ITC 4/49
Information Technology and Control
Vol. 49 / No. 4 / 2020
pp. 495-510
DOI 10.5755/j01.itc.49.4.27118

Deep Learning Based Semantic Similarity Detection Using Text Data

Received 2020/07/04 | Accepted after revision 2020/08/08

http://dx.doi.org/10.5755/j01.itc.49.4.27118

HOW TO CITE: Mansoor, M., Rehman, Z. U., Shaheen, M., Khan, M. A., Habib, M. Deep Learning Based Semantic Similarity Detection Using Text Data. *Information Technology and Control*, 49(4), 495-510. https://doi.org/10.5755/j01.itc.49.4.27118

Deep Learning Based Semantic Similarity Detection Using Text Data

Muhammad Mansoor, Zahoor Ur Rehman
Computer Science Department, COMSATS University Islamabad, Attock Campus, Pakistan

Muhammad Shaheen
Faculty of Engineering & IT, Foundation University Islamabad, Pakistan

Muhammad Attique Khan
Department of Computer Science, HITEC University Taxila, Pakistan

Mohamed Habib
College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia
Faculty of Engineering, Port Said University, Port Fuad City, Egypt

Corresponding author: attique@ciitwah.edu.pk

Similarity detection in the text is the main task for a number of Natural Language Processing (NLP) applications. As textual data are comparatively large in quantity and in volume than the numeric data, measuring textual similarity is one of the important problems. Most of the similarity detection algorithms are based upon word to word matching, sentence/paragraph matching, and matching of the whole document. In this research, a novel approach is proposed using deep learning models, combining Long Short-Term Memory Network (LSTM) with Convolutional Neural Network (CNN) for measuring semantics similarity between two questions. The proposed model takes sentence pairs as input to measure the similarity between them. The model is tested on publicly available Quora's dataset. In comparison to the existing techniques gave 87.50 % accuracy which is better than the previous approaches.

KEYWORDS: Deep Learning, Semantics, Similarity, Quora, question duplication, LSTM and CNN.

1. Introduction

In this digital era, the use of web has increased due to the advancement of the internet. A huge number of user-generated short text comes through the Internet as well as social networks, e.g. user opinions about the product, blogs, and services. Such short text is basically subjective and semantics oriented. Detecting similarity is the problem of finding the score to which the input pair of texts has similar content. Words are similar either lexically, in which words have a similar sequence of characters, or semantically, in which words are used in the same context or have the same meaning. Measuring the text similarity between words, sentences, etc. plays a vital role in information retrieval, clustering documents, short answer grading, auto essay scoring, text summarization, and machine translation. In terms of research, it is valuable to detect and match the short text at different granularities.

The volume of the data obtainable in the form of text is increasing rapidly. Therefore, the importance of applications of NLP is increased for improving the progress of the analysis and retrieval of data available in the form of text. The computation of semantic similarity among the sentences is the main task in many NLP applications like text classification and summarization [24], topic extraction [45], information retrieval [39] and for the grading of short answers generated automatically, finding duplicate questions [36], document clustering [22], etc. Different methods are available to measure semantic similarity among sentences. Most of the methods mainly depend on the features extracted by using different techniques. These features are then given to the machine learning algorithms to find similarity scores between the sentences. However, their accuracy still needs to improve.

The similarity between the texts of varying length produces better results when the length of the sentences is minimal. The techniques of similarity detection are based on capturing strong relationships in two sentences. For example two sentences from the paper [35] are given below:

- In jewelry, a gem is a stone that is used.
- In the rings and necklaces, a jewel is a precious stone.

The above sentences are about jewels and almost a similar concept is explained in two different ways which can easily be interpreted by humans.

Computing the text similarity between two texts is very difficult for the machine due to huge variance in natural languages. The authors in [14] presented a conceptual space framework, which permits the model to profess similarity [53] which "changes with changes in selective attention to specific perceptual properties." Meanwhile, the short text has its own sole characteristics. To classify short text is a difficult task since it provides many challenges related to the standard accuracy score. Due to the challenges and need, text classification has high demand. Recently, deep learning shows a huge advancement in many machine learning research areas such as text/signature [6], object classification [21, 47], visual surveillance [26, 51], biometrics [3], medical imaging [27, 29, 42, 48, 52], and many more [4, 25, 28, 30-32, 41]. In this research, a deep learning model is proposed for detecting similarity between short sentences such as question pairs. Contributions of the paper are the following:

1. A deep learning model is developed by using LSTM and CNN models to detect semantic similarity among short text pairs, specifically Quora question pairs.
2. To utilize the proposed method for finding similar questions, and validating effectiveness of method in the detection of similar questions.
3. Shown a wide range of comparative studies for semantic similarity detection problem.

The rest of the paper is organized as follows. In Section 2, we discussed the related work comprising deep learning and semantic similarity. In Section 3, we proposed our model in detail. The experimental setup is described in Section 4. Finally, Section 5 will give the conclusion.

1.1. Problem Statement

Online question-answering is gaining popularity in different IT based web applications. Quora is an online platform for people to ask questions and connect with others who give quality answers. Every month, around 100 million people visit Quora and post their

questions, so there are many chances that people may ask similar or existing questions. It will be beneficial for both users and community to detect similar questions and to extract the relevant answers in less time. This research is focused to find duplicate questions to improve the performance and efficiency of the questions asked on Quora community. Let there are two questions denoted by Q1 and Q2 respectively. These two questions are semantically equivalent or duplicate of each other if they convey the same meaning, otherwise non-duplicate.

The problems addressed in this research are formulated as questions below:

Q1: What is the significance of similarity detection in Textual Data?

Q2: How to detect duplicate question pairs?

Q3: How the efficiency of the existing algorithms can be enhanced?

2. Related Work

Semantic similarity detection is a critical task for most of the NLP applications such as question and answering, detecting paraphrases and IR i.e. Information Retrieval. Many approaches are proposed for the calculation of semantics similarity between sentences. Semantic similarity is the problem of identifying how similar the given pair of texts are.

2.1. Semantic Similarity Using Deep Learning

In NLP tasks, deep learning played an important role in the past few years. Many researchers used a variety of features for paraphrase identification tasks like n gram features [40], syntactic features [10], linguistic features [49], etc. The use of deep learning methods have moved researchers' consideration towards semantic representation of text [2], [8]. The use of CNN for learning sentence representations achieved remarkable improvement in the results of classification. Recurrent Neural Network (RNN) that can learn the relationship among different elements in input sequence were used by researchers to represent text features [34]. The authors in [33] gave a simple model that relied upon character level inputs, however, for predictions they used word level inputs. Although they utilized few parameters, their system outperformed some benchmark scores that employ word level embedding. In addition, the authors [11] proposed a CNN architecture for analyzing sentiments among short text pairs. Their proposed model use both networks and combined CNN and RNN for sentiment analysis [55].

2.2. Short Text Similarity

The authors in [59] proposed a MULTIP model to detect paraphrase on twitter's short text messages. They modelled paraphrase relations for both sentences and words. In research conducted in [12], the overlapping in unigram and bigram features is evaluated for paraphrase detection. The work did not consider the semantic similarity score. F1 score for paraphrase detection in the said study was 0.674. In [58], the authors proposed a new recurrent model named Gated Recurrent Averaging Network (GRAN) which is based on AVG and LSTM and gave better results than both of them. They [54] proposed simple features to execute PI and STS tasks on Twitter dataset. Based on the experiments, they concluded that by overlapping word/n gram, word alignment by METEOR, scores of BLEU and Edit Distance, one can find semantic information of Twitter dataset at a low cost. In [20], the authors presented a novel deep learning model for detecting paraphrases and semantics similarity among short texts like tweets. They also studied how different levels (character, word and sentence) of features could be extracted for modeling sentence representation. Character level CNN is used for finding character n gram features, word level CNN for word n gram features and LSTM for finding sentence level features. The model was tested on twitter dataset, and the results showed that character level features play an important role in finding similarity between tweets. They also concluded that the combination of two models gave better results than that of individual ones.

2.3. Convolutional and Recurrent Neural Network

The authors in [9] proposed an ensemble method that represents text features in an efficient way. The experimental results showed that the accuracy of the proposed approach largely depends on the size of the dataset. When the dataset is small, the system is vulnerable to overfitting. However, training on large

dataset gave good results. In [57], a model based on similarities and dissimilarities in the sentences is proposed. The model denoted each word as a single vector and calculated matching vector with respect to every word in another sentence and based on this matching score, the word vectors are separated into similar and dissimilar component matrix. A two-channel CNN model is then applied to capture feature vectors from the component matrix. The model was applied on answer selection problem and showed good performance. In [38], an interpretability layer called iSTS is added which depends on arrangement of sentences among pairs of segments. The authors in [61] proposed an attention-based CNN for modeling sentence pairs for different NLP tasks and found that attention based CNN outperformed the simple CNNs. A new model based on deep learning named as BiMPM was proposed [56]. In this model, for a given pair of sentence segments, the model first encoded the sentences by using BiMPM and then matched encoded sentences in both directions. In each matching direction, BiMPM compared sentence segments from different perspectives. The BiLSTM layer calculated the matching score which is then used for making the final decision. In [50], deep learning techniques to rerank short text pairs are proposed. They used CNN to learn sentence representation and build similarity matrix. The experimental results showed that deep learning models greatly improved results.

In [19], a deep neural model is proposed for modelling sentences hierarchically. The proposed model achieved better results for sentence completion and response matching task. The model did not perform well on paraphrase detection task. In [18], a Siamese gated neural network is proposed to find similar questions on Quora. The authors tried to detect semantically similar questions on online user forum and used the data from stack exchange forum [7]. They presented a CNN-based approach which generated vector representation of given question pairs and scored them using similarity matrix. The proposed CNN-based method was tested on data from stack exchange forums. The results of the study were compared with the standard machine learning algorithms like support vector machines. Domain word embedding are evaluated with Wikipedia word embedding and the accuracy of CNN method was high. In [1], the authors attempted to detect paraphrases in short text using deep learning techniques. Some of the existing approaches of finding paraphrases gave good results on clean text but are failed to perform well on noisy and short texts. Therefore, a new, generic and robust architecture which they called Deep Paraphrase is proposed which combined CNN and RNN models. The experiments are conducted on Twitter dataset and results showed that the proposed architecture performed well on both types of texts. In [43], semantic textual similarity in paraphrases of Arabic tweets is found by applying several text processing phases and extracting different features (Lexical, Syntactic and Semantic) to overcome the limitations of existing approaches. Machine learning classifiers like support vector regressions were trained by using these extracted features. The accuracy of the model in PI and STS tasks was high. In [15], the authors proposed Siamese Gated units of machine learning algorithms like support vector machines, adaboost and random forest for predicting similar questions on Quora dataset, they did not calculate accuracy but they calculated cross entropy loss. In [5] the authors used Attentive Siamese LSTM for calculating semantic similarity among sentences. A different architecture having bi- directional LSTM network with attention mechanism has been proposed [37]. They performed their experiments on six different datasets for the classification of sentiments and one for question classification and proved the model to be more accurate. In [46], authors proposed deep learning architecture, which used attention mechanisms. In their method, they decomposed the large problem into smaller subproblems that can be easily solvable separately. Their proposed method gave better accuracy and outperformed the complex deep neural models. A summary of the existing approaches is shown in Table 1.

The literature revealed that a lot of methods have been proposed for detecting similarity between short sentences, but these methods have some problems that need to be addressed. Different researchers applied different techniques to find similarity scores and tried to improve the accuracy. In this paper, our focus is to use deep learning models and develop a robust similarity detection system for two sentences. The model only requires a pair of sentences and finds the similarity between them. For this purpose, a combination of CNN and LSTM is used to find similarity between such sentence pairs.

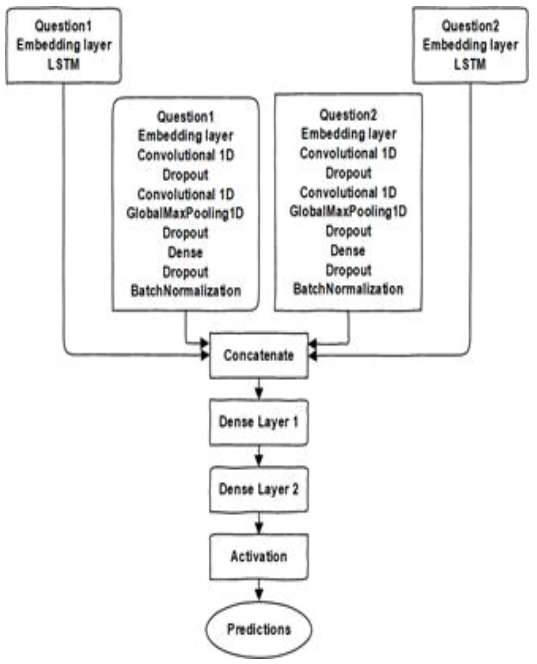
Table 1
Overview of Deep Learning Approaches and their Features

Work	Method	Features	Dataset
Mohammad, A.-S. et al. [43]	Multi-layer Neural Network	POS Tagger and Pre trained NN	MSPR, Twitter
Agarwal, B. et al. [1]	Hybrid model (deep learning and statistical features)	Pre trained word embedding and POS Tagger	MSPR
Huang, J., et al. [20]	Multi-layer Perceptron	Pre-trained word embedding	Twitter
Yin, W. et al. [61]	Logistic Regression	Word2vec based Embedding	MSPR
Godbole, A. et al. [15]	Gated Recurrent Unit (GRU) Network	GloVe vectors pre-trained on Wikipedia	Quora Dataset
Severyn, A. and A. Moschitti. [50]	Convolutional neural network	Word embedding	TREC: QA and Microblog Retrieval
Chen, G. et al. [9]	CNN-RNN model	word2vec embedding	Reuters-21578 and RCV1-v2.
Bogdanova, D. et al. [7]	Convolutional neural network	Skip gram neural network architecture	Quora Dataset
Homma, Y. et al. [18]	Siamese Gated Recurrent Network	300-dimensional GloVe vectors pre-trained on Wikipedia	Quora Dataset
Wang, Z. et al. [57]	Two-channel CNN model	Pre-trained word embedding by Mikolov et al. (2013)	QASent, WikiQA
Parikh, A.P. et al. [46]	Simple Attention-based model	300-dimensional GloVe vectors	SNLI
Bao, W. et al. [5]	Attentive Siamese LSTM	300D word embedding (English and Chinese)	SemEval 2014
Liu, G. and J. Guo [37]	AC-BiLSTM	Word embedding	TREC, SST1, SST2, IMDB etc.
Deep Similarity Model	LSTM and CNN	Google Pretrained word embedding	Quora Dataset

3. Proposed Work

In the field of text similarity detection, techniques based on deep learning moved researchers towards semantically distributed representations [2], [5], [37], [13], [23], [60], [17]. To find the semantic relatedness among question pairs, we proposed a technique based on deep learning. The goal is to find similar questions with reduced time and complexity. The method based on deep learning will require minimal preprocessing and will take a pair of sentences only hence reducing the time complexity. Figure 1 shows the diagram for our proposed model.

Figure 1
Proposed flow diagram for similarity detection



3.1. Pre-Processing

Cleansing Text: Preprocessing is one of the essential steps, especially in text mining. Regex plays an important role during the whole process. Regex is used to find interesting patterns in raw text, and then perform the desired action. For our task, 200000 unique words were used in the whole process due to computational expenses.

In the first step (as shown in Algorithm 1), all the questions are tokenized and converted into sequences. A threshold of no more than 30 words per question is applied. Stop words, punctuation and noise are removed from the text by using NLTK, and the words are converted into lower case. Regex function is applied to convert words into the original form. Stemming is performed through Snowball Stemmer[1] to remove the unnecessary portion of the text. A few examples of regex are given in Table 2.

1 http://www.nltk.org/howto/stem.html

Table 2
Regex examples

Original	Converted	Original	Converted
what's	what is	'll	will
've	have	0s	0
can't	cannot	e - mail	email
'nt	not	u s	American
i'm	i am	're	are

Algorithm 1: Process texts in datasets

Input: Text
Output: List of Words
Notation: T: text, S: stops, SW: stopwords, STW: stem words, st: stemmer, stw: stemmed_words
Initialisation : T, S, SW, STW, st, stw

```
T← T.lower().split()
if SW removes
    S ← set SW to English
    for w in T do
        if not w
            stops
        end if
    end for
    T ← w
end if
T ← join T
Clean the text using Regex
if STW
    T ← split T
    st ← english
    stw ← stw in T
    T ← join stw
end if
return T
```

Algorithm 2: Data Processing

Input: Questions

Output: Labels

Notation: t1: texts_1, t2: texts_2, lb: labels, ti: test_ids, rd: reader, hd: header, tt1: test_texts_1, tt2: test_texts_2, qd: q_dict, tsf: train_feat, tsf: test_feat

Initialisation : t1, t2, lb, ti, rd, hd, tt1, tt2

```
with open.TDF as f
    rd ← read csv
    hd ← next of rd
    for v in rd
            append tt1, tt2, lb
    end for
end with
with open.TDF as f
    rd ← read csv
    hd ← next of rd
    for v in rd
            append tt1, tt2, ti
    end for
end with
qd ← default dictionary
for i in questions
    qd ← concatenate questions
end for
trf ← intersect questions
tsf ← intersect questions
```

Algorithm 2 shows the process of collecting all the vocabulary words and making a dictionary of words. First, the training file is opened. All the questions in training file are being concatenated. After concatenation, the dictionary of words is formed using this text.

3.2. Tokenization and Sequence Creation

Tokenization simply divides a sentence into a list of words. We used Keras tokenizer function to tokenize the strings and the use of another important function 'texts_to_sequences' to make sequences of the words. After converting texts to the sequences, the sequences are padded to the desired length which is done by using Max Length argument (in this case max length is 30). The input sequences are padded that are smaller than the desired length while the longer ones are truncated.

Algorithm 3: Converting Text to Sequences

Input: Text

Output: Number Sequence

Notation: token: tokenizer, seq: sequences, tsseq: test_sequences, wi: word_index

Initialisation: token, seq, tsseq, wi

```
token ← tokenize words
seq ← tokenize words to sequences
tsseq ← tokenize words to test sequences
wi ← token.wi
```

3.3. Word Embedding

Word embedding is used to convert words which are then assigned to real valued vectors (known as embedding) to retain different types of information about the words. They are used to represent the words in an N-dimensional vector space. Word embedding is of great importance as this embedding denotes the meaning of the words, i.e. how words are semantically identical. Google's[2] word2vec representation of text is used in our study. The main advantage of using Google's word2vec is that it represents a word as a vector having 300 dimensions. They are used to learn word embedding for all the words in the dataset.

Algorithm 4: Embedding Matrix Preparation

Input: Text

Output: Vectors

Notation: nbw: nb_words, em: embedding_matrix, w: word, wi: word_index, voc: vocab

Initialisation: nbw, em, w, wi, voc

```
nbw ← storing maximum number of words
em ← embed with zeros
for w, i in wi
    if w in voc
            em[i] ← words vectors
    end if
end for
```

2 http://mccormickml.com/2016/04/12/googles-pre-trained-word2vec-model-in-python/

3.4. Recurrent Neural Network

RNNs are used for learning the objects which are occurring repeatedly, e.g. time series, repeating words, etc. Long Short-Term Memory networks (LSTM) are explicit types of RNNs that can learn the relationship among different elements in an input sequence. In our scenario, these elements are words. Given adequate information, deep neural networks are exhibited to be successful for a wide variety of machine learning tasks, including text learning tasks based on larger datasets. For this task, Google's word2vec is one of the good examples, because it provides an automated power of mean for mining semantic representations from big data [16]. With the use of large-scale text corpus or any other large dataset, word2vec develops a vocabulary of a fixed size and figures out how to depict words outside the vocabulary by building vector portrayals utilizing words from inside the vocabulary [44]. Two popular networks types are CNNs and RNNs, like LSTM network.

3.5. Convolutional Neural Network

Convolutional neural network, also called convnets, is an interesting and important development in the machine learning field. A standard Convolutional Neural Network (CNN) model encompasses single or multiple layers of subsampling and convolution, followed by fully connected layers which can be one or more and an output layer.

3.5.1. Max Pooling Layers

Convolutional networks may utilize max pooling layers. Max pooling is a type of non- linear down sampling which minimizes longitudinal size of conventional layer's output using the subdivision of output in rectangular boxes, which is only the best value for each filter. Additionally, bringing down the quantity of connection, and hence computational cost, max pooling lessens over fitting by making features more spatially free. Max pooling layers can be numbered after every convolutional layer or after some layers.

3.5.2. Dropout Layers

In dropout technique, during the training process randomly selected neurons are ignored. We can say that we "dropped-out" some neurons randomly. In simple words, the contribution of neurons is temporarily removed on forward pass during the activation process of downstream neurons, and no weight updates are applied to neurons on backward pass.

3.5.3. Batch Normalization

The next layer used is batch normalization layer. It is used for normalizing inputs of each layer to overcome covariate shift problem. We do normalization for an input layer by scaling and changing the activations to speed up the learning process. During our training process, the batch normalization[3] we perform the following tasks:

1 First, we calculate mean and variance of input layers

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{Batch Mean} \tag{1}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \quad \text{Batch Variance} \tag{2}$$

2 In the second step, normalization of input layers takes place using previously calculated statistics.

$$\overline{x}_l = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{3}$$

4 Lastly, we shift and scale the input, for obtaining output of the layer.

$$y_i = \sqrt{x_i} + \beta \tag{4}$$

3.5.4. Setting-up Model

A model is proposed in this study for the detection of similar questions on Quora, by using Convolutional Neural Network (CNN) and LSTM. CNNs extract either local or deep features from natural language. Studies have shown that CNN has produced good results in sentence classification. In this work, we have used CNN followed by dropout, dense and max pooling, in combination with LSTM network. The model comprised of one translational layer for each question. Two LSTM layers initialized by word embedding and two CNN which are also initialized by word embedding. The inputs (known as input tensors) from all the layers are concatenated and then passed to the series

3 https://arxiv.org/pdf/1502.03167v3.pdf

of dense layers, followed by activation function to make final predictions. Based on these predictions, accuracy, precision, recall and F1 score was calculated.

4. Experimental Results

In this section, we reported the performance of our proposed deep neural network-based model that we have implemented for detecting similarity among questions. We performed and evaluated multiple experiments shown in Table 3 to reach the best accuracy results and to develop the best fit technique for similarity detection.

We performed our experiments on Quora's dataset, in which duplicate data is labeled as positive samples and non-duplicate data is labeled as negative samples. First, we train our deep learning model on Quora's dataset to detect similarity between short questions. We investigated the best hyperparameters for our model. Our model is tested on testing dataset to calculate evaluation scores and comparing the same with previous models.

4.1. Dataset

Quora made the first dataset public in 2017. This dataset contains pairs of duplicate and non-duplicate questions. Duplicate means that given question pairs have the same meaning. The dataset contained 404290 question pairs having approx. 37% of positive or duplicate samples and 63% negative or non-duplicate samples.

4.2. Implementation

The proposed method was implemented in Keras using TensorFlow backend. The model was trained for 100 epochs, while patience value was set to 5. The process is set to stop if the performance stopped improving. There are total 149262 duplicate pairs and 255028 non-duplicate pairs. The dataset is split into 80/20 (training/validation) featuring 323432 instances for training and 64687 for validation set, which is further split into validation and testing sets having 16171 instances for testing set. The model is evaluated at the end of each epoch. By doing this, the model with the best score on the validation set is used to make predictions for testing data.

4.3. Hyperparameter Setting

In this experiment, we have selected hyperparameters by first investigating training data and searching for the best hyperparameters. We choose optimizer, activation unit, dropout with which our model performed best. The setting of hyperparameters varies from dataset to dataset. We tested different hyperparameters on this dataset. To our knowledge, these selected hyperparameters performed best and gave better results than others. Table 4 shows the hyperparameters setting of values for our model. While designing our network, ReLU activation unit was used, which is widely used as activation unit in many deep learning models. We have applied different optimizers for training our model and evaluated the performances. Figure 2 shows that the nAdam optimizer gave the best accuracy on our dataset. The dropout value was set to 0.2. and applied to every layer. The learning rate hyperparameters was set to 5, which means that the model waited for 5 epochs before reducing the learning rate. Likewise, when investigating the size of training data, we have seen that if the size of the training data becomes less, the accuracy and F1 score slightly dropped.

4.4. Evaluation Metrics

For the evaluation of our proposed model, we have used standard evaluation metrics that are widely used.

Precision: Precision indicates the proportion of predicted positive cases that are real positives.

$$\text{Precision} = \frac{TP}{TP + TN} \tag{5}$$

Recall: Recall is the proportion of actual positive cases that were correctly predicted.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{6}$$

F1 score: The harmonic mean between precision and recall.

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

Accuracy: Accuracy represents the number of true negative and true positive samples to the total number of samples.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)} \quad (8)$$

4.5. Results

Our CNN and LSTM models are combined in this study for which the accuracy gained is better than existing methods. For extraction of local features, CNN is used, while LSTM is used to learn long term dependencies and get sentence level representations. Our experimental results on Quora's dataset are summarized in Tables 5-6. Table 5 shows the accuracy obtained through the proposed model. Whereas, 83.15, 87.02 and 85.04 values for precision, recall and F1 score, respectively, are shown in Table 6. As we can see, the accuracy of our presented model is 87.50 which is better than that of existing approaches. In addition, the experimental results showed that proposed model obtained best precision, recall and F1 score, 83.15, 87.02 and 85.04 respectively. The results are better than that of previous textual similarity models.

We have applied different models while doing our research. Table 3 shows different methods applied during our research. We have used machine learning algorithms like Naïve Bayes and Random Forest. For

Table 3
Different models applied on dataset

Name	Method	Accuracy
Naïve Bayes	Features + Distance measures	**76.33 %**
Random Forest	Features + Distance measures	**77.14 %**
CNN 1D	Severyn, A. and A. Moschitti. [50] CNN only	**82.23 %**
LSTM	LSTM with Glove	**82.60 %**
LSTM	LSTM with Pre-trained Google's word to vector	**85.90 %**
LSTM + CNN	**LSTM and CNN with Pre-trained Google's word to vector**	**87.50 %**

Table 4
Hyperparameter values for our model

Hyperparameter	Value	Remarks
Max Sequence Length	30	-
Num of filters (CNN)	64	-
Filter length (CNN)	5	-
Batch size	2048	-
Max number of Words	200,000	-
Epochs	100	-
Loss	Binary cross entropy	-
Optimizer	nAdam	nAdam is a good optimizer to use
Activation unit	ReLU -Rectified linear units	Applied on almost all deep learning models.
Dropout	0.2	Dropped out visible or hidden units in Neural Networks to avoid overfitting
LR patience	5	Epochs to wait before reducing LR

using our data on machine learning algorithms, we need feature set. For this purpose, different features were collected like length of question, difference in length, common words etc. In addition to these features, distance measures, e.g. Cosine distance, Jaccard distance, etc. were used to measure distance between vectors of questions 1 and 2. After the collection of all features, Naïve Bayes and Random forest algorithms were applied to build a model and calculated results. Random forest performed better that Naïve Bayes and gave an accuracy of 77.14 %, greater than Naïve Bayes having 76.33 % accuracy with these features.

In addition to these models, we have also applied CNN and LSTM models individually and their results are shown in Table 3. Simple LSTM model using pre-trained Google's word2vec gave an accuracy of 85.90. We have seen that the performance of CNN + LSTM combined, are better than that of individuals.

Table 6 shows the results of precision, recall and F1 score. According to this table, we have seen that our method performed best (comparing with these techniques of sentence similarity) and achieved better values of precision, recall and F1 score.

The authors in [7] presented a CNN-based approach which generates vector representation of given question pairs and score them using similarity metric. Their proposed CCN-based method was tested on data from stack exchange forums. However, they did not mention F1 score, precision and recall. The authors of [57] applied a two-channel CNN to capture feature vectors from component matrix. They applied their model on answer selection problems. They calculated

Figure 2
Comparison of different optimizers applied on the dataset

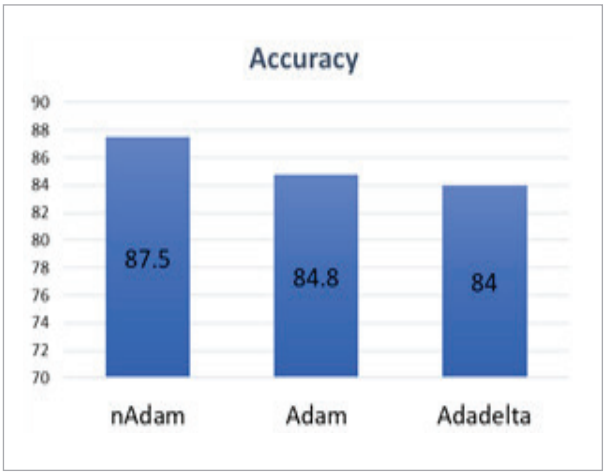


Table 5
Comparison of proposed model with existing approaches

Model	Accuracy
Agarwal, B., et al. [1]	77.70 %
Yin, W. and H. Schütze. [60]	78.10 %
Parikh, A.P. et al. [46]	86.00 %
Wang, Z. et al. [57]	77.14%
Jiang, J.-Y. et al. [23]	82.19 %
Homma, Y. et al. [18]	86.80 %
DeepSimilarity Model	87.50 %

Table 6
Comparison of precision, recall and F1 score

Approaches	Precision	Recall	F1 Score
Ferreira, R. et al. [13]	82.00	80.20	83.10
Jiang, J.-Y. et al. [23]	79.87	86.77	83.18
Yin, W. et al. [61]	-	-	84.80
Yin, W. and H. Schütze. [60]	-	-	84.40
Homma, Y. et al. [18]	-	-	81.73
Huang, J., et al. [20]	72.00	74.80	69.40
DeepSimilarity Model	83.15	87.02	85.04

mean average precision (MAP) and mean reciprocal rank (MRR) for QASent dataset and WikiQA dataset. In [15], the authors proposed Siamese Gated units in combination with machine learning algorithms like support vector machines, adaboost and random forest for predicting similar questions on Quora dataset, but they did not calculate accuracy rather they calculate cross entropy loss. Their measured cross entropy loss was 0.29568. Likewise, a deep learning model presented by [20] for detecting paraphrases and semantics similarity among short texts like tweets. For evaluating their model, they used a twitter dataset, and their results showed that character level features play an important role in finding similarity between tweets. Their max F1 score was 72.0%, whereas Pearson correlation was 62.6%. The authors of [18] tried to detect semantic similarity between Quora question pairs. They applied deep learning techniques and used the Siamese GRU network. They used data augmentation techniques to improve the performance of the model. Their model was trained on an augmented dataset using two layers similarity network and achieved an accuracy of 86.80% and 85.0% on validation and test set; however, the f1 score was 81.73% and 84.18% for validation and test set, respectively.

Compared to existing approaches and experimental results, we have seen that our model gave better accuracy. In addition, we explored that the joint model of LSTM and CNN gave better results than that of CNN and LSTM alone in similarity detection tasks. We take advantage of both these models and therefore achieve greater accuracy than existing approaches.

Figure 3
Model accuracy curve

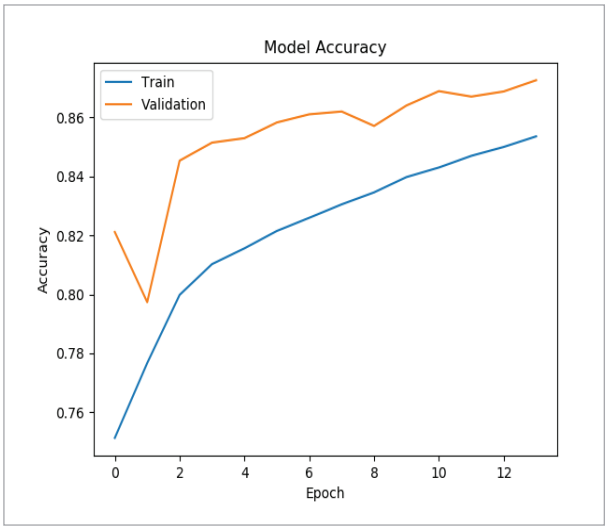
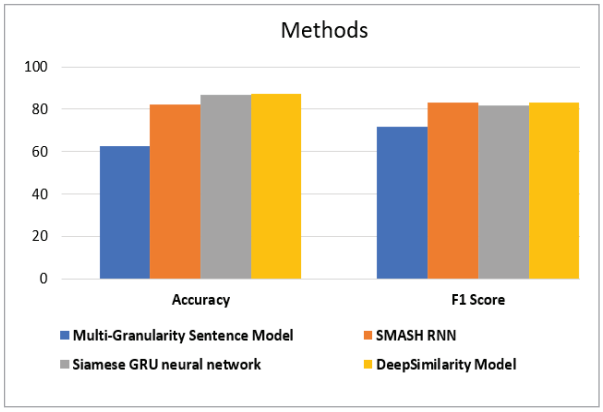


Figure 4
Performance of models



5. Conclusion

Measuring textual similarity between short sentences is a renowned problem in the area of NLP. In this paper, we have presented a novel deep learning approach to predict duplicate question pairs. We have used CNN and LSTM (Long Short-Term Memory) network which is good at storing long term dependencies. We take advantage of both CNN and LSTM models and thus gained accuracy better than existing methods. For our method, we have used Google's word2vec representation of text. Our proposed work on finding duplicate questions using deep learning models performed very well and gave better results than existing approaches. The proposed method was evaluated on Quora's dataset and gained an accuracy of 87.50%. The main advantage of using deep learning is that it only took sentences as inputs, thus required minimal preprocessing and saved time by avoiding complex feature engineering. For our future research, we will investigate how to apply our proposed model on similar tasks like information retrieval, document matching, etc. Moreover, we will apply word embedding other than Google's Word to vectors and investigate their impact.

References

1. Agarwal, B., Ramampiaro, H., Langseth, H., Ruocco, M. A Deep Network Model for Paraphrase Detection in Short Text Messages. Information Processing & Management, 2018, 54(6), 922-937. https://doi.org/10.1016/j.ipm.2018.06.005
2. Arora, S., Liang, Y., Ma, T. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. ICLR, 2017.
3. Arshad, H., Khan, M. A., Sharif, M. I., Yasmin, M., Tavares, J. M. R., Zhang, Y. D., Satapathy, S. C. A Multilevel Paradigm for Deep Convolutional Neural Network Features Selection with an Application to Human Gait Recognition. Expert Systems, 2020, e12541. https://doi.org/10.1111/exsy.12541
4. Aurangzeb, K., F. Akmal, M. A. Khan, M. Sharif, Javed, M. Y. Advanced Machine Learning Algorithm Based System for Crops Leaf Diseases Recognition. In 2020 6th IEEE Conference on Data Science and Machine Learning Applications (CDMA), 2020. https://doi.org/10.1109/CDMA47397.2020.00031
5. Bao, W., Bao, W., Du, J., Yang, Y., Zhao, X. Attentive Siamese LSTM Network for Semantic Textual Similarity Measure. In 2018 IEEE International Conference on Asian Language Processing (IALP). 2018. https://doi.org/10.1109/IALP.2018.8629212
6. Batool, F. E., Attique, M., Sharif, M., Javed, K., Nazir, M., Abbasi, A. A., Iqbal, Z., Riaz, N. Offline Signature Verification System: A Novel Technique of Fusion of GLCM and Geometric Features Using SVM. Multimedia Tools and Applications, 2020, 1-20. https://doi.org/10.1007/s11042-020-08851-4
7. Bogdanova, D., dos Santos, C., Barbosa, L., Zadrozny, B. Detecting Semantically Equivalent Questions in Online User Forums. In Proceedings of the Nineteenth Conference on Computational Natural Language Learning, 2015. https://doi.org/10.18653/v1/K15-1013
8. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics, 2017, 5, 135-146. https://doi.org/10.1162/tacl_a_00051
9. Chen, G., Ye, D., Xing, Z., Chen, J., Cambria, E. Ensemble Application of Convolutional and Recurrent Neural Networks for Multi-Label Text Categorization. In IEEE 2017 International Joint Conference on Neural Networks (IJCNN), 2017. https://doi.org/10.1109/IJCNN.2017.7966144
10. Das, D., Smith, N. A. Paraphrase Identification as Probabilistic Quasi-Synchronous Recognition. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1, 2009. Association for Computational Linguistics. https://doi.org/10.3115/1687878.1687944
11. Dos Santos, C., Gatti, M. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, 2014.
12. Eyecioglu, A., Keller, B. Twitter Paraphrase Identification with Simple Overlap Features and SVMs. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), 2015. https://doi.org/10.18653/v1/S15-2011
13. Ferreira, R., Cavalcanti, G. D., Freitas, F., Lins, R. D., Simske, S. J., Riss, M. Combining Sentence Similarities Measures to Identify Paraphrases. Computer Speech & Language, 2018, 47, 59-73. https://doi.org/10.1016/j.csl.2017.07.002
14. Gardenfors, P., Conceptual Spaces-the Geometry of Thought. Cambridge, 2000, MA. https://doi.org/10.7551/mitpress/2076.001.0001
15. Godbole, A., Dalmia, A., Sahu, S. K. Siamese Neural Networks with Random Forest for Detecting Duplicate Question Pairs. arXiv preprint arXiv:1801.07288, 2018.
16. Goldberg, Y., Levy, O. word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method. arXiv preprint arXiv:1402.3722, 2014.
17. Hasan, S. A., Farri, O. Clinical Natural Language Processing with Deep Learning. In Data Science for Healthcare, 2019, 147-171. https://doi.org/10.1007/978-3-030-05249-2_5
18. Homma, Y., Sy, S., Yeh, C. Detecting duplicate Questions with Deep Learning. In Proceedings of the International Conference on Neural Information Processing Systems, NIPS, 2016.
19. Hu, B., Lu, Z., Li, H., Chen, Q. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In Advances in Neural Information Processing Systems, 2014.
20. Huang, J., Yao, S., Lyu, C., Ji, D. Multi-Granularity Neural Sentence Model for Measuring Short Text

Similarity. In International Conference on Database Systems for Advanced Applications, 2017. https://doi.org/10.1007/978-3-319-55753-3_28

21. Hussain, N., Khan, M. A., Sharif, M., Khan, S. A., Albesher, A. A., Saba, T., Armaghan, A. A Deep Neural Network and Classical Features Based Scheme for Objects Recognition: An Application for Machine Inspection. Multimedia Tools and Applications, 2020. https://doi.org/10.1007/s11042-020-08852-3
22. Janani, R., Vijayarani, S. Text Document Clustering Using Spectral Clustering Algorithm with Particle Swarm Optimization. Expert Systems with Applications, 2019, 134, 192-200. https://doi.org/10.1016/j.eswa.2019.05.030
23. Jiang, J.-Y., Zhang, M., Li, C., Bendersky, M., Golbandi, N., Najork, M. Semantic Text Matching for Long-Form Documents. In The World Wide Web Conference, 2019. https://doi.org/10.1145/3308558.3313707
24. Joshi, A., Fidalgo, E., Alegre, E., Fernández-Robles, L. SummCoder: An Unsupervised Framework for Extractive Text Summarization Based on Deep Auto-Encoders. Expert Systems with Applications, 2019, 129, 200-215. https://doi.org/10.1016/j.eswa.2019.03.045
25. Khan, M. A., Akram, T., Sharif, M., Javed, K., Raza, M., Saba, T. An Automated System for Cucumber Leaf Diseased Spot Detection and Classification Using Improved Saliency Method and Deep Features Selection. Multimedia Tools and Applications, 2020, 1-30.
26. Khan, M. A., Javed, K., Khan, S. A., Saba, T., Habib, U., Khan, J. A., Abbasi, A. A. Human Action Recognition Using Fusion of Multiview and Deep Features: An Application to Video Surveillance. Multimedia Tools and Applications, 2020, 1-27. https://doi.org/10.1007/s11042-020-08806-9
27. Khan, M. A., Khan, M. A., Ahmed, F., Mittal, M., Goyal, L. M., Hemanth, D. J., Satapathy, S. C. Gastrointestinal Diseases Segmentation and Classification Based on Duo-Deep Architectures. Pattern Recognition Letters, 2020, 131, 193-204. https://doi.org/10.1016/j.patrec.2019.12.024
28. Khan, M. A., Lali, M. I. U., Sharif, M., Javed, K., Aurangzeb, K., Haider, S. I., Altamrah, A. S., Akram, T. An Optimized Method for Segmentation and Classification of Apple Diseases Based on Strong Correlation and Genetic Algorithm Based Feature Selection. IEEE Access, 2019, 7, 46261-46277. https://doi.org/10.1109/ACCESS.2019.2908040
29. Khan, M. A., Sharif, M. I., Raza, M., Anjum, A., Saba, T., Shad, S.A. Skin Lesion Segmentation and Classification: A Unified Framework of Deep Neural Network Features Fusion and Selection. Expert Systems, 2019, e12497. https://doi.org/10.1111/exsy.12497
30. Khan, S. A., Hussain, A., Usman, M. Reliable Facial Expression Recognition for Multi-Scale Images Using Weber Local Binary Image Based Cosine Transform Features. Multimedia Tools and Applications, 2018, 77(1), 1133-1165. https://doi.org/10.1007/s11042-016-4324-z
31. Khan, S. A., Hussain, S., Yang, S. Contrast Enhancement of Low-Contrast Medical Images Using Modified Contrast Limited Adaptive Histogram Equalization. Journal of Medical Imaging and Health Informatics, 2020, 10(8), 1795-1803. https://doi.org/10.1166/jmihi.2020.3196
32. Khan, S. A., Ishtiaq, M., Nazir, M., Shaheen, M. Face Recognition Under Varying Expressions and Illumination Using Particle Swarm Optimization. Journal of Computational Science, 2018, 28, 94-100. https://doi.org/10.1016/j.jocs.2018.08.005
33. Kim, Y., Jernite, Y., Sontag, D., Rush, A. M. Character-Aware Neural Language Models. In Thirtieth AAAI Conference on Artificial Intelligence, 2016.
34. Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R. Urtasun, R., Torralba, A., Fidler, S. Skip-Thought Vectors. In Advances in Neural Information Processing Systems, 2015.
35. Li, Y., McLean, D., Bandar, Z. A., O'shea, J. D., Crockett, K. Sentence Similarity Based on Semantic Nets and Corpus Statistics. IEEE Transactions on Knowledge and Data Engineering, 2006, 18(8), 1138-1150. https://doi.org/10.1109/TKDE.2006.130
36. Liang, D., Zhang, F., Zhang, W., Zhang, Q., Fu, J., Peng, M., Gui, T., Huang, X. Adaptive Multi-Attention Network Incorporating Answer Information for Duplicate Question Detection. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019. https://doi.org/10.1145/3331184.3331228
37. Liu, G., Guo, J. Bidirectional LSTM with Attention Mechanism and Convolutional Layer for Text Classification. Neurocomputing, 2019, 337, 325-338. https://doi.org/10.1016/j.neucom.2019.01.078
38. Lopez-Gazpio, I., Maritxalar, M., Gonzalez-Agirre, A., Rigau, G., Uria, L., Agirre, E. Interpretable Semantic Textual Similarity: Finding and Explaining Differences Be-

tween Sentences. Knowledge-Based Systems, 2017, 119, 186-199. https://doi.org/10.1016/j.knosys.2016.12.013

39. Ltaifa, I. B., Hlaoua, L., Romdhane, L. B. Hybrid Deep Neural Network-Based Text Representation Model to Improve Microblog Retrieval. Cybernetics and Systems, 2019, 1-25.
40. Madnani, N., Tetreault, J., Chodorow, M. 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, 2012. Association for Computational Linguistics.
41. Mahmood, A., Khan, S. A., Hussain, S., Almaghayreh, E. M. An Adaptive Image Contrast Enhancement Technique for Low-Contrast Images. IEEE Access, 2019, 7, 161584-161593. https://doi.org/10.1109/ACCESS.2019.2951468
42. Majid, A., Khan, M. A., Yasmin, M., Rehman, A. Yousafzai, A., Tariq, U. Classification of Stomach Infections: A Paradigm of Convolutional Neural Network Along with Classical Features Fusion and Selection. Microscopy Research and Technique, 2020. https://doi.org/10.1002/jemt.23447
43. Mohammad, A.-S., Jaradat, Z., Mahmoud, A.-A., Jararweh, Y. Paraphrase Identification and Semantic Text Similarity Analysis in Arabic News Tweets Using Lexical, Syntactic, and Semantic Features. Information Processing & Management, 2017, 53(3), 640-652. https://doi.org/10.1016/j.ipm.2017.01.002
44. Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., Muharemagic, E. Deep Learning Applications and Challenges in Big Data Analytics. Journal of Big Data, 2015, 2(1), 1. https://doi.org/10.1186/s40537-014-0007-7
45. Onan, A. Two-Stage Topic Extraction Model for Bibliometric Data Analysis Based on Word Embeddings and Clustering. IEEE Access, 2019, 7, 145614-145633. https://doi.org/10.1109/ACCESS.2019.2945911
46. Parikh, A. P., Täckström, O., Das, D., Uszkoreit, J. A Decomposable Attention Model for Natural Language Inference. arXiv preprint arXiv:1606.01933, 2016. https://doi.org/10.18653/v1/D16-1244
47. Rashid, M., Khan, M. A., Sharif, M., Raza, M., Sarfraz, M., Afza, F. Object Detection and Classification: A Joint Selection and Fusion Strategy of Deep Convolutional Neural Network and SIFT Point Features. Multimedia Tools and Applications, 2019, 78(12), 15751-15777. https://doi.org/10.1007/s11042-018-7031-0
48. Rehman, A., Khan, M. A., Mehmood, Z., Saba, T., Sardaraz, M., Rashid, M. Microscopic Melanoma Detection and Classification: A Framework of Pixel-Based Fusion and Multilevel Features Reduction. Microscopy Research and Technique, 2020. https://doi.org/10.1002/jemt.23429
49. Sahi, M., Gupta, V. A Novel Technique for Detecting Plagiarism in Documents Exploiting Information Sources. Cognitive Computation, 2017, 9(6), 852-867. https://doi.org/10.1007/s12559-017-9502-4
50. Severyn, A., Moschitti, A. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015. https://doi.org/10.1145/2766462.2767738
51. Sharif, M., Akram, T., Raza, M., Saba, T., Rehman, A. Hand-crafted and Deep Convolutional Neural Network Features Fusion and Selection Strategy: An Application to Intelligent Human Action Recognition. Applied Soft Computing, 2020, 87, 105986. https://doi.org/10.1016/j.asoc.2019.105986
52. Sharif, M. I., Li, J. P., Khan, M. A., Saleem, M. A. Active Deep Neural Network Features Selection for Segmentation and Recognition of Brain Tumors Using MRI Images. Pattern Recognition Letters, 2020, 129, 181-189. https://doi.org/10.1016/j.patrec.2019.11.019
53. Smith, L. B., Heise, D. Perceptual Similarity and Conceptual Structure. In Advances in Psychology. 1992, 233-272. https://doi.org/10.1016/S0166-4115(08)61009-2
54. Vo, N. P. A., Magnolini, S., Popescu, O. Paraphrase Identification and Semantic Similarity in Twitter with Simple Features. In Proceedings of the Third International Workshop on Natural Language Processing for Social Media, 2015. https://doi.org/10.3115/v1/W15-1702
55. Wang, X., Jiang, W., Luo, Z. Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical papers, 2016.
56. Wang, Z., Hamza, W., Florian, R. Bilateral Multi-Perspective Matching for Natural Language Sentences. arXiv preprint arXiv:1702.03814, 2017. https://doi.org/10.24963/ijcai.2017/579
57. Wang, Z., Mi, H., Ittycheriah, A. Sentence Similarity Learning by Lexical Decomposition and Composition. arXiv preprint arXiv:1602.07019, 2016.

58. Wieting, J., Gimpel, K. Revisiting Recurrent Networks for Paraphrastic Sentence Embeddings. arXiv preprint arXiv:1705.00364, 2017. ttps://doi.org/10.18653/v1/P17-1190

59. Xu, W., Ritter, A., Callison-Burch, C., Dolan, W. B., Ji, Y. Extracting Lexically Divergent Paraphrases from Twitter. Transactions of the Association for Computational Linguistics, 2014, 2, 435-448. https://doi.org/10.1162/tacl_a_00194

60. Yin, W., Schütze, H. Convolutional Neural Network for Paraphrase Identification. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015. https://doi.org/10.3115/v1/N15-1091

61. Yin, W., Schütze, H., Xiang, B., Zhou, B. Abcnn: Attention-based Convolutional Neural Network for Modeling Sentence Pairs. Transactions of the Association for Computational Linguistics, 2016, 4, 259-272. https://doi.org/10.1162/tacl_a_00097