Semantic Similarity Detection

BTP Advisor: Dr. Amit Praseed.
Project Team:
Hrishabh Pandey (**S20180010064**),
Ayush Gairola (**S20180010s020**),
Rakesh Muchimari (**S20180010109**)
Group Code: B21AP01

Index

Introduction

What is semantic similarity?

The task of measuring sentence similarity is defined as determining how similar the meanings of two sentences are.

Why is it important?

Today we have an abundance of text data, but the utilizing this resource if tricky as Computing sentence similarity is not a trivial task, due to the variability of natural language expressions. If done correctly, this metric could be used to collect, compare and classify large corpus of raw data, which could be utilized efficiently.

About the Project

1. Medical : when a new case comes to an practitioner, he/she can look for similar case in past and get the most closely resembling case and make better decisions.
2. Law : When a new case comes, law firms look over there database for similar cases in past and then strategies from those case studies. (Currently used by some firms)

There examples are a reality but are still kept as an internal tool by organizations, as it brings them a competitive edge.

We wish to present this technology in the hands of general consumer with this platform where individuals and organizations will be able to Collect and Classify Text data, which will accelerate there processes.

Progress Till Now

- Comparison Metric
- Training Data Set
- Methods Utilized

Comparison Metric : Cosine Similarity

Cosine similarity is a measure of **similarity** between two non-zero vectors of an inner product space. It is **defined** to equal the **cosine** of the angle between them, which is also the same as the inner product of the same vectors normalized to both have length 1.

Cos(x, y) = x . y / ||x|| * ||y||

- **x . y** = product (dot) of the vectors ‘x’ and ‘y’.
- **||x||** and **||y||** = length of the two vectors ‘x’ and ‘y’.
- **||x|| * ||y||** = cross product of the two vectors ‘x’ and ‘y’.

Advantages :

- The cosine similarity is beneficial because even if the two similar data objects are far apart by the Euclidean distance because of the size, they could still have a smaller angle between them. Smaller the angle, higher the similarity.
- When plotted on a multi-dimensional space, the cosine similarity captures the orientation (the angle) of the data objects and not the magnitude.

Data Set Utilized

Sick DataSet : Marelli et al.compiled the SICK dataset for sentence level semantic similarity/relatedness in 2014 composed of 10,000 sentence pairs obtained from the ImageFlickr 8 and MSR-Video descriptions dataset. The sentence pairs were derived from image descriptions by various annotators. 750 random sentence pairs from the two datasets were selected,followed by three steps to obtain the final SICK dataset: sentence normalisation, sentence expansion and sentence pairing.

Data Set Utilized

STS DataSet : In order to encourage research in the field of semantic similarity, semantic textual similarity tasks called SemEval have been conducted from 2012. The organizers of the SemEval tasks collected sentences from a wide variety of sources and compiled them to form a benchmark data set against which the performance of the models submitted by the participants in the task was measured

List Of Methods Used

1. Word Count Similarity
2. Tf-Idf Semantic Similarity
3. Word Vector Based Similarity

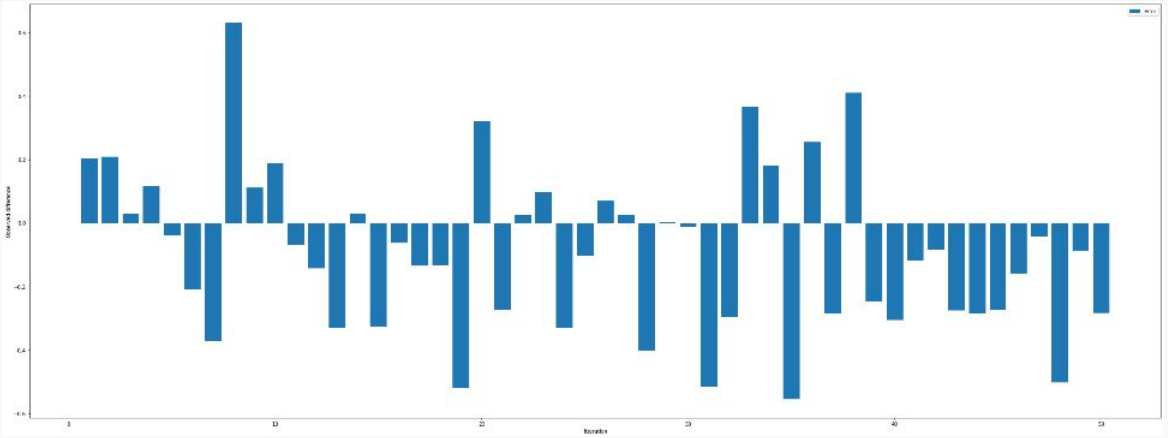
Word Count Semantic Similarity

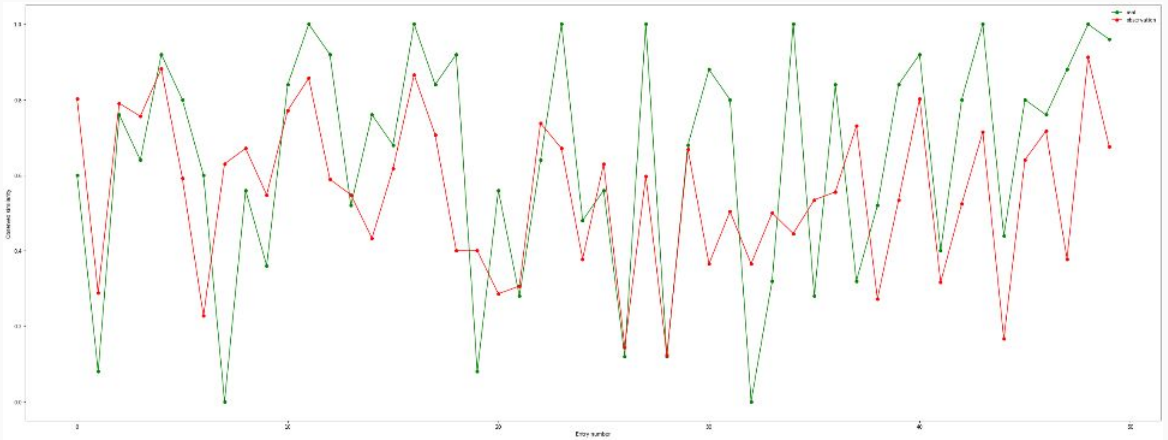
Utilizing the Jaccard similarity index (sometimes called the Jaccard similarity coefficient) compares members for two sets to see which members are shared and which are distinct. It's a measure of similarity for the two sets of data, with a range from 0% to 100%. The higher the percentage, the more similar the two populations.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Observations

Average Difference between Expected and real Comparison
-0.0901593368115178
Variance of this difference Comparison
0.06523802218468647





Tf-Idf Semantic Similarity

TF-IDF stands for "**Term Frequency — Inverse Document Frequency**". This is a technique to quantify a word in documents, we generally compute a weight to each word which signifies the importance of the word in the document and corpus. By vectorizing the documents we can further perform multiple tasks such as finding the relevant documents, ranking, clustering and so on.

Term Frequency : This measures the frequency of a word in a document.

tf(t,d) = count of t in d / number of words in d

Tf-Idf Semantic Similarity (Contd ...)

Document Frequency : This measures the importance of document in whole set of corpus. DF is the number of documents in which the word is present. We consider one occurrence if the term consists in the document at least once, we do not need to know the number of times the term is present. **df(t) = occurrence of t in documents**

Inverse Document Frequency : IDF is the inverse of the document frequency which measures the informativeness of term t. When we calculate IDF, it will be very low for the most occurring words such as stop words (because stop words such as “is” is present in almost all of the documents, and N/df will give a very low value to that word). This finally gives what we want, a relative weightage. **df(t) = N/df**

For large datasets (link corpus size 10,000) the idf term will explode. to dampen the effect we take log of IDF.

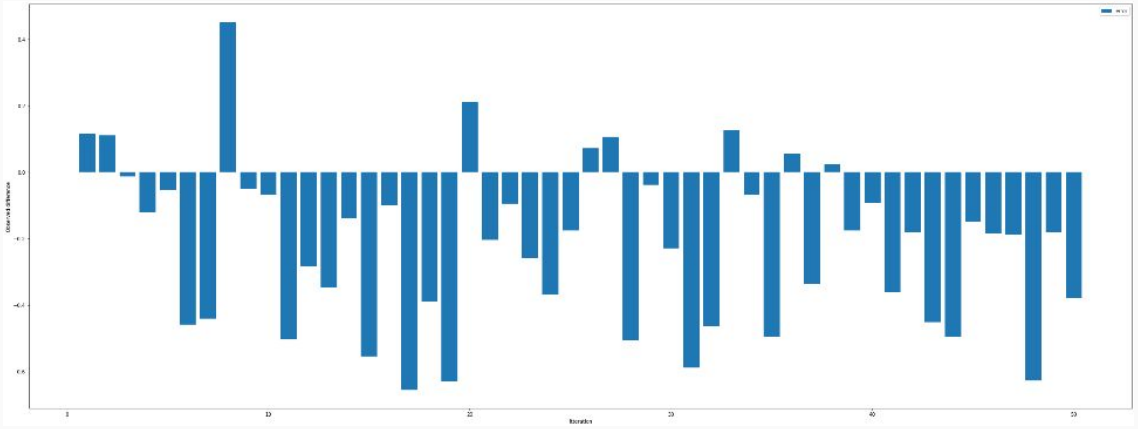
IDF(t) = log(N/(df + 1)) (df+1 to avoid division by zero error)

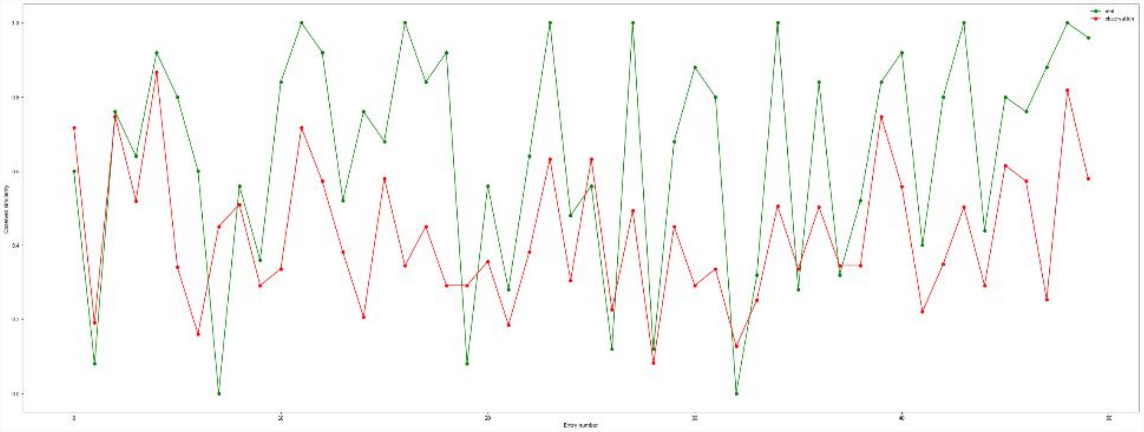
By taking a multiplicative value of TF and IDF, we get the TF-IDF score

TF-IDF(t, d) = tf(t, d) * log(N/(df + 1))

Observations

Average Difference between Expected and real Comparison
-0.18109047743741563
Variance of this difference Comparison
0.050807551174712036





Word Embedders Similarity

Creating sentence representation using word embeddings in vector form and comparing different sentences with cosine similarity.

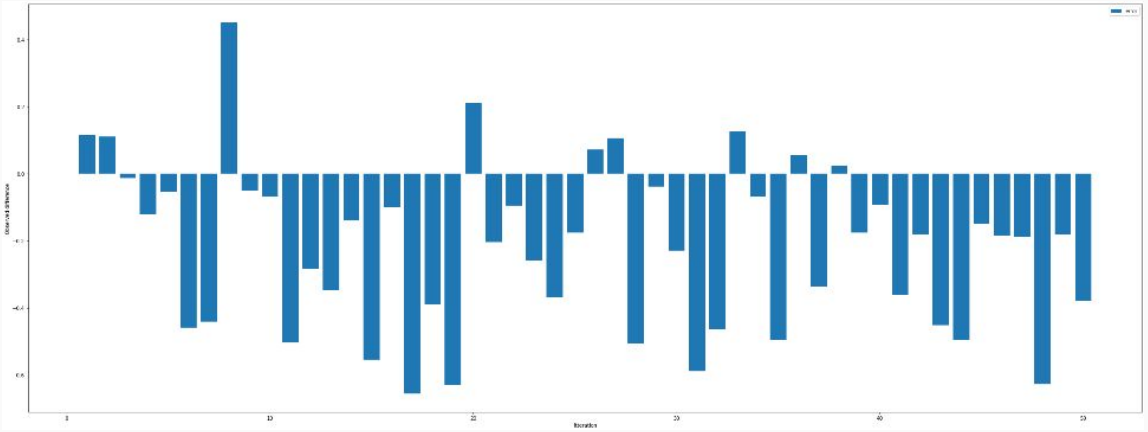
Word embedders are models that convert text to vector representations obtained by training a neural network on a large corpus. It has been widely used in text classification using semantic similarity.

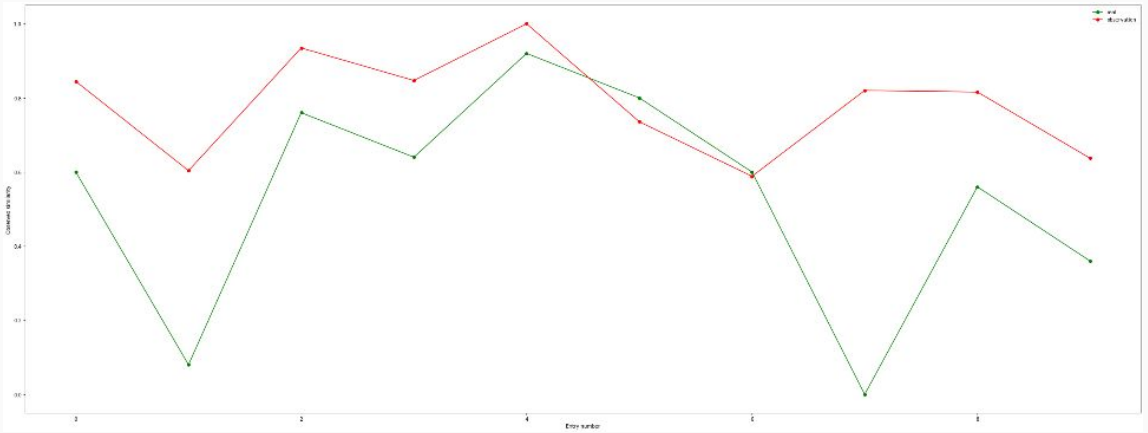
For **example**, ‘apples’ and ‘oranges’ might be regarded as more similar than ‘apples’ and ‘Jupiter’.

Utilizing the word embedder, we create a more robust vectorized representation of the sentence, which is then used for comparison using soft cosine similarity.

Observations

Average Difference between Expected and real Comparison
0.250725989818573
Variance of this difference Comparison
0.060431102208014 25





Future Plans

For 3rd quarter

On Development side

1. Creation of routes, db layer, an user interface

On Comparison Engine side.

1. Experimenting with transformers, and fine tuning them for our use case.
2. Creating server side scripts for clients to train on there dataset as to provide better result.

For 4th quarter

1. Engine and Server side script integration.

References

Thanks to the amazing research community for providing us with these research

- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors forword representation. In Empirical Methods in Natural Language Processing (EMNLP).
- Chandrasekaran, D., & Mago, V. (2020). Domain Specific Complex Sentence (DCSC) Semantic Similarity Dataset. *arXiv preprint arXiv:2010.12637*.
- Marelli, M., Menini, S., Baroni, M., Bentivogli, L., Bernardi, R.,Zamparelli, R., et al., 2014. A SICK cure for the evaluation of compositional distributional semantic models., in: LREC, pp. 216–223.