Similarity Predictor

Overview:

Devise a system that can calculate similarity score between different text chunks. This similarity will also take into consideration contextual overlap apart from text based similarity.

Prerequisites:

- Python IDLE(3.3 or greater)
- NLTK Library along with all its data.
- Flask- application framework written in Python
- Wordnet Thesaurus

** Links and commands for this prerequisites installation are mentioned at last of this document.

The motive of using python language was that it provides a powerful library (NLTK – Natural Language Tool Kit) to deal with English grammar. The NLTK library contains wordnet thesaurus, it is a lexical database for English language.It groups English words into sets of synonyms called 'synsets', provides short definitions and usage examples, and records a number of relations among these synonym sets or their members which is the base for our semantic analysis.

Concept:

For a given pair of text segments:

- First we have remove the stopwords (i.e most frequently occurring word and that generally don't affect the semantic meaning of sentence).
- Then created sets of open-class words, with a separate set created for nouns, verbs, adjectives, and adverbs
- Next , we lemmatize the sets so to be more accurate with replication of word in different form(like go, goes must be treated in same manner)
- Next we try to determine pairs of similar words across the sets corresponding to the same open-class in the two text segments .For nouns and verbs, we use a measure of semantic similarity based on WordNet ,

while for the other word classes we apply lexical matching. The reason behind this decision is the fact that most of the semantic similarity measures apply only to nouns and verbs, and there are only one or two relatedness metrics that can be applied to adjectives and adverbs

- Next we have calculated the cosine similarity that calculates the cosine of angle between text chunks.
- Finally taken average of both variants of similarity calculation to get the final result.

About User Interface and Its Interaction:

- We used the flask framework for interaction of our python code with the html which contain two text box for user input of text chunks.
- Main concept of flask that is responsible for its application is **@app.route** which is a python decorator .It takes the function directly below it and modifies it. We use this to route traffic from a specific URL to the function directly below it .Using different @app.route we can even trigger different parts of code when user visit different part of an application.

Learning:

During the course of development of this system, we got to learn about NLTK library of python which is used in majority of Natural Language Processing (NLP) applications. We also got to learn about various similarity measures used to calculate similarity between two different words. We also got a good knowledge about implementing web interface with python flask.

Results:

The two text chunks given in the document that contain the problem statement (Claim 1 and Claim 11 of US8708211B2) are producing the similarity of 93.36%.

**Installing Procedures for the prerequisites:-

- For installing python visit https://software.rc.fas.harvard.edu/training/scraping/install/
- Flask and nltk can be easily installed using pip:
  - pip install flask

- pip install nltk

- Data and packages that we need from NLTK can be downloaded using following commands on python idle:-
  - import nltk
  - nltk.download()