DETECTING SIMILARITY BETWEEN QUORA QUESTIONS

Riti Gupta, Sunhera Paul
riti.gupta@sjsu.edu, sunherabarunkumar.paul@sjsu.edu

ABSTRACT

In this work, We have performed a comparative study to detect similarity between questions on Quora using machine learning techniques. The performance between different techniques has been done using metrics like accuracy on test data, ROC and confusion matrix. Some of the models, give the similarity score depending on how close two questions are. While some of the models give the classification.

INTRODUCTION

There are various question and answer portals available on the internet, some of the popular ones being Stack Overflow and Quora. In order to build a high-quality knowledge database, it is important that each question exists on Quora only once. This would enable the readers to find a canonical page for a particular question.

Example of a duplicate question:

- *Why did Trump win the Presidency?*
  How did Donald Trump win the 2016 Presidential Election?

Example of non-duplicate questions:

- *How can I start an online shopping (e-commerce) website?*
  Which web technology is best suitable for building a big E-commerce website?

DATASET

The dataset was publically release by Quora containing actual Quora questions. This has been made at http://qim.fs.quoracdn.net/quora_duplicate_questions.tsv The dataset contains 400,000 lines of potential duplicate question pairs [1] . Each line contains the following information
:

- id: the id of a question pair
- qid1,   unique ids of each question
- question1, question2 - the full text of each question
- is_duplicate - the target variable, set to 1 if question1 and question2 have the same meaning, and 0 otherwise.

The number of samples with duplicate questions are around 250,000 whereas non-duplicate questions are around 150,000.

DATA CLEANING

After analyzing the dataset, we concluded on the following areas for data preprocessing:

1. To not remove the stop words such as ‘what’, ‘which’, ‘how’ as they can impart meaning to a question.
2. To not use stemming on the words as they might change the meaning of the questions.
3. To remove the punctuation as they don’t change the questions
4. Modify the typographical errors in the dataset
5. Modify the abbreviations to their expanded forms
6. Modify the special characters to words
7. Remove comma from between the numbers greater than 999.

FEATURE EXTRACTION

Count Vectorizer: Documents are identified by the word occurrences in them. Count Vectorizer helps in tokenizing a collection of documents and builds a vocabulary of words which is known as the Bag of Words. It represents the words that are present in the questions in our case and the number of occurrences of the words which specify the importance of the words [6].

Tf-Idf: Word counts are a good deciding factor, but they aren’t efficient. For e.g. In a dataset, the word ‘the’ would occur the most number of times but this wouldn’t be very meaningful in determining the similarity between the questions. Thus, we use the concept of Term Frequency-Inverse Document Frequency. The term frequency summarizes the number of times a word has appeared within a question whereas the inverse document frequency downscales the words that have appeared numerous times across questions [7].

Fuzzy string matching is a technique of matching strings where the strings are matched approximately. Thus, even if the user makes a spelling error in one the questions, it would be able to relate to the correctly spelled same word in the other question. This can be seen to be used by Google search, even if we make an error in writing the search query, it searches for the correct representation.

WMD(Word Mover’s Distance) is a method that allows us to assess the “distance” between two documents in a meaningful way, no matter they have or have no words in common. It uses word2vec vector embeddings of words. It measures the dissimilarity between two text documents as the minimum amount of distance that the embedded words of one document need to “travel” to reach the embedded words of another document [8]. Since two questions can have extreme WMD scores that are, large distances, we normalize them into normalized WMD scores as well.

Sentence2vector embeddings: Sentence to vector embeddings are a representation of the meaning of the questions as the whole question in the form of a sentence is taken into consideration and not only the words that are occurring in the question. Word2vec helps in recognizing semantically similar words and sentence2vec helps in understanding the semantic similarity of the questions as a whole.

The other features that are used in the project are:

a. The number of words in question
b. The number of characters in question
c. The length of the common word in the questions
d. The difference in the length of the questions
e. The cosine distance between the vectors of the questions
f. The Manhattan distance between the vectors of the questions
g. The Jaccard distance between the vectors of the questions
h. The Canberra distance between the vectors of the questions
i. The Euclidean distance between the vectors of the questions
j. The Minkowski distance between the vectors of the questions
k. The Bray-Curtis distance between the vectors of the questions
l. The skew and kurtosis of the vectors of the questions

MACHINE LEARNING TECHNIQUES USED

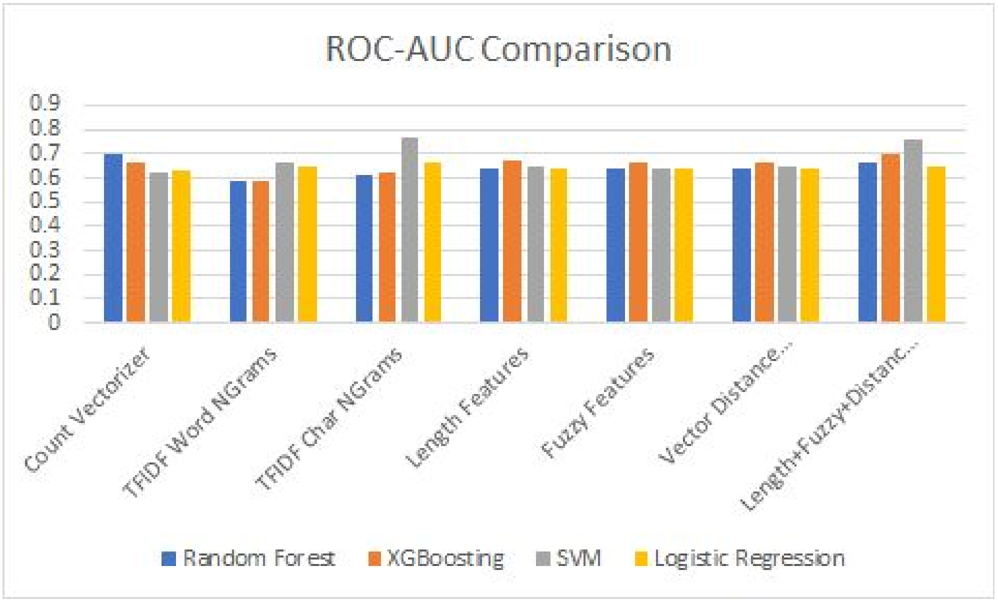
Three machine learning models are implemented:

1. Random Forest Classifier: Random Forest is an ensemble learning method. It builds multiple decision trees on the features available and then merges the results together to get a more accurate prediction.
2. XGBoosting Algorithm: XGB is also an ensemble learning method. It builds trees one at a time, where each new tree helps to correct the errors made by the previous tree.
3. Support Vector Machine: SVM is a supervised learning method which defines a separating hyperplane to classify the dataset given. The algorithm outputs a hyperplane which helps in the categorization of the dataset.
4. Logistic Regression: It is a predictive analysis technique. It helps in identifying the relationship between the independent and the dependent variable. Here to identify if the questions are similar or not, we are using logistic regression for binary classification of the data between identical or not.

Data Splitting: After all the features from the questions were calculated, it was stored in a new file ‘quora_features.csv’. The data was split into the training data used for training the model and testing data used for validation of the model trained. The data was split in a ratio of 0.67:0.33 for training and testing data respectively.

Training Data: XGBoost(XGB) and Random Forest(RF) models were trained on the bag of words obtained from Count Vectorizer and the TF-IDF frequencies. Sets of features were then used to train XGB, RF and SVM models. The models were created first with a set of fuzzy string matching features and then with the set of distances features obtained from the vectors of the questions. Later, both the fuzzy and distance features were combined to create all three machine learning models.

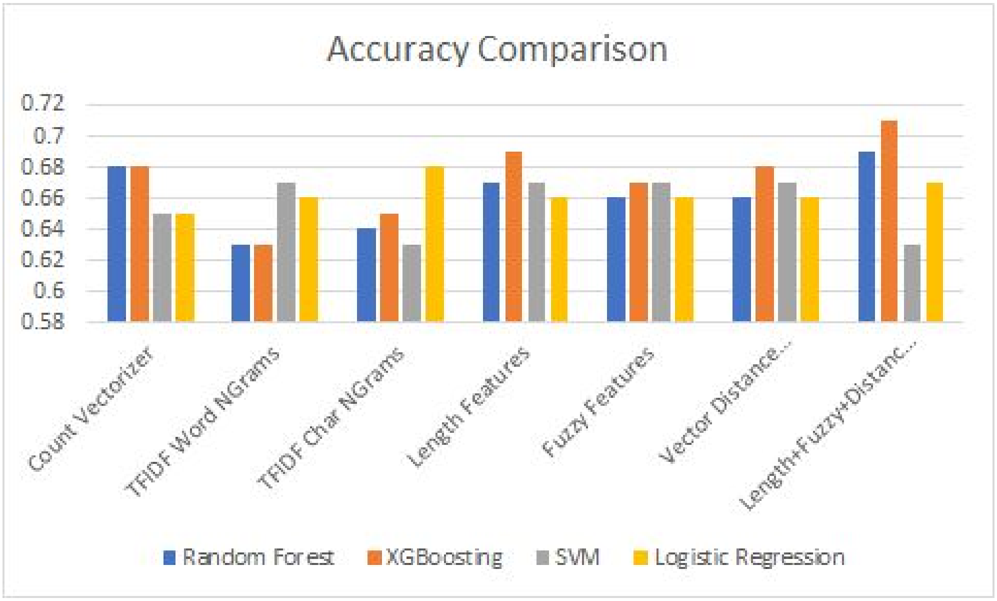
Evaluating the trained model: For every trained model, the testing data was applied to it and was validated. These results were then used to plot a ROC curve and the area under the ROC curve depicted the efficiency of the model. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate(FPR) at various threshold settings. The chart below depicts the various areas under the ROC curves plotted for every model. In the chart, we can observe that we achieve the highest accuracy in comparison to the others using TFIDF Character Ngrams features with SVM and all the engineered features (length+fuzzy+vector distance) with SVM. Hence, we can say that in comparison to the other models SVM gives us a better classification.



The confusion matrix is another evaluation method of the testing data results. It is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. The table below depicts the confusion matrix of the various models trained.

Features	Models	TN	FP	FN	TP
Count Vectorizer	Random Forest	233653	14995	112641	38709
	XGBoosting	209241	39407	89276	62074
	SVM	195989	52659	86486	64864
	Logistic Regression	194245	54402	83347	103574
TFIDF Word NGrams	Random Forest	202964	45684	103574	47776
	XGBoosting	208195	40453	108108	43243
	SVM	225980	22667	109503	22319
	Logistic Regression	226329	22319	38360	38360
TFIDF Char NGrams	Random Forest	208892	39755	47776	47776
	XGBoosting	202266	46381	93112	58238
	SVM	248299	348	147864	3487
	Logistic Regression	215867	32781	94856	56495
Length Features	Random Forest	191456	57192	76024	75326
	XGBoosting	1802040	66608	56146	95204
	SVM	180993	67654	62772	88578
	Logistic Regression	186922	61726	72885	78465
Fuzzy Features	Random Forest	194245	54402	79860	71490
	XGBoosting	179599	69049	61028	90322
	SVM	183086	65562	68003	83347
	Logistic Regression	183784	64864	70095	81255
Vector Distance Features	Random Forest	192851	55797	78116	73234
	XGBoosting	186922	61726	65213	86137
	SVM	184133	64516	67654	83696
	Logistic Regression	187620	61028	73234	78116
Length+Fuzzy+Distance Features	Random Forest	193897	54751	70095	81255
	XGBoosting	185179	63469	50915	100436
	SVM	248300	348	148561	2789
	Logistic Regression	177855	70793	62772	88578

We can find the accuracy of the model from the values of the confusion matrix. Below is a chart of the comparison of the various accuracies that are evaluated from the confusion matrix of the models using different feature sets. We can observe that from this chart we achieve the highest accuracy when we use all the engineered features (length+fuzzy+vector distance) with XGBoosting algorithm.



Thus, from the evaluation using accuracy and ROC-AUC, we can conclude that we get an average accuracy in classifying the questions into whether they are similar or not and that is not enough. However, using deep learning techniques we can obtain higher accuracy.

DEEP LEARNING TECHNIQUES

Data Preparation : We built a vocabulary of all the data present in the form of question pairs. The entire data was encoded as numbers to be used the deep learning model. The sequences were padded with 0.0 to make them of equal length.

Splitting the data : The entire data was split into training and test data, with test data being 33% of the entire data.

Deep Learning Model : We experimented with different Deep Learning Models to get maximum accuracy. The model that gave maximum accuracy included LSTM, Embedding layers(initialized by GloVe embeddings), Convolutional 1D, Dropout Layers and batch normalization [2].

Embedding Layer is used for neural networks on text data. This layer can either be loaded with pretrained vectors or can be learnt from the vocabulary by training the model. We used GloVe embedding with this layer [3].

GloVe (Global Vectors For Word Representation) : It is an unsupervised learning algorithm to obtain vector representation for words. We used pre-trained word vectors for our project [4] [5].

We used LSTM layers in our project. LSTM is a type of RNN which has memory cells and forget gates which enables the model to remember important information in the model. It takes into consideration the context of the entire data being passed over time (not good = bad) which is needed in our project to understand the similarity between two questions.

We applied Convolution1D of width-5 is applied over the data. The number of convolution kernels used were 64. Higher level representation of the data was obtained. They do not have sequence processing ability of LSTMs.

Using GlobalMaxPooling, most important features were selected from the above convoluted vectors. This helped in dimensionality reduction.

To avoid overfitting, Dropout and BatchNormalization techniques were used. Twenty percent of the input layers were dropped at each training epoch to avoid overfitting. The hidden layers were normalized using Batch normalization which reduced the amount by what the hidden unit values shift around. It helped in reducing the overfitting and getting rid of noise.

The activation functions used for output layer was sigmoid which gives the degree of similarity and Relu was used for hidden layers.

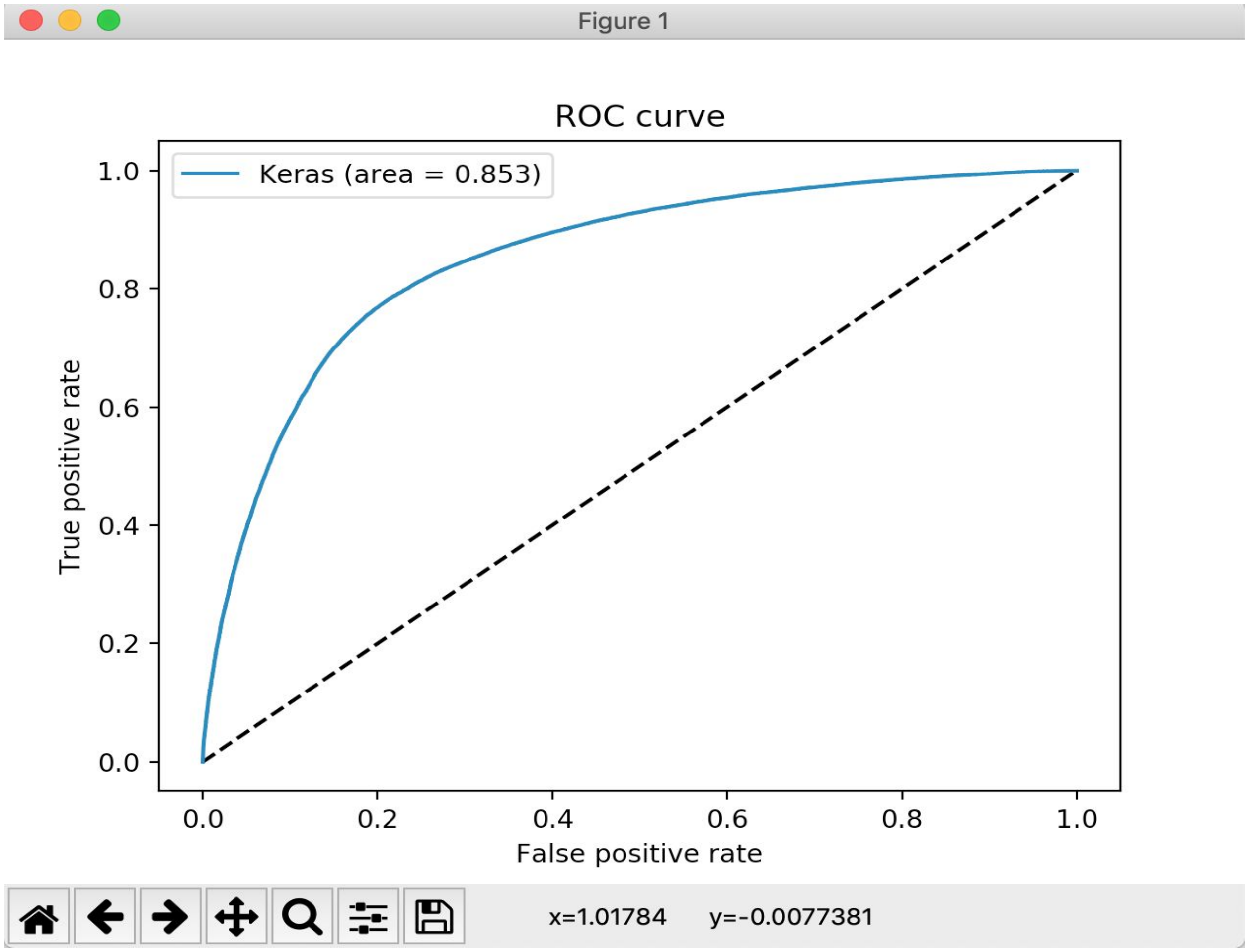
Evaluation of Model :
The time taken to train the model was around 12 hours for 38 epochs.

```
Epoch 36/100
243786/243786 [==============================] - 1505s 6ms/step - loss: 0.0215 - acc:
0.9924 - val_loss: 1.3314 - val_acc: 0.7966

Epoch 00036: val_acc did not improve from 0.80142
Epoch 37/100
243786/243786 [==============================] - 1551s 6ms/step - loss: 0.0207 - acc:
0.9929 - val_loss: 1.3601 - val_acc: 0.8002

Epoch 00037: val_acc did not improve from 0.80142
```

Accuracy achieved :

Loss value = 0.7461
Accuracy on test data = 0.80142

ROC curve :



The results obtained as per ROC-AUC (area being 0.853) look satisfactory and can be expected to give even better results if trained with more epochs.

Confusion Matrix:

We also plotted confusion matrix with various threshold to get some insight into the model predictions.

Threshold	TP	TN	FP	FN	Misclassifications
0.5	69938	35928	14037	13513	27550
0.3	65804	38829	18171	10612	28783
0.7	73284	32204	10691	17237	27928

CONCLUSION

We have used ROC-AUC and Confusion Matrix evaluation to analyze the results obtained from the various machine learning models that have been implemented. We first used count vectorizer as a feature extraction method. Both the XGB and RF models have a ROC-AUC of around 65%. Then we extracted features using the TFIDF method, which also gave us a ROC-AUC of around 65% which is pretty low. Thus, we moved on to implement different types of feature extraction. We used three different types of feature sets, one set consisted of the lengths of the questions and words, the second feature set consisted of the fuzzy string matching features and the third set consisted of the sentence2vector distances features. We implemented machine learning models on each of the feature sets and also on all the features combined. We observed that we get the highest ROC-AUC of 75% for SVM model when we consider all feature sets together. We can use deep learning techniques to take into account the complete structure of the questions which would give higher accuracy and get the similarity between two questions. Thus, to evaluate the percentage of similarity and also to get even better accuracy in the prediction of the results we implement deep learning techniques which gave an accuracy of around 80% on test data.

FUTURE WORK

The accuracy of the network can be further improved by adding more epochs until the accuracy stops improving. More Machine Learning techniques can be explored to further improve the accuracy. We need to have fast machines for doing the same.

REFERENCES

1. https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs
2. https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning
3. https://towardsml.com/2018/06/12/understanding-word-embeddings/
4. https://nlp.stanford.edu/projects/glove/
5. J. Pennington, R. Socher, C. D. Manning : GloVe: Global Vectors for Word Representation, Computer Science Department, Stanford University, Stanford, CA 94305
6. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html
7. http://www.tfidf.com/
8. https://markroxor.github.io/gensim/static/notebooks/WMD_tutorial.html