Final Project

Team members:

1. **Mithil Vijay Gotarne** **mxg180018**
2. **Sreeram Chittela** **sxc180025**

Problem statement

The basic objective of the problem is to identify how similar the given two sentences are. The similarity score takes an integer value between 1 and 5 (included). The higher the score, the more similar the two sentences are.

In general, semantic textual similarity (STS) is a challenging problem; as it requires both an understanding of lexical-level similarity, and the semantic composition of the two chunks of text being analyzed. As a reference, here are some motivating examples:

Sentence 1: Birdie is washing itself in the water basin.
Sentence 2: The bird is bathing in the sink.
Score: 4
Comment: Both sentences convey the message that a bird is taking a bath.

Sentence 1: The young lady enjoys listening to the guitar.
Sentence 2: The young lady enjoys playing the guitar.
Score: 2
Comment: Both sentences involve a lady and a guitar, but convey different actions i.e. listening to the guitar and playing the guitar respectively.

Proposed Solution

Approach 1:

Our proposed solution makes use of the word similarity as measuring the sentence similarity is very closely related to the similarity of the words. While there are myriad number of similarity measures that range from lexical similarity to semantic similarity, we made use of the ones that use semantic similarity.

Steps:

1. We have made use of the Lesk algorithm to identify the best synset given the word and the context.
2. For the obtained best synset, we compute the WuPalmer, Leacock Chordorow, and the path similarity.
3. We have also used the sentence similarity by spacy that converts each sentence into a vector and performs a cosine similarity and returns a value that indices the sentence similarity.
4. Now, we create a 3d matrix with the similarities as features and train our model.
5. We have used SVC, Random forest, and AdaBoost classifiers as models.

Approach 2:

This approach is based on the dependency parse trees. We generate two dependency parse trees and find the similarity for each node in the trees along with the level (Depth) difference as features.

Steps:

1. Generated the dependency parse trees by passing the sentences.
2. For each node in tree1, we iterate recursively over the other tree and keep track of the most similar match and the corresponding level/depth difference.
3. Created the feature matrix accordingly and trained our model.
4. We have used SVC, Random forest, and AdaBoost classifiers as models.

Full implementation details

1. **Programming tools used:**
   a. **Tokenization:** NLTK.
   b. **Lemmatize:** WordNet lemmatizer.
   c. **POS-tagging:** NLTK.
   d. **Dependency Parse Tree:** Spacy.
   e. **WordNet Components:** NLTK WordNet.
   f. **Most Probable Synset:** Lesk.
2. **Results:**
   a. **AdaBoost classifier:**
      i. **Model 1:**

Class	P	R	F1	
micro	0.307		0.15	0.181
10.0	0.0	0.0		
20.163		0.021		0.037
30.357		0.161		0.222
40.342		0.388		0.364
50.674		0.179		0.283

Pearson correlation coefficient: 0.3773903011743604

      ii. **Model 2:**

micro		0.21	0.166	0.13
1	0.0	0.0	0.0	
2	0.286		0.003	0.005
3	0.0	0.0	0.0	
4	0.31	0.687		0.427
5	0.456		0.141	0.216

Pearson correlation coefficient: 0.08211247473194398

b. Multi-Layer perceptron:

i. Model 1:

Class	P	R	F1
micro	0.307	0.15	0.181
10.0	0.0	0.0	
20.163	0.021	0.037	
30.357	0.161	0.222	
40.342	0.388	0.364	
50.674	0.179	0.283	

Pearson correlation coefficient: 0.3773903011743604

ii. Model 2:

micro	0.21	0.166	0.13
1	0.0	0.0	0.0
2	0.286	0.003	0.005
3	0.0	0.0	0.0
4	0.31	0.687	0.427
5	0.456	0.141	0.216

Pearson correlation coefficient: 0.08211247473194398

c. Random Forest classifier:

i. Model 1:

Class	P	R	F1
micro	0.307	0.15	0.181
10.0	0.0	0.0	
20.163	0.021	0.037	
30.357	0.161	0.222	
40.342	0.388	0.364	
50.674	0.179	0.283	

Pearson correlation coefficient: 0.3773903011743604

ii. **Model 2:**

micro	0.21	0.166	0.13
1	0.0	0.0	0.0
2	0.286	0.003	0.005
3	0.0	0.0	0.0
4	0.31	0.687	0.427
5	0.456	0.141	0.216

Pearson correlation coefficient: 0.08211247473194398

3. Problems Encountered:

a. Lesk doesn't always give us perfect synset based on the context.
b. The feature matrix cannot be a matrix in scikit learn machine learning models.
c. Converting feature matrix to low dimensions was expensive and unfruitful.
d. Understanding and formulating math for features.
e. Classes getting zeroes which relates to generalization.
f. Comparing triples from the dependency parse trees at tree levels.

4. Pending Issues:

a. While we used determinant of the matrix to convert the feature into a scalar, the actual mathematical transformation that retains most information are computationally expensive.
b. Combining the pos tag and the dependency to the approach 2 feature space did not help even though they contains critical information.

5. Potential Improvements:

a. Tensors could be used as we are obtaining high dimensional data (4 in our case) and use it as feature space.

b. Figuring out a way to use non uniform dimension data into a uniform dimensional space. For example, in our second case, the matrix length is determined by the words in the sentence which may is not the same across data points.