Duplicate Quora Questions Detection

Lei Guo
Department of Computer Science
New York University
lg2681@nyu.edu

Chong Li
Department of Computer Science
New York University
cl4056@nyu.edu

Haiming Tian
Department of Computer Science
New York University
ht971@nyu.edu

***Abstract*—Quora is a platform to ask questions and connect with people who contribute unique insights and quality answers. In this paper, we are mainly focusing on the duplicate questions detection. The main idea is to first vectorize questions and extract features, train and predict using machine learning techniques based on question vectors and features previously built. We implement two approaches to detect if two questions are duplicate. Different vectorization and feature extracting methods are used in the two approaches, one is based on the Word2Vec model and TF-IDF score, the other one is a Neural Network method based on term frequency. Different classification methods are also used, for example, KNN, SVM and Random Forest. We reach accuracies of nearly 80% in both two approaches. Finally, we build a search engine with duplication detection integrated.**

***Keywords*—Word2Vec, TF-IDF, LSTM, Classification.**

I. INTRODUCTION

Over 100 million people visit Quora every month, so it's no surprise that many people ask similarly worded questions. Multiple questions with the same intent can cause seekers to spend more time finding the best answer to their question, and make writers feel they need to answer multiple versions of the same question. Thus, detecting duplicate questions could greatly benefit the community. Currently, Quora uses a Random Forest model to identify duplicate questions[1].

To detect whether two questions are duplicate or not, the biggest challenge is to extract the semantic meaning of a question. There are quite a few research papers on measuring semantic similarities between sentences. For example, in Thanh Ngoc Dao's paper[2], he proposed several methods of measuring similarity between sentences based on WordNet. Inspired by his work, in this project, we implement two new approaches to tackle this challenge. The main idea of both two approaches is to first perform vectorization and feature extractions of all questions, then train and predict based on question vectors and features previously built. Our data source is the open Quora question pairs provided by Quora.

In our first approach, we first vectorize all words in our question set based on Google's Word2Vec model[3]. To build a vector for the question from its words' vectors, we simply take the mean of all words vectors. Note that in this method, we did not consider the effect of word sequences in the questions. Also, it is very likely that two questions with different words appearances could happen to have very similar vector representations. Further, to make up this loss of information, we first multiply each word vector by its TF-IDF weight, then we still take the mean of each word vectors to construct the vector representations for the question. We keep the question vectors in both methods to achieve a better result. Based on these two kinds of question vectors, after normalization, we can compute the similarity of two questions by performing an inner product on two vectors. The two similarity scores along with the common words percentage (number of common words in two questions divided by total length of questions), length difference percentage (difference in length of two questions divided by total length of questions), forms our 4 features for training the classifier.

Our second approach is an experimental one based on LSTM (long short term memory) recurrent neural network. In this approach, due to the input requirement of LSTM, we followed the classic way of assigning each word a unique id based on its frequency of occurrence, words with same frequency will be randomly ordered. Then a question can be represented by a sequence of numbers. The embedding weight is again based on word2vec model. Then we build, compile, fit and evaluate the LSTM model. This approach is based on the keras deep learning library.

Further, we tried 5 different classification methods, which are Decision Tree, Random Forest, K Nearest Neighbors, SVM and Adaboost, to train the dataset we acquired from the first approach. Since the KNN method gives us the least Mean Square Errors and Training Error, we finally choose to use the KNN model for prediction on the test set. As for the second approach, we simply ran the built-in evaluation function to see the accuracy, no classification method performed since we regarded this as just an experiment.

We finally build a search engine based on our labeled test dataset, thus, it can show the results of not only the related question pairs, but also whether the pairs are duplicate.

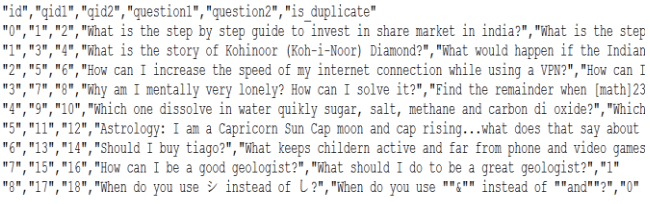
II. EXPERIMENTAL AND COMPUTATIONAL DETAILS

2.1. Data Acquisition and Dataset Description

Our datasets are downloaded from Quora Question Pairs Competition, Kaggle. The training dataset contains a total number of 404,290 valid question pairs. In addition, each row has column labels: "id", "qid1", "qid2", "question1", "question2", "is_duplicate". As a comparison, the test dataset contains totally 2,345,796 question pairs but without any

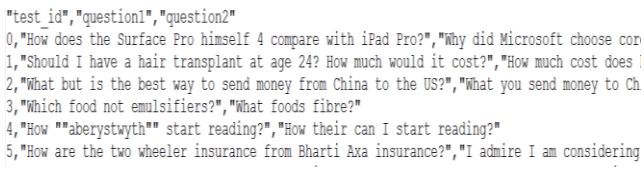
"is_duplicate" label. This is the dataset for conducting further predictions on.

The sample screenshots of the datasets are shown in Fig.1.



```
"id","qid1","qid2","question1","question2","is_duplicate"
"0","1","2","What is the step by step guide to invest in share market in india?","What is the step
"1","3","4","What is the story of Kohinoor (Koh-i-Noor) Diamond?","What would happen if the Indian
"2","5","6","How can I increase the speed of my internet connection while using a VPN?","How can I
"3","7","8","Why am I mentally very lonely? How can I solve it?","Find the remainder when [math]23
"4","9","10","Which one dissolve in water quikly sugar, salt, methane and carbon di oxide?","Which
"5","11","12","Astrology: I am a Capricorn Sun Cap moon and cap rising...what does that say about
"6","13","14","Should I buy tiago?","What keeps childern active and far from phone and video games
"7","15","16","How can I be a good geologist?","What should I do to be a great geologist?","1"
"8","17","18","When do you use シ instead of し?","When do you use ""&"" instead of ""and""?","0"
```

(a) train



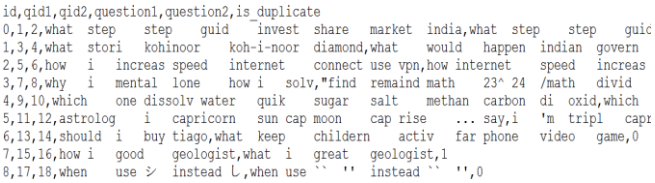
```
"test_id","question1","question2"
0,"How does the Surface Pro himself 4 compare with iPad Pro?","Why did Microsoft choose cor
1,"Should I have a hair transplant at age 24? How much would it cost?","How much cost does
2,"What but is the best way to send money from China to the US?","What you send money to Ch
3,"Which food not emulsifiers?","What foods fibre?"
4,"How ""aberystwyth"" start reading?","How their can I start reading?"
5,"How are the two wheeler insurance from Bharti Axa insurance?","I admire I am considering
```

(b) test

Fig. 1 Screenshots of the original datasets

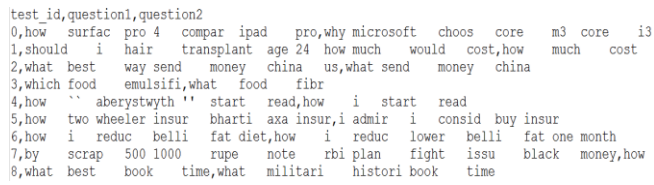
2.2. Data Preprocessing and Word Stemming

To build the word list for each question for training the Word2Vec model, as well as to speed up the training process of the classifier, we only select the first 100,000 rows in the training set and the first 20,000 rows in the test set, then stem the words in both the selected training and test datasets. See Fig.2 the sample screenshots for the stemmed datasets.



```
id,qid1,qid2,question1,question2,is_duplicate
0,1,2,what  step    step    guid    invest  share   market  india,what  step    step    guic
1,3,4,what  stori   kohinoor    koh-i-noor  diamond,what    would   happen  indian  govern
2,5,6,how   i   increas speed   internet    connect use vpn,how internet    speed   increas
3,7,8,why   i   mental  lone    how i   solv,"find  remaind math    23^ 24  /math   divid
4,9,10,which    one dissolv water   quik    sugar   salt    methan  carbon  di  oxid,which
5,11,12,astrolog    i   capricorn   sun cap moon    cap rise    ... say,i   'm  tripl   capr
6,13,14,should  i   buy tiago,what  keep    childern    activ   far phone   video   game,0
7,15,16,how i   good    geologist,what  i   great   geologist,1
8,17,18,when    use シ  instead し,when use ``  ''  instead ``  '',0
```

(a) train



```
test_id,question1,question2
0,how   surfac  pro 4   compar  ipad    pro,why microsoft   choos   core    m3  core    i3
1,should    i   hair    transplant  age 24  how much    would   cost,how    much    cost
2,what  best    way send    money   china   us,what send    money   china
3,which food    emulsifi,what   food    fibr
4,how   ``  aberystwyth ''  start   read,how    i   start   read
5,how   two wheeler insur   bharti  axa insur,i admir   i   consid  buy insur
6,how   i   reduc   belli   fat diet,how    i   reduc   lower   belli   fat one month
7,by    scrap   500 1000    rupe    note    rbi plan    fight   issu    black   money,how
8,what  best    book    time,what   militari    histori book    time
```

(b) test

Fig. 2 Screenshots of the stemmed datasets

However, to acquire a generalized Word2Vec model, we still stemmed the original training dataset, this stemmed dataset is used for training the Word2Vec model.

2.3. Word2Vec Model

Word2vec is a neural network that processes text. Given a text corpus as input, it returns a set of vector representations of each word in the input. The word vector can be used to figure out the semantic similarity of words by calculating the cosine of two word vectors.

Vectorization of words play a crucial role in both of our approaches. We use the gensim library and training data from Quora to train our own word2vec model for future use.

Word2Vec model performs well when analyzing semantics of single word, but it can not be directly applied to sentence semantics analysis. Because the semantic of a sentence is not simply the sum of words' vectors. The order of words each word's parts of speech both affect the semantic. We implement a new way to represent a sentence based on vectors of words, it will be described in the following paper.

2.4. Sentence2Vec Model

This Sentence2Vec model is our first approach mentioned in the introduction and is the basis for the further classification and prediction in 3.1.

2.4.1 Without TF-IDF Weight

After we acquire the Word2Vec model from 2.3, we come up with our first thought of constructing the vector representations for questions. We simply take the mean of the 300-dimension word vectors which appeared in the word list along each dimension. Thus, the result is also a 300-dimension vector and this is the vector representation for the question. Let's see a simplified question pair: "you and me" and "me and you" in Fig.3.

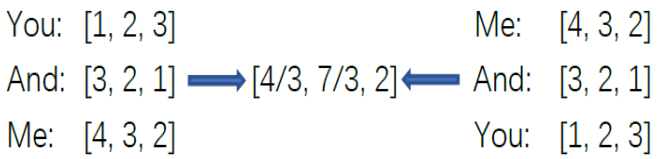


Fig. 3. Simple example

So, disregarding to the word sequences, the two sentences have exactly the same vector representations.

Since the sentence vectors are all normalized, we could easily compute the cosine similarity score between any question pair simply through an inner product. This similarity score we call it "similarity_noidf".

2.4.2 With TF-IDF Weight

Since we simply construct the sentence vector by taking means of each word vectors, two questions with completely different word appearances could collide. Let's see an simple example in Fig.4.

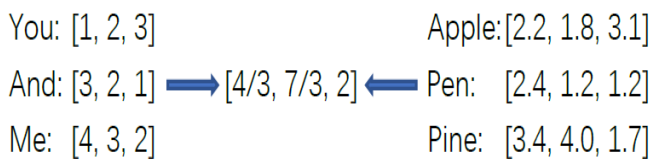


Fig. 4. Another simple example

One way to tackle this issue is to add a TF-IDF weight to each word vector before taking the mean. Note that in our context, the document in term "DF" only refers to the two questions, not all the questions in the training set. Thus, if a word appears in both questions, it's IDF might be zero, and this is bad since it would lead to a zero-similarity score between two identical sentences! So, we add a residual of 0.01 when computing the IDF score. We name the similarity scores from this step "similarity_idf".

It's worth noting that in this step, we also considered common words percentage ("com_perc", number of common words in two questions divided by total length of questions), length difference percentage ("len_diff_perc", difference in length of two questions divided by total length of questions). For example, if the "com_perc" is very large while the "len_diff_perc" is rather small, the possibility of duplicate would increase dramatically.

Finally, we got four features for further training of classifier in section 3.1: "com_perc", "len_diff_perc", "similarity_noidf" and "similarity_idf". See the Fig.5 below for more details.

id	similarity_noidf	is_duplicate	len_diff_perc	com_perc	similarity_idf
0	0.964573119	-1	0.06666667	0.4	0.15058543
1	0.70511319	-1	0.33333333	0.266667	0.01470412
2	0.785734041	-1	0.14285714	0.285714	0.36932708
3	-0.019561515	-1	0.06666667	0	-0.0212053
4	0.648771924	-1	0.29411765	0.176471	0.22158918
5	0.762949112	1	0	0.25	0.17794489
6	0.115340995	-1	0.33333333	0	0.115341
7	0.741760872	1	0	0.25	0.41152183
8	0.709098796	-1	0.16666667	0.25	0.26208913
9	0.691429306	-1	0.06666667	0.266667	0.24916462
10	0.131960224	-1	0.06666667	0	0.13196023

Fig. 5. Feature vector

2.5. LSTM Model Experiment[5]

This LSTM model is the second approach we implement as an experiment to build our own neural network.

2.5.1 Sequence Representation of Questions

The LSTM model requires that the input training data be a sequence of real number. To achieve that, we assign each word a unique id based on the frequency of occurrence of each word, words with same frequency will be randomly ordered. See Fig.6.

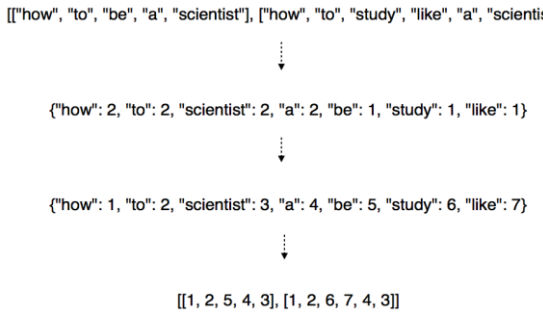


Fig. 6.

2.5.2 Truncate and Pad Input Sequences

Since questions have different length, we need to truncate and pad our input sequences so that they have the same shape. Here we decide to make all sequences to have the same length which is the length of the longest sequence. Shorter sequences will be padded with 0. The model will learn that 0 carries no information. This is also why the id/index we assign to each word must be a non-zero number. See Fig.7

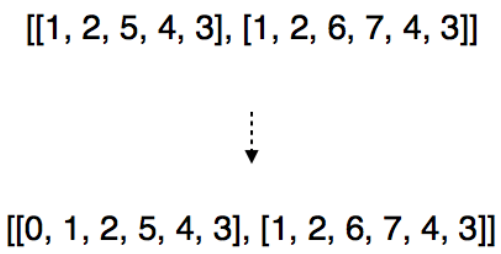


Fig. 7.

2.5.3 Build LSTM Model

After processing input, we need to build our own LSTM model. This is simply done with the provided package in keras and pre-trained word2vec model.

2.5.4 Experiment Results

The LSTM model we build performs well on the data set, it reaches an accuracy of 77%. We expect a better performance with other pre-trained word2vec models based on larger data set. See Fig.8 for more details.

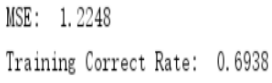
```
Building model...
2017-05-01 09:09:48.086022: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
available on your machine and could speed up CPU computations.
2017-05-01 09:09:48.086077: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
available on your machine and could speed up CPU computations.
2017-05-01 09:09:48.086082: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
on your machine and could speed up CPU computations.
2017-05-01 09:09:48.086087: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
on your machine and could speed up CPU computations.
2017-05-01 09:09:48.086090: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
on your machine and could speed up CPU computations.
quora_lstm.py:84: UserWarning: The `Merge` layer is deprecated and will be removed after 08/2017. Use
`concatenate`, etc.
  model.add(Merge([ques1_enc, ques2_enc], mode="sum"))
Building model costs: 196.7135670185089
Training...
Train on 291088 samples, validate on 32344 samples
Epoch 1/1
Epoch 00000: val_loss improved from inf to 0.48071, saving model to ../data/quora_dul_best_lstm.hdf5
9240s - loss: 0.5052 - acc: 0.7552 - val_loss: 0.4807 - val_acc: 0.7719
Training neural network costs: 9440.463582992554
predict...
80858/80858 [==============================] - 485s
Evaluation...
80858/80858 [==============================] - 473s
Test loss/accuracy final model = 0.4821, 0.7714
80858/80858 [==============================] - 468s
Test loss/accuracy best model = 0.4821, 0.7714
```

Fig. 8.

III. RESULTS AND DISCUSSION

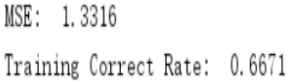
3.1. Classification and Prediction

The labels in our datasets are simply 0/1, so all we need is a classifier which is capable of binary classification, no multiclass needed. So after generating the final training set and testing set, we tried five different classifiers, which are Decision Tree, Random Forest, K Nearest Neighbors, SVM and Adaboost from the scikit-learn library[6]. Each of the trained classifiers is pickled and their MSE and training correct rate is shown in Fig.9

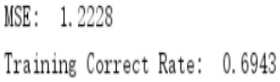


(a) Adaboost

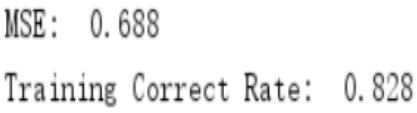
(b) Decision tree



(c) SVM



(d) Random forest



(e) K Nearest Neighbors

Fig. 9 MSE and training correct rate

As we can see from the figure above, the KNN method gives a MSE of 0.688 and a training correct rate of 82.8%, so we choose this trained model for further prediction.

Finally, we use the KNN classifier to predict the "is_duplicate" label for the test dataset and add the label to the original test dataset. See Fig.10 for the details of some of our predictions.

How do I get a job for a earth profile in Canada?	How do I get a wouldn job at .net?	1
Which age is the best age to what get married?	What is the best age to get for a woman?	1
Which mentor the best ways to lose weight?	What has been the most successful way phones lose weight?	1
How does first time sex hasn feel?	How sex first time?	1
How dry I make my website?	How can make website?	1

(a) sample questions whose labels are "1"

Is God facebook?	Does yours god exist?	-1
Why do we have a color?	Are there pte animals that have favorite colors?	-1
What universities is sociobiology?	Are there any sociobiology s courses?	-1
What is the density and volume of contine	Which blogs would stretched space inside of a b	-1
How should I austin about meditating?	Is Abraham for justice?	-1
How can political get MNNIT Allahabad?	How is Allahabad?	-1

(b) sample questions whose labels are "-1"

Fig. 10. Sample predictions

As we can see the overall performance looks good. However, we've noticed that our prediction gets extremely unstable when a single word could affect or even reverse the whole meaning of the question, for example, "lose weight" for a phone or for a person is different. We would discuss this interesting fact in section 4.

3.2. Duplicate Detection Integrated Search Engine

We also build a search engine based on our labeled testing data. It takes two queries at the same time, and return a results of related question pairs as well as the "is_duplicated" label. We indexed the dataset through inverted indexing, and rank the question pairs by their TF-IDF scores.

We also add the hyperlink which directs to the corresponding Quora question page to each result. Fig.11 is the screenshot of our search engine running.

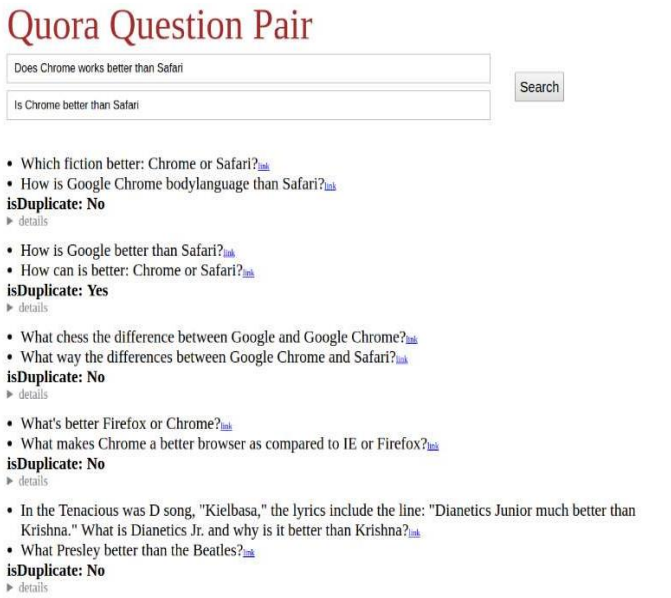


Fig. 11. Search engine running

IV. FUTURE WORK

Our work can be improved in mainly two areas.

In our first approach, we are hoping to build more precise sentence vectors. We need to develop a new method that can take the order of words in a sentence and the parts of speech of each word into account. Feature engineering can also be performed more thoroughly. Besides the cosine distance between two question vectors, we can add euclidean distance, minkowski distance, skewness of vectors, etc.

As for our experiment, same problem exists in word embedding, the order of words and their parts of speech are not taken into account, we need to figure out a way to merge that into our model. Also, we are hoping to study more about neural network in the future because although we successfully build up the LSTM network, we still don't quite understand the mechanism of this neural network. We expect to build a better model with better understanding of neural network.

V. CONCLUSION

In the process of trying to predict whether a question pair is duplicate, we first trained a Word2Vec model to acquire vector

representations for each word as described in section 2.3. Then in section 2.4, we explained how do we build our own Sentence2Vec model with and without taking TF-IDF into consideration. After combine the two methods together, we extract totally four features: “com_perc”, “len_diff_perc”, “similarity_noidf” and “similarity_idf” for the training of classifier in section 3.1.

In section 3, after tried five different classification methods, we choose a KNN classifier which has a MSE of 0.688 as the classification model for prediction on the test dataset. We also analyzed some interesting facts from our predictions. Finally, in section 3.2, we described how we build a search engine based on our labeled test dataset with duplicate detection integrated.

We also discussed several issues we have been faced with and possible ways to improve the performance of our model in section 4.

ACKNOWLEDGMENT

We would like to acknowledge Professor Matt Doherty for giving us support on selecting project topic, fixing technical issues and the techniques in building our models. We would also like to acknowledge the technicians of NYU Linux Servers.

REFERENCES

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

[2] Thanh Ngoc Dao, Troy Simpson. Measuring Similarity Between Sentences.

[3] https://www.tensorflow.org/tutorials/word2vec

[4] https://code.google.com/archive/p/word2vec/

[5] http://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/

[6] http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

[7] http://www.erogol.com/duplicate-question-detection-deep-learning/

[8] https://www.linkedin.com/pulse/duplicate-quora-question-abhishek-thakur

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