan B, Quirk C, Brockett C (2004) Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In: Proceedings of the 20th international conference on Computational Linguistics, p 350. Association for Computational Linguistics
10. Fernando S, Stevenson M (2008) A semantic similarity approach to paraphrase detection. In: Proceedings of the 11th annual research colloquium of the uk special interest group for computational linguistics, pp 45–52
11. Finch A, Hwang Y-S, Sumita E (2005) Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In: Proceedings of the third international workshop on paraphrasing (IWP2005)
12. Freund Y, Schapire R, Abe N (1999) A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence 14(771-780):1612
13. Gomaa WH, Fahmy AA (2013) A survey of text similarity approaches. Int J Comput Appl 68(13):13–18
14. Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J (2016) Lstm: A search space odyssey. IEEE transactions on neural networks and learning systems 28(10):2222–2232
15. He H, Gimpel K, Lin J (2015) Multi-perspective sentence similarity modeling with convolutional neural networks. In: Proceedings of the 2015 conference on empirical methods in natural language processing, pp 1576–1586
16. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural computation 9(8):1735–1780
17. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167
18. Islam A, Inkpen D (2009) Semantic similarity of short texts. Recent Advances in Natural Language Processing V 309:227–236
19. Ji Y, Eisenstein J (2013) Discriminative improvements to distributional sentence similarity. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp 891–896
20. Kenter T, Borisov A, De Rijke M (2016) Siamese cbow: Optimizing word embeddings for sentence representations. arXiv:1606.04640
21. Kenter T, De Rijke M (2015) Short text similarity with word embeddings. In: Proceedings of the 24th ACM international on conference on information and knowledge management, pp 1411–1420. ACM
22. Kotsiantis SB, Zaharakis I, Pintelas P (2007) Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering 160:3–24
23. Kozareva Z, Montoyo A (2006) Paraphrase identification on the basis of supervised machine learning techniques. In: International conference on natural language processing (in Finland), pp 524–533. Springer
24. Logeswaran L, Lee H (2018) An efficient framework for learning sentence representations. arXiv:1803.02893
25. Madnani N, Tetreault J, Chodorow M (2012) Re-examining machine translation metrics for paraphrase identification. In: Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies, pp 182–190. Association for Computational Linguistics
26. Mihalcea R, Corley C, Strapparava C et al (2006) Corpus-based and knowledge-based measures of text semantic similarity. In: Aaai, vol 6, pp 775–780
27. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv:1301.3781
28. Milajevs D, Kartsaklis D, Sadrzadeh M, Purver M (2014) Evaluating neural word representations in tensor-based compositional settings. arXiv:1408.6179
29. Nasrabadi NM (2007) Pattern recognition and machine learning. Journal of electronic imaging 16(4):049901

30. Pedersen T, Patwardhan S, Michelizzi J (2004) Wordnet:: Similarity: measuring the relatedness of concepts. In: Demonstration papers at HLT-NAACL 2004, pp 38–41. Association for Computational Linguistics
31. Pennington J, Socher R, Manning C (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
32. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. arXiv:1802.05365
33. Radford A, Narasimhan K, Salimans T, Sutskever I (2018) Improving language understanding by generative pre-training. https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf
34. Ramachandran P, Zoph B, Le QV (2018) Searching for activation functions.(2018). arXiv:1710.05941
35. Rish I et al (2001) An empirical study of the naive bayes classifier. In: IJCAI 2001 workshop on empirical methods in artificial intelligence, vol 3, pp 41–46
36. Shahmohammadi H, Dezfoulian M, Mansoorizadeh M (2018) An extensive comparison of feature extraction methods for paraphrase detection. In: 2018 8th International Conference on Computer and Knowledge Engineering (ICCKE), pp 47–51. IEEE
37. Sjöblom E, Creutz M, Aulamo M (2018) Paraphrase detection on noisy subtitles in six languages. arXiv:1809.07978
38. Socher R, Huang EH, Pennin J, Manning CD, Ng AY (2011) Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In: Advances in neural information processing systems, pp 801–809
39. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research 15(1):1929–1958
40. Ul-Qayyum Z, Altaf W (2012) Paraphrase identification using semantic heuristic features. Res J Appl Sci Eng Technol 4(22):4894–4904
41. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems, pp 5998–6008
42. Wang A, Singh A, Michael J, Hill F, Levy O, Bowman SR (2018) Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv:1804.07461
43. Wang Z, Mi H, Ittycheriah A (2016) Sentence similarity learning by lexical decomposition and composition. arXiv:1602.07019
44. Zhang X, Rong W, Liu J, Tian C, Xiong Z (2017) Convolution neural network based syntactic and semantic aware paraphrase identification. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp 2158–2163. IEEE
45. Zhu Y, Kiros R, Zemel R, Salakhutdinov R, Urtasun R, Torralba A, Fidler S (2015) Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In: Proceedings of the IEEE international conference on computer vision, pp 19–27