Measuring Sentences Similarity

Kripanshu Bhargava
Computer Science Department
The University of Texas at Dallas
Dallas, USA
kxb162030@utdallas.edu

***Abstract*—Over 100 million people visit Q&A websites every month, so it's no surprise that many people ask similarly worded questions because of which Q&A websites struggle to keep track of high quality answers and also has adverse impact on their users experience. This project aims at measuring the similarity between two sentences (questions) on a dataset from Quora. It uses both, the semantic and syntactic features along with SVM and Logistic Regression classifiers to resolve this problem. This projects analyzes how syntactic and semantic features work together (and independently).**

I. Introduction

Text (sentence) similarity measure plays an important role in text related research and applications such as information retrieval, text classification, sentence similarity, questions generation, question answering, essay scoring, text summarizing and others. Finding similarity between words is a fundamental part of text similarity which is then used as a primary stage for sentence, paragraph and document similarities. Words can be similar in two ways- lexically and semantically. Words are similar lexically if they have a similar character sequence and structure. Words are similar semantically if they are related to each other, i.e they have the same meaning, are opposite of each other, used in the same way, used in the same context or one is a type of another. One of the earliest works on text similarity used the vectorial model in information retrieval [4]. In most of the works done on text similarity, one of three basic feature types are used - syntactic, semantic or a mixture of both as discussed earlier. Among the literature covered, although limited, [1] and [2] explore semantic features to a great extent whereas [3] explores a mixture of both syntactic and semantic features. In this project, a hybrid approach was chosen using both syntactic and semantic features [5]. Syntactic features from [3] and semantic features from [1] are taken. The report is organized in the following way: Section 2 contains the background of tools and features used. Section 3 describes dataset used in the project. Section 4 describes the approach and usage of syntactic and semantic features. Section 5 describes the implementation of the project. Section 6 describes results and explanation of the measures from features calculation. Section 7 describes conclusion and future work.

II. Tools and Features

This section focuses on details of the tools and features used.

A. WordNet

WordNet is a lexical database in English developed by Princeton University [12]. It is one of the most widely used programmable database by researchers in Natural Language processing. The base of the WordNet is synsets which is the set of synonyms. It contains the short definitions and example of each word (word sense). Along with semantic relations WordNet represent other relations such as hypernyms, antonyms, homonyms, hyponyms etc. In this project we have used WordNet for word sense disambiguation, calculation of similarity features and getting synsets of words when required.

B. NLTK - Natural Language Toolkit

NLTK is a platform which provide researchers with libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries.[10] In this project all the features calculation, tokenization, stemming, tagging and other NLP practices are performed using NLTK.

C. Quora

Quora is a question-and-answer site where questions are asked, answered, edited and organized by its community of users [8]. In addition, as per [9] Quora claimed to have 200 million monthly unique visitors. Although there are no exact statistics for number of questions asked on Quora, the high number of unique visitors makes measuring similarity between the questions asked on Quora an intriguing problem to work on. In addition to this, the fact that the questions are not validated syntactically makes this problem more challenging. As a result, many of the traditional NLP techniques are not applicable directly or have to be modified for the problem.

D. Syntactic Features

Vector Similarity is one of the import aspect of measuring syntactic features. Basically the word is given a value in a corpus and is thus represented as a vector. This vector representation is used to measure the syntactic features for finding how two sentences are similar[6].

1) Cosine Similarity

cosine similarity or normalized dot product is a straight forward method which calculates the cosine angle between two vectors.

$$d_{cosine} = \frac{\sum_{i=1}^{N} A_i \times B_i}{\sqrt{\sum_{i=1}^{N} A_i^2}\sqrt{\sum_{i=1}^{N} B_i^2}} \quad (1)$$

2) Jaccard Similarity

It is another simple method to calculate the similarity and is nothing but a binary operations of intersection and union on tokens which are created from the corpus set.

$$d_{jaccard} = \frac{\sum_{i=1}^{N} (A_i - B_i)^2}{\sum_{i=1}^{N} (A_i)^2 + \sum_{i=1}^{N} (B_i)^2 + \sum_{i=1}^{N} (A_i B_i)} \quad (2)$$

3) Lemma Jaccard Similarity

It is similar to the Jaccard similarity mentioned above just instead of tokens, lemmas are used to form sets.[14]

E. Semantic Features

Semantic features is nothing but degree of synonimity. Over the time many features have been introduced[7]. The basic features are based on taxonomy trees and distance between those words in the tree [10].

1) Path Similarity

It is a measure of similarity between two word concepts, based on the shortest path that connects the concepts in the is-a (hypernym/hypnoym) taxonomy.

2) Wu-Palmer Similarity

It is a measure of similarity between two word senses, based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer (most specific ancestor node).

III. Data

This is a dataset from Kaggle[15] which contains annotated observations from Quora[8]. The dataset consists of two files one for training and another for testing. Both consist of more than 40,000 observations. Each observation of the dataset consisting of a pair of two questions (q_id1 and q_id2) along with a binary value associated with each pair. This binary value represents whether the questions in the pair are similar or not.

IV. Approach

The general approach followed in the project is described in the Fig1 below.

Several approaches have been introduced till now in order to find the similarity between two sentences. The approach revolves around the syntactic and semantic features. For generating the Syntactic features we use the bag of words approach, where we perform pre-process (Fig 1).

We are considering the following syntactic features:

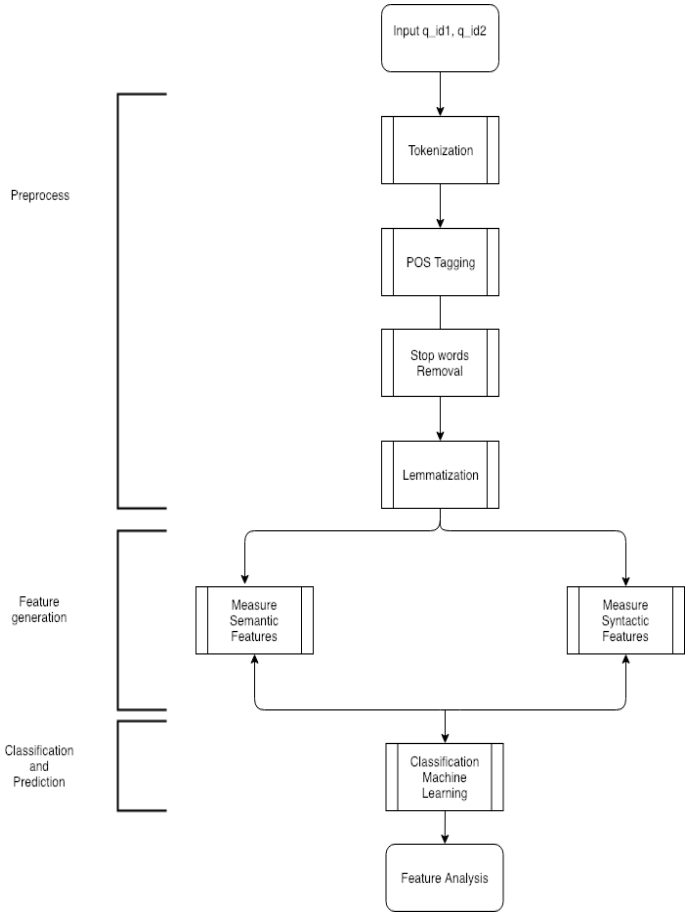

Fig. 1. General flow description of the project.

- **Cosine Similarity:** . It is calculated using the equation given in section II. It uses bag of words generated from the pre-process.
- **Jaccard Distance:** . It is calculated using the equation given in section II. It also uses bag of words generated from the pre-process.
- **Lemma Jaccard distance:** . It is calculated using the equation given in section II for jaccard distance. It uses lemmas (instead of tokens) generated from the pre-process.

For measuring the Semantic similarity, there are many features available. Earlier approach was to use Lin Similarity, Rensik Similarity, Jiang-Conrath Similarity. But later decided to not use them as we require information content (IC) from the external corpus. This project focuses on the semantic parameters such as (**Path Similarity**, **Wu-Palmer Similarity**) mentioned in Section II. Initial approach is to tokenize, POS tag, lemmatize the questions to generate the set. And use this set as an input to the Lesk algorithm. Using the output of the Lesk algorithm, we measure the Semantic feature.

The Lesk algorithm is famous for word sense disambiguation (WSD). The Lesk algorithm uses the WordNet to gather the gloss of all the senses of the word in the sentence and then calculates the maximum overlap. The implementation here is

a bit different from both the paper[2] and the Lesk algorithm. Project adapted Lesk algorithm finds the sense which is best related to the sentence (excluding the stop words)[15]. Later, the semantic features are measured using the word Senses produced.

V. Implementation

The project's implementation is divided into 3 major parts:

*1) **Pre-proecess***

Here we take the input sentences (q_id1, q_id2) and the following operations are performed.

- First, the input strings are **tokenized**, then the **POS tagging** is done. After this **stop words** are removed.
- Once the pre-processing is done, the **lemmatization** is performed depending upon the requirement of the features (not used in syntactic features).

*2) **Features Generation***

Once we get the filtered bag of words, the sets are passed for generation of semantic and syntactic features.

- The semantic and syntactic features returns the generated values.

In this project, Wordnet, Sklearn, numpy and pandas were used in performing calculations and matrix manipulations for the features.

*3) **Machine Learning Classification***

For evaluating how the generated features work, two classifiers are used - Logistic Regression classifier and Support Vector Machine classifier. For this project, we use the Sklearn python library and its implementation of Logistic Regression and SVM. The motivation to choose these two classifiers is as follows:

- Logistic Regression uses a probabilistic approach for classification and SVM does not.
- SVM works better for binary classification with small dataset if compared to other ML algorithms. Thus it was best suited here. Random forest was another option but it is generally used for multiclass classification.

VI. Experiments and Results

For the project, 80% of the training data (due to large size and time consumption) was used to train the model and the resulting model was tested using the testing data. Initially the project did not use lemmatization which led to poor results in measuring semantic features. For ex, the lemma for "better" and "good" is "good", which have same feature values in the sentence, but if we do not lemmatize them then they produce different values in semantic featuring. So to avoid this we used lemmatization.

Also, the POS tagging was done to avoid the confusion when a word can take more than one POS form. Moreover, stopwords removal is done after POS tagging to avoid the ambiguity in the POS tags. These are the main reasons for doing the pre-process and words filtering.

It turned out to that the syntactic features performed well if compared to Semantic features. For syntactic features, all the features together (cosine + jaccard + lemma jaccard) performed well.

For semantic features Path similarity gave the best performance. Although, the results were not up-to the expectations. Modified Lesk algorithm worked better that the simple Lesk but it could have been improved. Together, Syntactic and Semantic features performed well and it was expected due to the approach explained before, although the gain was limited.

Below is the results for the **feature analysis** done to show the performance of the features. The analysis is done using the accuracy, precision, recall and F1 score [16].

TABLE I
SVM Results

Semantic Features	**Metrics (Class1, Class 0)**			
	Accuracy	**Precision**	**Recall**	**F-score**
Path Sim(PS)	0.74	(0.70,0.63)	(0.88,0.35)	(0.78,0.45)
WUP Sim (WS)	0.73	(0.69,0.66)	(0.90,0.31)	(0.78,0.42)
PS+WS	0.74	(0.70,0.65)	(0.88,0.36)	(0.78,0.46)
Syntactic Features	**Metrics (Class1, Class 0)**			
	Accuracy	**Precision**	**Recall**	**F-score**
Cosine Sim(CS)	0.60	(0.60,-)	(-,-)	(0.754,-)
Jaccard (NJ)	0.80	(0.72,0.56)	(0.70,0.58)	(0.71,0.57)
Lemma Jacc. (LJ)	0.81	(0.72,0.56)	(0.69,0.59)	(0.71,0.58)
CS+NJ+LJ	0.82	(0.76,0.61)	(0.79,0.58)	(0.77,0.59)

TABLE II
Logistic Regression Results

Semantic Features	**Metrics (Class1, Class 0)**			
	Accuracy	**Precision**	**Recall**	**F-score**
Path Sim(PS)	0.74	(0.70,0.64)	(0.88,0.36)	(0.78,0.46)
WUP Sim (WS)	0.72	(0.69,0.65)	(0.90,0.30)	(0.78,0.42)
PS+WS	0.74	(0.70,0.65)	(0.88,0.36)	(0.78,0.46)
Syntactic Features	**Metrics (Class1, Class 0)**			
	Accuracy	**Precision**	**Recall**	**F-score**
Cosine Sim(CS)	0.63	(0.63,-)	(-,-)	(0.77,-)
Jaccard(NJ)	0.74	(0.69, 0.57)	(0.82,0.39)	(0.75,0.46)
Lemma Jacc.(LJ)	0.75	(0.71,0.63)	(0.86,0.40)	(0.78,0.49)
CS+NJ+LJ	0.76	(0.72,0.63)	(0.85,0.49)	(0.78,0.51)

However, after all the work is done, during the implementation part there were lot of new factors explored which could have improved the accuracy of this project. The main aim of the project was to find a better way of measuring the similarity and, also to implement the knowledge of NLP in measuring similarity which could have been done using other methods like machine learning or deep learning. Few of the things which might have affected the results were:

- TFIDF: it better works with the large corpus rather than small length dataset.
- Stemming: In place of lemmatization, stemming was initially used. However, lemmatization not only gives

better accuracy, but it also is useful for semantic feature generation when word sense disambiguation needs to be done.
- word2vec also works better with large corpus and was not a useful tool here. Alternative to that was the frequency count of each word in the set of two sentences was used for vector representation.

VII. Conclusion and Future Work

The project described two feature sets (syntactic and semantic) for measuring sentence similarity. Three techniques for syntactic and one algorithm along with three distance metrics for semantic feature generation were used. The results obtained are close to the one's seen in [1], however, semantic features generated failed to capture the context effectively and increase the accuracy of prediction. A maximum accuracy of 82.04% from SVM classifier was achieved by using all the syntactic features together. Although, between the two classifiers, no classifier outperformed the other classifier drastically for all features. The evaluation for cosine similarity failed due to the labeling error in the code. It can be the part of future work to resolve the error. Analyzing the feature value distribution of semantic features, it seems that additional domain knowledge, as suggested in future work might be needed. Finally, the specific task of measuring question similarity resulted in evaluating various features which can be used for other sentence similarity measurement tasks.

Future work, Although in this project features from [1], [2] and [3] are used, there is no significant improvement in accuracy due to the semantic features used. Surprisingly, the syntactic features work better individually as compared to the semantic features.

- K-Beam search for WSD [2] could have been used where the sense that matches the senses of K-nearby words is chosen. This is an improved version of Lesk algorithm.
- As mentioned before that Semantic features such as Lin,Rensik use (IC) for calculating the feature value. It could have been done if we plan to use the external corpus.
- Other machine learning algorithm such as neural network or boosting can be used in feature result evaluation.
- cosine similarity can perform well if the vector given to the function is more informative rather than simply using frequencies (more research work needed).
- Sets of nouns, verbs, adverbs, adjectives could have been generated for each question and then directional measure of similarity [1] could have been used to improve the results.

References

[1] Han, Lushan and Kashyap, Abhay L and Finin, Tim and Mayfield, James and Weese, Jonathan, UMBC EBIQUITY-CORE: Semantic Textual Similarity Systems., NAACL-HLT, 44-52, 2013.

[2] Dao TN, Simpson T, Measuring similarity between sentences, WordNet. Net, Tech. Rep., 2005.

[3] Sravanthi, P., and Srinivase, D.,SEMANTIC SIMILARITY BETWEEN SENTENCES, 2017

[4] Salton, G. and Lesk, M.E., Computer evaluation of indexing and text processing, Journal of the ACM (JACM), 15(1), pp.8-36, 1968.

[5] Hybrid approach-https://drive.google.com/file/d/1kd79of1vrQ–9fSxY9g6hvvduDoPpQIv/view

[6] Choi, S. S., Cha, S. H., Tappert, C. C., A survey of binary similarity and distance measures. Journal of Systemics, Cybernetics and Informatics, 8(1), 43-48, 2010.

[7] Jurafsky, D., Speech and language processing: An introduction to natural language processing. Computational linguistics, and speech recognition, 2000.

[8] https://en.wikipedia.org/wiki/Quora

[9] https://www.quora.com/How-many-people-use-Quora-7/answer/Adam-DAngelo

[10] WordNet NLTK documentation -http://www.nltk.org/howto/wordnet.html

[11] Documents similarity- http://text2vec.org/similarity.html

[12] George A. Miller, A Lexical Database for English, Communications of the ACM Vol. 38, No. 11: 39-41, 1995.

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

[14] lemma jaccard- https://jktauber.com/2017/07/29/nt-book-similarity-jaccard-distance-lemma-sets/

[15] Kaggle Similarity- https://www.kaggle.com/antriksh5235/semantic-similarity-using-wordnet

[16] Result analysis features - https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9