# Quora Question Pairs

**Zihan Chen, Hongbo Zhang, Xiaoji Zhang, Leqi Zhao**
University of Waterloo
{zihan.chen, hongbo.zhang, joy.zhang.1, l67zhao}@uwaterloo.ca

## 1 Introduction

Quora Question Pairs is an active Kaggle Competition, which challenges participants to tackle the natural language processing (NLP) problem of identifying duplicate questions [1]. The issue of duplicate questions stems from the enormous number of visitors on the Quora website (a platform for asking questions and connecting with people that contribute answers), making it hard to avoid having similar worded questions from different users. Effectively detecting duplicate questions not only saves time for seekers to find the best answer to their questions, but also reduces the effort of writers in terms of answering multiple versions of the same question[1].

To tackle the duplicate detection problem, we apply simple feature engineering as well as more complex neural-network based models. We evaluate the performance by computing the log loss between the predicted values and the ground truth [1].

We start by introducing the previous work in paraphrase identification, which includes the proposal of three classes of language models ("siamese", "compare-aggregate" and "attention-based"). Subsequently, we analyze the data provided and observe three properties of the data set that are important for feature extraction and model selection. We then elaborate on our feature engineering process, discuss the choice of neural networks (RNNs and LSTMs) in modeling the question pairs, and propose a new siamese LSTM structure based on the existing Manhatten LSTM model [7]. Finally, we run experiments regarding the proposed features as well as our new model, and achieve a log loss score of $0.28446$, which ranks the top $8\%$ among all participants of the competition.

## 2 Related Work

The task of identifying duplicated questions can be viewed as an instance of the paraphrase identification problem, which is a well-studied NLP task that uses natural language sentence matching (NLSM) to determine whether two sentences are paraphrase or not [3].

With the renaissance of neural networks [3], two types of neural-based frameworks have been proposed for the task of paraphrase identification. The first framework is based on a "siamese" neural network consisting of two sub-networks joined at their outputs, where the sub-networks share the same weights at all levels and are responsible for extracting features from the input, and the output level computes the distance (cosine of the angle) between the two feature vectors generated by the sub-networks [6]. Although the siamese structure is lightweight and easy to train given that the sub-networks share parameters, there is no interaction between the two sentences during the training process, which might cause information loss [3]. To cope with the limitations of the siamese framework, a second framework named "compare-aggregate" is proposed [4], which captures the interaction between two sentences by performing a word-level matching and aggregating the matching results into a vector for the final classification. However, the "compare-aggregate" model fails to account for other types of granular matchings (e.g. phrase-by-sentence) and only performs matching in a single direction, which also neglects information in the sentence pairs [3].

To tackle the limitations of the neural-based frameworks, Wang et al. proposed a bilateral multi-perspective matching model (BiMPM) which also applies the "compare-aggregate" framework, but instead matches the two sentences bidirectionally. Specifically, the model encodes the input sentences

using a Bidirectional Long-Short Term Memory Network (BiLSTM), matches the encodings in two directions as well as multiple perspectives in each direction, and aggregates the matching results into a fixed-length vector using another BiLSTM layer [3]. Tomar et al. later used character-based $n$-gram embeddings instead of word embeddings to improve the performance of BiMPM [2]. By matching the sentences from multiple perspectives, the model captures more interactive features between the sentence pairs and achieves state-of-the-art performance in paraphrase identification [3].

Aside from the "siamese" and "compare-aggregate" frameworks, another effective approach to the paraphrase identification problem is by using the "attention-based" framework, which models the interdependence between the two sentences in a sentence pair [8]. One of the best-performing attention-based models is the attention-based convolutional neural network (ABCNN), which achieves attention at multiple levels of granularity (word-level, phrase-level and sentence-level) by stacking multiple convolutional layers, thus capturing the information in the sentences at different levels of abstraction [8].

## 3 Data Analysis

In this section, we introduce the general features as well as three specific properties that we observe with respect to the data provided. The observed features not only provide us with a more profound understanding of the data, but are also important factors to consider in the process of feature extraction and model selection.

### 3.1 Overview

The training data set for the competition contains 404290 question pairs, whereas the test data set is almost 6 times larger than the training set, with 2345796 question pairs. Each data entry consists of the ID of the question pair, the unique IDs and full text of each question, as well as the target variable indicating whether the two questions are duplicates (i.e. have the same meaning) or not.

The question pairs cover a great range of topics includes technology, entertainment, politics, culture and philosophy. Some questions also include special characters such as mathematics symbols and foreign language characters.

It is worth mentioning that the duplicate labels in the training set are provided by human experts and are inherently subjective [1], since the true meaning of sentences can never be known with certainty. As a result, the labels on the training data set represent a reasonable consensus and are considered ground truth, but are not 100% accurate [1], i.e. the data set might be noisy, as we see in later analysis.

### 3.2 Analysis

Using simple analysis, we find out that the duplicated question pairs only make up 37% of the training data, indicating that the data set is heavily imbalanced with many more non-duplicated question pairs than duplicated ones. In addition, 20% of questions appear multiple times in different question pairs. Furthermore, we make three observations with respect to the data sets, illustrated as follows.

The first observation is that the training data set is noisy, i.e. there are question pairs labelled as non-duplicated where the two questions share almost all the words and the exact same meaning. For instance, the question pair consisting of sentences "What is the solution for this question?" and "What is the solution to this question?" is labelled as non-duplicated, despite the fact that the two questions only differ in an insignificant stop word.

The second observation is that the order of words have a significant influence on the identification of duplicated questions, meaning that simple methods involving only the word-to-word matching between sentences and neglecting the higher level abstractions will fail to capture the question semantics and falsely label some question pairs. A good example is the question pair ["Why is Google Chrome not working but Internet Explorer is?", "Why is Internet Explorer not working but Google Chrome is?"], where the two questions consist of the exact same words but differ in meanings because of the different word orderings.

The third observation concerns the first words of the questions, in that the most common first words in the data set (What, How, Why, Is, Which, Can, I and Who) are mostly interrogative, as expected given

the typical semantic structure of questions. However, the first words of questions are not significant in the detection of duplicates, since the proportion of question pairs where the two questions share the same first word for duplicated and non-duplicated classes are 62.3% and 41.9% respectively, which do not have a huge difference.

# 4 Feature Engineering

In this section, we elaborate on our feature engineering process, which involves the extraction of several word-level and sentence-level features, as well as the comparison of features in terms of the predictive power in identifying the duplicated questions.

## 4.1 Simple Features

To evaluate the similarity between two sentences, we use a set of five simple word-level and sentence-level metrics that can be computed in a straight-forward fashion, i.e. do not involve the abstraction process of generating word or sentence embeddings.

The first two features focus on the direct similarity of the two questions, i.e. the difference in the sentence length and the proportion of words in common. Additionally, a "match-ratio" feature is introduced, which utilizes the Python *SequenceMatcher* class to find the longest contiguous matching subsequence with no "junk" element [9] between two questions. The other two features are based on the Term Frequency-Inverse Document Frequency metric (TF-IDF), which reflects the importance of a word to the document [10] and assigns a corresponding score to each word in the data set. The first TF-IDF based feature focuses on the normalized score of the words shared by the two questions, whereas the second feature emphasizes the difference in the total score of the two questions (calculated by adding up the scores of individual words in each question). In particular, the TF-IDF features remove the influence of "stop words", i.e. the most common words in the language.

## 4.2 Word-to-Mover Distance

In addition to the simple features, we apply a Word-to-Mover Distance (WMD) metric, which is useful in cases where two questions have different wordings but convey the same information. Specifically, WMD measures the difference between two questions by calculating the minimum distance that the Word2Vec embeddings of words in one question need to "travel" to reach the embedded words in the other question [11]. However, the WMD metric performs poorly in separating the duplicated question pairs from non-duplicated ones, mainly because it fails to account for the fact that even the questions in a non-duplicated pair share words with similar meanings.

# 5 RNN-LSTM

Aside from the simple word-level and sentence-level metrics for evaluating the dissimilarity between two questions, we want to utilize the power of neural networks in modeling the data, given the large number of data entries we are presented with. Recurrent neural networks (RNNs) are rather effective at modeling sequential data, in that they have a memory storage that captures the previous inputs as well as the corresponding computations, and allows information to cycle inside the networks for arbitrarily long time (hence the name recurrent) [12]. However, RNN has its restrictions in learning long-term dependencies, and decreases in performance with the increasing length of the input sequence.

Given the limitations of RNN, a Long-Short Term Memory (LSTM) structure is widely used in sequence modeling, which is capable of learning long-term dependencies using four interacting layers, as opposed to the one information processing layer in the RNN structure [13]. In addition to the better performance in modeling long-term dependencies, LSTM provides more flexibility in terms of controlling the amount of stored information as needed, which makes it an ideal choice for our purpose of sequence modeling.

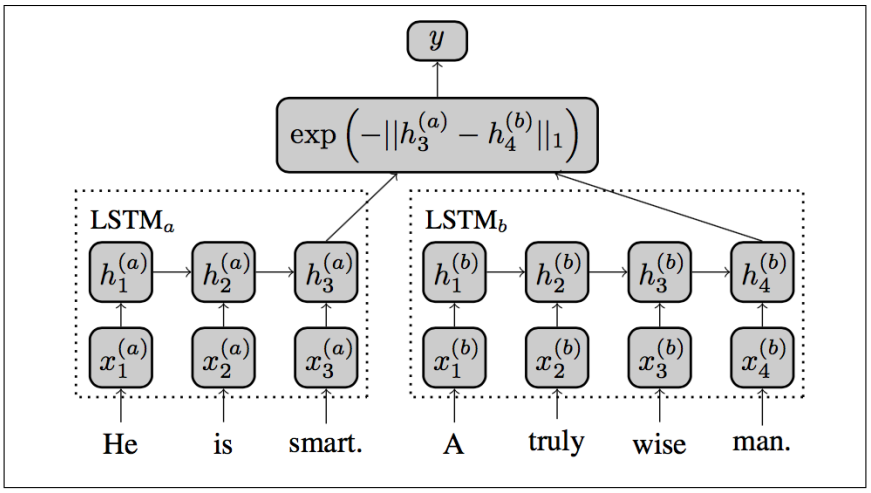


Figure 1: An illustration of the Manhattan LSTM structure. *Adapted from Mueller, Jonas, and Aditya Thyagarajan. 'Siamese Recurrent Architectures for Learning Sentence Similarity." AAAI. 2016. Adapted with permission.*

### 5.1 Siamese Neural Network

Siamese neural network is an architecture that contains two identical sub-networks joined at the outputs [6], and is widely used in tasks that involve finding the similarity between two comparable patterns, e.g. paraphrase identification [14]. A key feature of the siamese structure is the shared weights across the sub-networks, which reduce the number of parameters for training as well as the tendency of overfitting [14]. In addition, by processing similar inputs with similar models, the sub-networks generate input representations that share the same semantics and are easy to compare [14].

### 5.2 Manhattan LSTM

The Manhattan LSTM model (MaLSTM) utilizes the siamese architecture, where there are two identical LSTM sub-networks with tied weights (each processing one of the questions) that learn the mappings from the variable-length input sequences to the fixed-length vector representations. Subsequently, a pre-defined similarity function (as shown in Figure 1) is applied to the outputs of the sub-networks, which then generates a value that is used for inferring the semantic similarity of the two questions [7]. The name "Manhatten" stems from the similarity function, which applies the Manhatten distance metric (or L1 norm) and forces the model to entirely capture the semantic differences between the two questions in the training process [7].

### 5.3 Our Structure

Inspired by the Manhattan LSTM model, we design a siamese structure that consists of two LSTM sub-networks (one for each question in the question pair) and a subsequent concatenation layer that merges the outputs of the two sub-networks into a single representation for the final classification, which employs a 3-layer neural network as opposed to directly measuring the similarity of the outputs in the Manhattan LSTM model. Our structure is elaborated as follows (also shown in Figure 2).

The structure begins with two input layers, each passing the vector representation of a question in the input question pair to the sequence embedding layer, which then generates a sentence embedding for the question and feeds the embedding into the LSTM network. Subsequently, the LSTM layers generate the vector representations of the two questions in the input question pair. A concatenation layer is then applied for concatenating the two input representations into a single vector representation, which is then used for the final classification.

The classification is conducted using a simple 3-layer neural network consisting of an input layer, a hidden layer, and an output layer. Given the concatenated vector representation of the input question pair, the neural network outputs a single value indicating the probability that the two question are

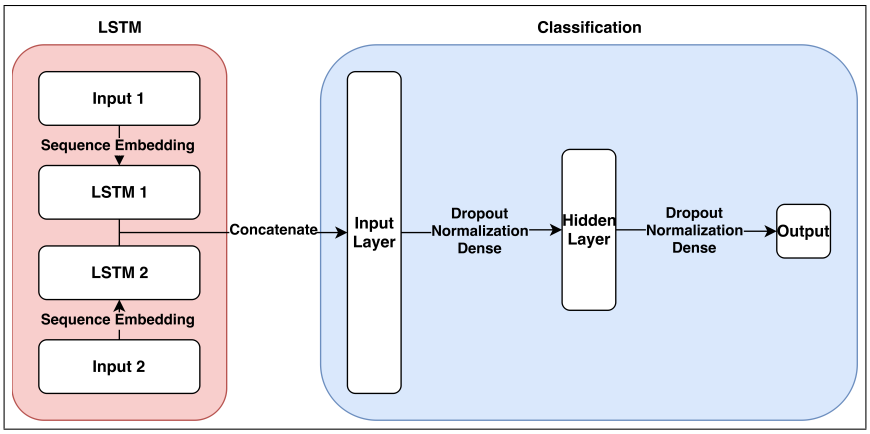


Figure 2: An illustration of our siamese LSTM structure.

duplicates. In addition, the normalization and dropout layers are inserted between every two layers for normalizing data and preventing overfitting.

### 5.4 Implementation Details

Upon loading the question pairs in the training and test data sets, we construct a tokenizer that maps each word to a globally unique token, i.e. among all the text components in the training and test data sets. We then obtain a fixed-length vector representation for each question via tokenizing and padding. Subsequently, we use the pre-trained Word2Vec data to generate a sequence embedding for each question.

To void the influence of the order of the two questions, we duplicate every question pair in the training data but reverse the order of the questions. Furthermore, to determine whether the questions $q_1$ and $q_2$ are duplicates, we compute the probabilities of both ($q_1$, $q_2$) and ($q_2$, $q_1$) and average the two outputs to get the final result.

## 6 Experiments and Results

We start by performing data cleansing, which involves removing the stop words, replacing the abbreviations with the corresponding full words, and converting all the text into lower-case. Subsequently, we conduct feature engineering with respect to the simple word-level and sentence-level features illustrated above. The results are shown in Figure 3, where it is evident that the two features with respect to the difference in question lengths as well as the difference in the total TF-IDF scores of questions hardly separate the duplicated question pairs from the non-duplicated ones. Additionally, we evaluate the level of importance for each feature (Figure 4) and determine that the match_ratio (longest contiguous matching subsequence with no "junk" element) and tfidf_sum (normalized TF-IDF score of the shared words) have the strongest predictive power in the problem of identifying duplicated questions. Our feature engineering achieves a log loss score of $0.36376$.

By applying our siamese LSTM structure, we obtain a log loss score of $0.30266$ from a single classifier (trained for one hour). For improvement, we train ten classifiers and average the predictions to reduces the log loss to $0.28446$, which is our best result so far, with a rank of 136 out of 1766 participants (top 8%).

## 7 Conclusion

We have introduced the Quora Question Pairs Kaggle competition, the paraphrase identification NLP problem as well as the properties of the given data set. We have elaborated on our feature engineering process of extracting word-level and sentence-level features that have strong predictive power for identifying duplicated questions. Additionally, we have discussed the use of neural networks (RNNs

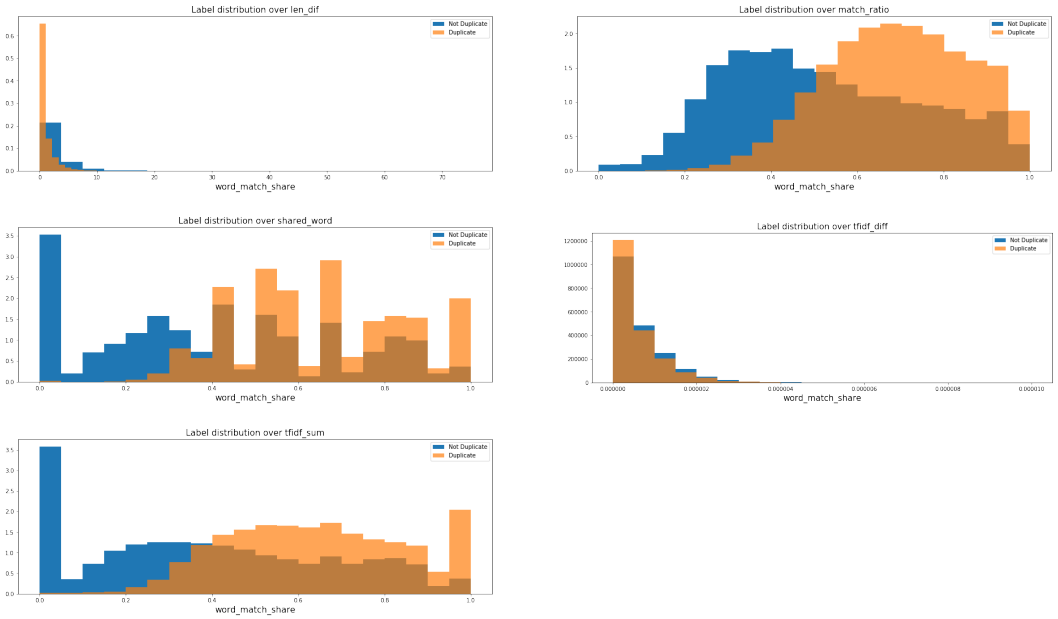


Figure 3: Label distribution of the duplicated/non-duplicated question pairs over different features. First column (top to bottom): len_diff (difference in question lengths), shared_words (proportion of shared words in the two questions), tfidf_sum (normalized TF-IDF score of the shared words). Second column (top to bottom): match_ratio (longest contiguous matching subsequence with no "junk" element), tfidf_diff (difference in the total TF-IDF score of the two questions).

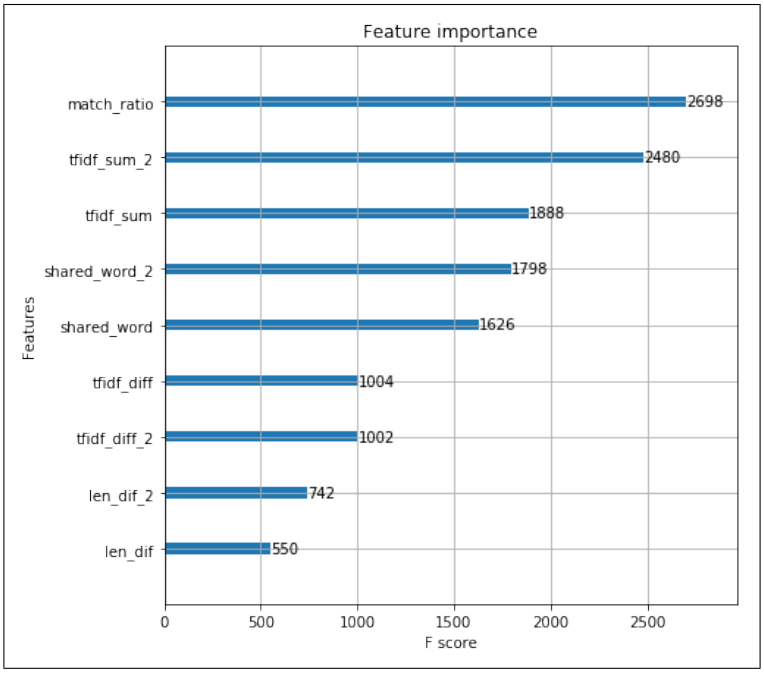


Figure 4: The importance level of all features in the identification of duplicate questions.

and LSTMs) as well as the siamese structure for modeling the question pairs. Furthermore, inspired by the Manhatten LSTM model, we have proposed a new siamese LSTM structure which makes use of two identical LSTM sub-networks and employs a 3-layer neural network for the final classification. We have achieved a log loss score of $0.28446$ by applying our siamese LSTM structure, which is currently at the top 8% among all participants of the competition.

Future effort is required in terms of exploring different approaches to improve our siamese LSTM model, as well as experimenting with the state-of-the-art models in paraphrase identification regarding the "compare-aggregate" and "attention-based" framework, which include the BiMPM model and the ABCNN model.

## References


[1] "Quora Question Pairs | Kaggle". *Kaggle.com*. N.p., 2017. Web. 23 Apr. 2017.

[2] Tomar, Gaurav Singh, et al. "Neural Paraphrase Identification of Questions with Noisy Pretraining." *arXiv preprint arXiv:1704.04565* (2017).

[3] Wang, Zhiguo, Wael Hamza, and Radu Florian. "Bilateral Multi-Perspective Matching for Natural Language Sentences." *arXiv preprint arXiv:1702.03814* (2017).

[4] Wang, Shuohang, and Jing Jiang. "A Compare-Aggregate Model for Matching Text Sequences." *arXiv preprint arXiv:1611.01747* (2016).

[5] Palangi, Hamid, et al. "Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval." *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 24.4* (2016): 694-707.

[6] Bromley, Jane, et al. "Signature Verification Using A "Siamese" Time Delay Neural Network." IJPRAI 7.4 (1993): 669-688.

[7] Mueller, Jonas, and Aditya Thyagarajan. "Siamese Recurrent Architectures for Learning Sentence Similarity." *AAAI*. 2016.

[8] Yin, Wenpeng, et al. "Abcnn: Attention-based convolutional neural network for modeling sentence pairs." *arXiv preprint arXiv:1512.05193* (2015).

[9] "7.4. Difflib — Helpers For Computing Deltas — Python 2.7.13 Documentation". *Docs.python.org*. N.p., 2017. Web. 24 Apr. 2017.

[10] "Tf–Idf". *En.wikipedia.org*. N.p., 2017. Web. 24 Apr. 2017.

[11] Kusner, Matt, et al. "From word embeddings to document distances." *International Conference on Machine Learning*. 2015.

[12] Mikolov, Tomas, et al. "Recurrent neural network based language model." *Interspeech.* Vol.2. 2010.

[13] "Understanding LSTM Networks" *Colah's blog*. Aug 27, 2015. Web. 24 Apr. 2017.

[14] Pond Premtoon, anwers to Quora question "What are Siamese neural networks, what applications are they good for, and why?" *quora.com*. Jun 1, 2016. Web. 24 Apr. 2017.