

Quora Semantic Similarity

Using Natural Language Processing to assess text similarity

Project Overview

Problem Set	Task	Action
<p>The dataset provided has a pair of Quora questions in each row. Another column is provided indicating whether the pair is asking the same question (duplicate) or not (non-duplicate).</p>	<p>We want to build a generalizable NLP procedure that will distinguish between the classes and correctly identify whether a question pair is a duplicate.</p>	<ul style="list-style-type: none">- Preprocess texts with tools like tf-idf, word embeddings- Compare numerical vector similarities- Plot to find optimal thresholds- Perform on test data

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when 23^{24} [/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0
...
404007	404346	789792	789793	How many keywords are there in the Racket prog...	How many keywords are there in PERL Programmin...	0
404008	404347	789794	789795	Do you believe there is life after death?	Is it true that there is life after death?	1
404009	404348	789796	789797	What is one coin?	What's this coin?	0
404010	404349	789798	789799	What is the approx annual cost of living while...	I am having little hairfall problem but I want...	0
404011	404350	789800	789801	What is like to have sex with cousin?	What is it like to have sex with your cousin?	0

Models

Overview of potential models

TF-IDF Vector

Universal Sentence Encoder

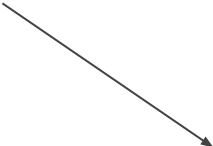
FuzzyWuzzy

TF-IDF: Term Frequency Inverse Data Frequency

Images used from Reference 1 cited on last page.

Sentence 1 : The car is driven on the road.

Sentence 2: The truck is driven on the highway.



	The	car	driven	highway	is	on	road	the	truck
0	0.0	0.043004	0.0	0.000000	0.0	0.0	0.043004	0.0	0.000000
1	0.0	0.000000	0.0	0.043004	0.0	0.0	0.000000	0.0	0.043004

TF-IDF: Term Frequency Inverse Data Frequency

Images used from Reference 1 cited on last page.

$$w_{i,j} = tf_{i,j} \times \log \left(\frac{N}{df_i} \right)$$

$tf_{i,j}$ = number of occurrences of i in j

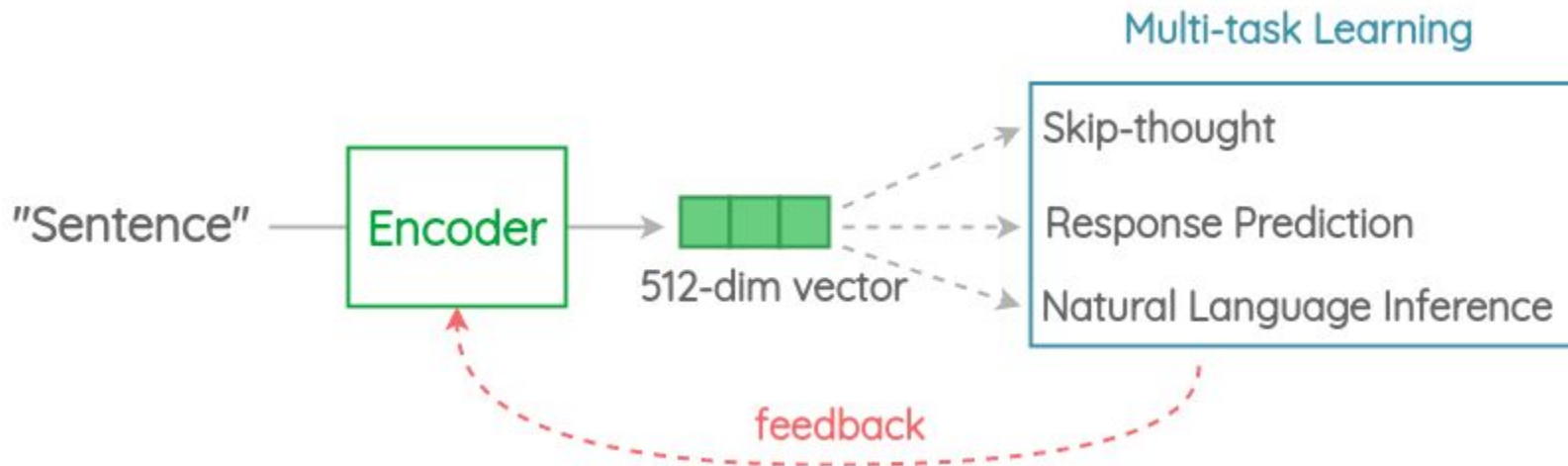
df_i = number of documents containing i

N = total number of documents

Word	TF		IDF	TF*IDF	
	A	B		A	B
The	1/7	1/7	$\log(2/2) = 0$	0	0
Car	1/7	0	$\log(2/1) = 0.3$	0.043	0
Truck	0	1/7	$\log(2/1) = 0.3$	0	0.043
Is	1/7	1/7	$\log(2/2) = 0$	0	0
Driven	1/7	1/7	$\log(2/2) = 0$	0	0
On	1/7	1/7	$\log(2/2) = 0$	0	0
The	1/7	1/7	$\log(2/2) = 0$	0	0
Road	1/7	0	$\log(2/1) = 0.3$	0.043	0
Highway	0	1/7	$\log(2/1) = 0.3$	0	0.043

Universal Sentence Encoder

Images used from Reference 2 cited on last page.



FuzzyWuzzy

Images used from Reference 3 cited on last page.

Levenshtein Distance (LD)

$$\text{lev}(a, b) = \begin{cases} |a| & \text{if } |b| = 0, \\ |b| & \text{if } |a| = 0, \\ \text{lev}(\text{tail}(a), \text{tail}(b)) & \text{if } a[0] = b[0] \\ 1 + \min \begin{cases} \text{lev}(\text{tail}(a), b) \\ \text{lev}(a, \text{tail}(b)) \\ \text{lev}(\text{tail}(a), \text{tail}(b)) \end{cases} & \text{otherwise.} \end{cases}$$

“ Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other ”

Levenshtein score will be between 0 and 100

```
fuzz.token_sort_ratio("fuzzy wuzzy was a bear", "wuzzy fuzzy was a bear")  
100
```


Scores

[illegible]

Model Evaluation

Overview of potential models

TF-IDF Vector

Universal Sentence Encoder

FuzzyWuzzy

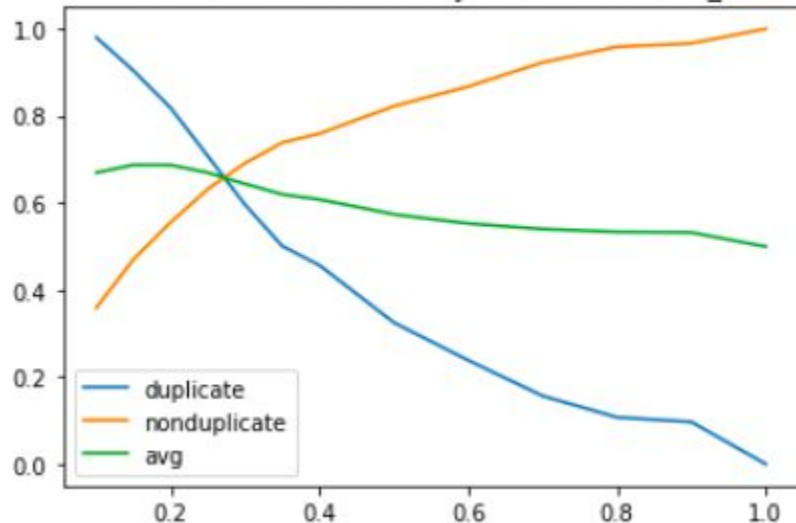
How the Model will be Built

Since we have the different similarity scores, we can determine optimal thresholds for each model.

1. We split the dataset into training and test sets.
2. On the training data, we set a range of threshold values between 0 and 1.
For each threshold, any similarity score above the threshold will be assigned 1 (duplicate) and 0 if equal to or less than the threshold.
3. With the predicted assignments (0 or 1), we plot various thresholds (on x-axis) and its accuracy (y-axis) in predicting the duplicate and non-duplicate class on the training data. We determine the *optimal threshold* value which is where the line plots for the duplicate and non-duplicate accuracies intersect.
4. We will use this optimal threshold value to predict the class on the test set and output the confusion matrix which will show precision, recall and f-1 score for the duplicate and non-duplicates.

TF-IDF Vectorizer Model

Percent of class identified correctly off thresholds: TF_IDF Vector

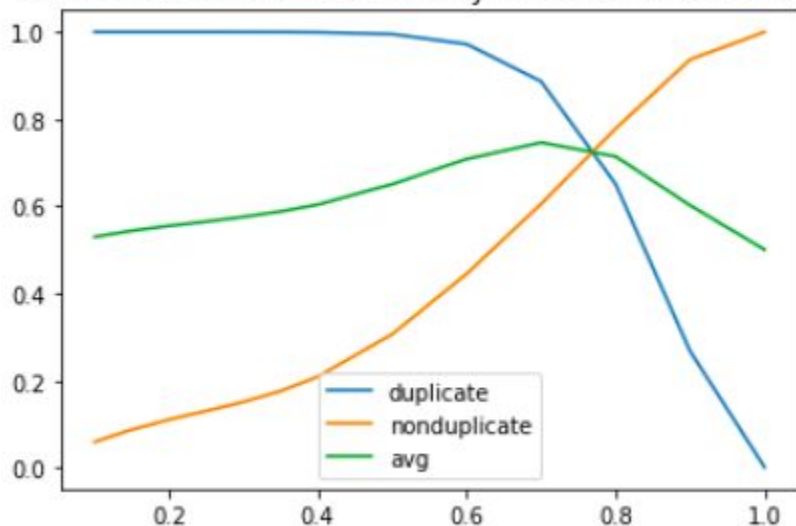


f1-score=0.661

	precision	recall	f1-score	support
0	0.75	0.69	0.72	50967
1	0.53	0.60	0.56	29836
accuracy			0.66	80803
macro avg	0.64	0.64	0.64	80803
weighted avg	0.67	0.66	0.66	80803

Universal Sentence Encoder Model

Percent of class identified correctly off thresholds: USE Metric

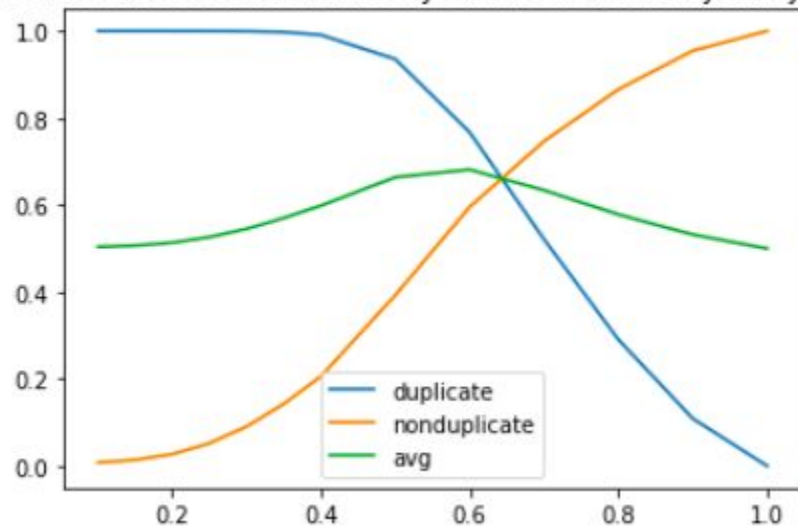


f1-score=0.737

	precision	recall	f1-score	support
0	0.82	0.74	0.78	50967
1	0.62	0.71	0.66	29836
accuracy			0.73	80803
macro avg	0.72	0.73	0.72	80803
weighted avg	0.74	0.73	0.74	80803

FuzzyWuzzy Model

Percent of class identified correctly off thresholds: FuzzyWuzzy Metric



f1-score=0.673

	precision	recall	f1-score	support
0	0.78	0.66	0.72	50967
1	0.54	0.68	0.60	29836
accuracy			0.67	80803
macro avg	0.66	0.67	0.66	80803
weighted avg	0.69	0.67	0.67	80803

Winner is.... Universal Sentence Encoder

- Power network model called *Deep Averaging Network*
- The numerical vectors produced are all 512 dimension and mapped to a vector space where sentences with similar meanings will have vectors that are close in distance

```
f1-score=0.737
      precision    recall  f1-score   support

0         0.82      0.74      0.78     50967
1         0.62      0.71      0.66     29836

 accuracy          0.73     80803
 macro avg         0.72      0.73      0.72     80803
weighted avg         0.74      0.73      0.74     80803
```

Model Deployment

How we would use USE to compare question similarity

1. Input a pair of questions
 2. Turn questions into 512 dimensional word embedding vectors
 3. Compute cosine similarity score between the pair
 4. If score is larger than 0.78 let the user know the questions are duplicates. Otherwise, it is not.
-

Future Improvements

Metric Other than Vector Similarity

For model 1 and model 2, we used the cosine similarity to compute the similarity score between numerical vectors. Maybe we can use K nearest neighbor to better understand if the two questions pairs are close enough in distance to be considered similar. Going forward, I would research into more similarity score options

Data Cleaning

While building the models, I realized that some question pairs were labeled as non-duplicates when they arguably are duplicates. One instance is question id 404350. Though mildly inappropriate, the questions are, “*What is like to have sex with cousin?*” and “*What is it like to have sex with your cousin?*” If we can find instances like this and change the values, the performance of our model could improve.

References

1. TF-IDF Vector Calculation Example for a pair of sentences:
<https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3/>
2. Universal Sentence Encoder Vector Explanation:
<https://amitness.com/2020/06/universal-sentence-encoder/>
3. Fuzzywuzzy Explanation:
<https://www.geeksforgeeks.org/fuzzywuzzy-python-library/#:~:text=FuzzyWuzzy%20is%20a%20library%20of%20Python%20which%20is,a%20service%20to%20find%20sport%20and%20concert%20tickets.>