

# Semantic Similarity Detection

BTP Project By:

Hrishabh Pandey S20180010064

Ayush Gairola S20180010020

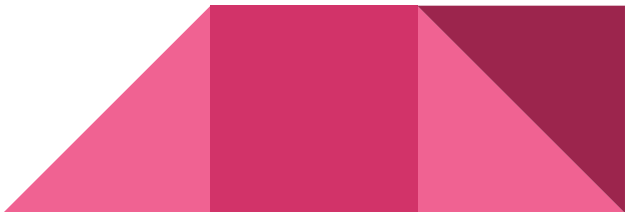
Rakesh Muchimari S20180010109

BTP Code : **B21AP01**

Mentor: Dr. Amit Praseed

# Contents

1. About Project
  - a. What is semantic Similarity
  - b. What is our project
  - c. Overview of Methods and models utilized
2. Semantic Net and Corpus Statistics
3. Sentence Embeddings using Siamese BERT-Network
4. Data Sets Utilized
5. Progress
6. References



# About Project

# What is Semantic Similarity

Semantic Textual Similarity (STS) is defined as the measure of “semantic equivalence” between two blocks of text, phrases, sentences, or documents. Semantic similarity methods usually give a ranking or percentage of similarity between texts.

The main objective Semantic Similarity is to measure the distance between the semantic meanings of a pair of words, phrases, sentences, or documents.



# What is our project ( Deliverables )

For our BTP project, we took inspiration from the below listed implementations which are currently in use.

1. **Customer Support** : Companies can create a corpus of pre seen frequent queries and with our engine they can look for queries which resemble the most similarity, and send an automated response saving lots of unnecessary work-force.
2. **Medical Search Space** : when a new case comes to a practitioner, he/she can look for similar cases in the past and get the most closely resembling case and make better decisions.

We wish to present this technology in the hands of the general consumer with this platform where individuals and organizations will be able to collect and classify text data, which will accelerate their processes.



# How are we going to achieve this.

## **The basic idea is to divide the entire process in 2 steps**

1. Generate Embeddings for sentences
2. Compare those Embeddings

## **Methods for generating Embedding**

1. Using Semantic Nets and Corpus Statistics [1]
2. Sentence Embeddings using Siamese BERT-Networks [3]

## **Methods Used for Comparing Embeddings**

1. Cosine Based similarity





# Semantic Nets and Corpus Statistics

# Introduction

Following are quick points on this method

1. This method dynamically forms a joint word set only using all the distinct words in the pair of sentences creating dynamic vocabulary.
2. A raw semantic vector is derived with the assistance of a lexical database.
3. A word order vector is formed for each sentence, again using information from the lexical database.

Semantic similarity is computed based on the two semantic vectors.





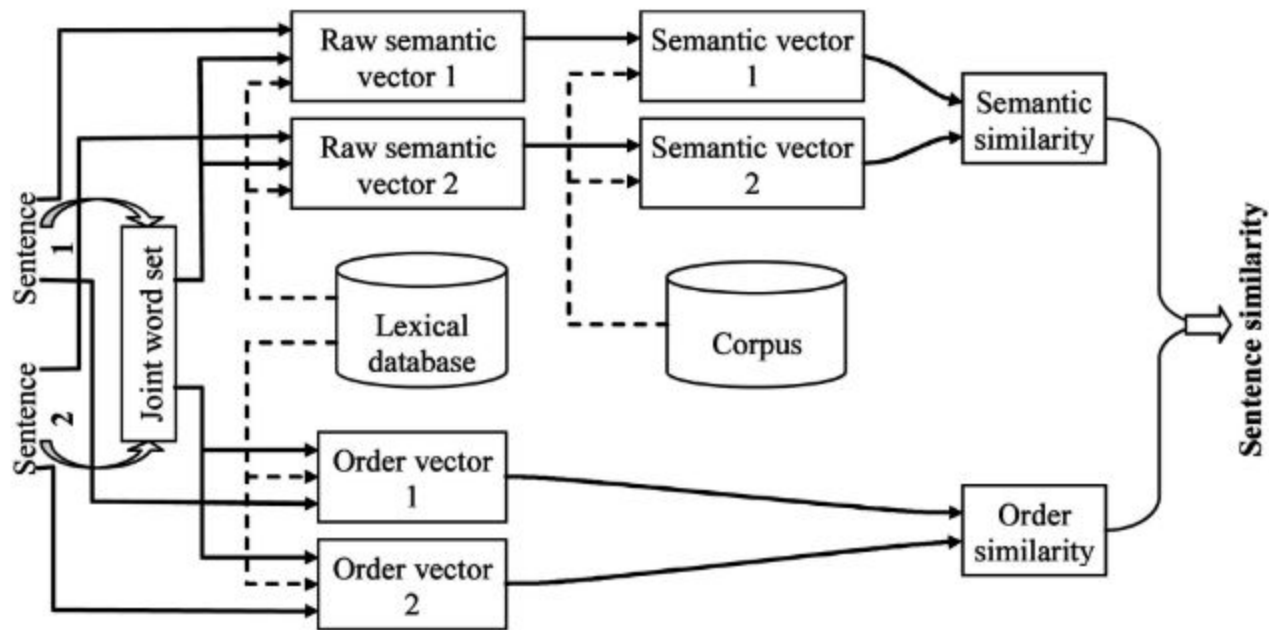


Fig. 1. Sentence similarity computation diagram.

# Semantic Similarity between words

In **WordNet**, words are organized into synonym sets (synsets) in the **knowledge base graph**, with semantics and relation pointers to other synsets.

The similarities  $s(w_1, w_2)$  between words  $w_1$  and  $w_2$  as a function of path length and depth as follows:

- $l$  is the shortest path length between  $w_1$  and  $w_2$ ,
- $h$  is the depth of subsumer in the hierarchical semantic nets.
- $\alpha$  is experimental constants.
- $\beta > 0$  is a smoothing factor and  $\beta \rightarrow \infty$  then the depth of a word in the semantic nets is not considered.

$$s(w_1, w_2) = e^{-\alpha l} \cdot \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}},$$

semantic nets is not

# Semantic Similarity between Sentences

For 2 sentences  $T1$  and  $T2$ , their joined set will be  $T = T1 \cup T2$ .

Since the joint word set is purely derived from the compared sentences, it is compact with no redundant information. The joint word set,  $T$ , can be viewed as the **semantic information** for the compared sentences.

The vector derived from the joint word set is called the **lexical semantic vector**, denoted by  $\tilde{t}$



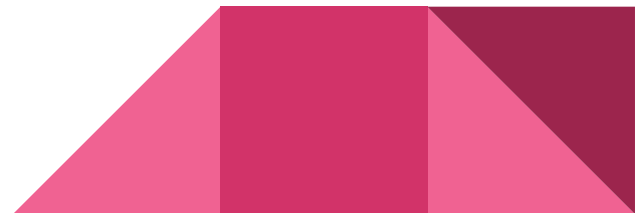
The value of an entry of the lexical semantic vector,  $s_i$  ( $i=1,2,\dots$ ) , is determined by the semantic similarity of the corresponding word to a word in the sentence. Take T1 as an example:

**Case 1.** If  $w_i$  appears in the sentence,  $s_i$  set to 1

**Case 2.** Else, a semantic similarity score is computed between  $w_i$  and each word in the sentence T1, using above method. let call it  $x$ , then  $s_i = ( x > \epsilon ) ? x : 0$

$$S_s = \frac{s_1 \cdot s_2}{\| s_1 \| \cdot \| s_2 \|}.$$

Now that we have the lexical semantic vector, the semantic similarity between two sentences is defined as the cosine coefficient between the two vectors.



# Word Order Similarity between Sentences

Let us consider a pair of sentences, T1 and T2, that contain exactly the same words in the same order with the exception of two words from T1 which occur in the reverse order in T2. For example:

- T1: A quick brown dog jumps over the lazy fox.
- T2: A quick brown fox jumps over the lazy dog.

For the example pair of sentences T1 and T2, the jointword set is:

- $T : \{ A, \text{quick}, \text{brown}, \text{dog}, \text{jumps}, \text{over}, \text{the}, \text{lazy}, \text{fox} \}$



We assign a unique index number for each word in T1 and T2. The index number is simply the order number in which the word appears in the sentence.

indexes : { A : 0 , quick : 1 , brown : 2 , dog : 3 , jumps : 4 , over : 5 , the : 6 , lazy : 7 , fox : 8 }

word order vectors for T1 and T2 are  $r_1$  and  $r_2$ , respectively.

$$S_r = 1 - \frac{\| \mathbf{r}_1 - \mathbf{r}_2 \|}{\| \mathbf{r}_1 + \mathbf{r}_2 \|}.$$

- $r_1 : \{ 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$
- $r_2 : \{ 1, 2, 3, 9, 5, 6, 7, 8, 4 \}$

The measure for measuring the word order similarity of two sentences is:

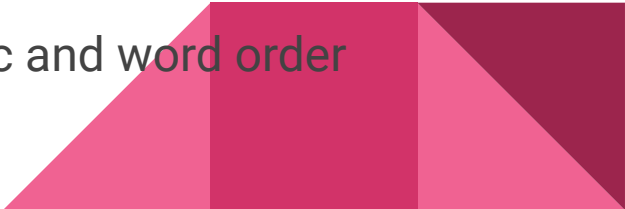


# Overall Sentence Similarity

Both semantic and syntactic information (in terms of word order) play a role in conveying the meaning of sentences. Thus, the overall sentence similarity is defined as a combination of semantic similarity and word order similarity:

$$\begin{aligned} S(T_1, T_2) &= \delta S_s + (1 - \delta) S_r \\ &= \delta \frac{\mathbf{s}_1 \cdot \mathbf{s}_2}{\|\mathbf{s}_1\| \cdot \|\mathbf{s}_2\|} + (1 - \delta) \frac{\|\mathbf{r}_1 - \mathbf{r}_2\|}{\|\mathbf{r}_1 + \mathbf{r}_2\|}, \end{aligned}$$

where  $\delta < 1$  decides the relative contributions of semantic and word order information to the overall similarity computation.



# Sentence Embeddings using Siamese BERT-Networks

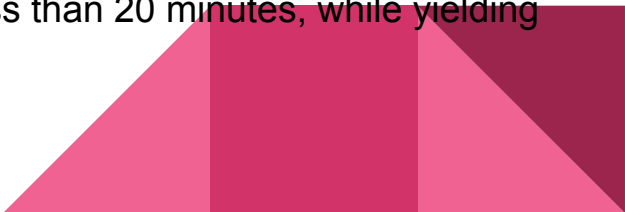


# BERT

We will start with **BERT**. BERT (Devlin et al., 2018) is a pre-trained transformer network (Vaswani et al., 2017), which set for various NLP tasks new state-of-the-art results. The input for BERT for sentence-pair regression consists of the two sentences, separated by a special [SEP] token. Multi-head attention over 12 (base-model) or 24 layers (large-model) is applied and the output is passed to a simple regression function to derive the final label.

A large disadvantage of the BERT network structure is that no independent sentence embeddings are computed, which makes it difficult to derive sentence embeddings from BERT.

we use the pre-trained BERT network and only fine-tune it to yield useful sentence embeddings. This reduces significantly the needed training time: SBERT can be tuned in less than 20 minutes, while yielding better results than comparable sentence embedding methods.



# SentBERT

SBERT adds a pooling operation to the output of BERT to derive a fixed sized sentence embedding. There are many available pooling strategies, but we are going to use the MEAN-strategy.

In order to fine-tune BERT / RoBERTa, we create siamese and triplet networks (Schroff et al.,2015) to update the weights such that the produced sentence embeddings are semantically meaningful and can be compared with cosine-similarity.



we make use of **Regression Objective Function (ROF)** . In ROF The cosine-similarity between the two sentence embeddings  $u$  and  $v$  is computed (Figure 2). We use mean-squared-error loss as the objective function.

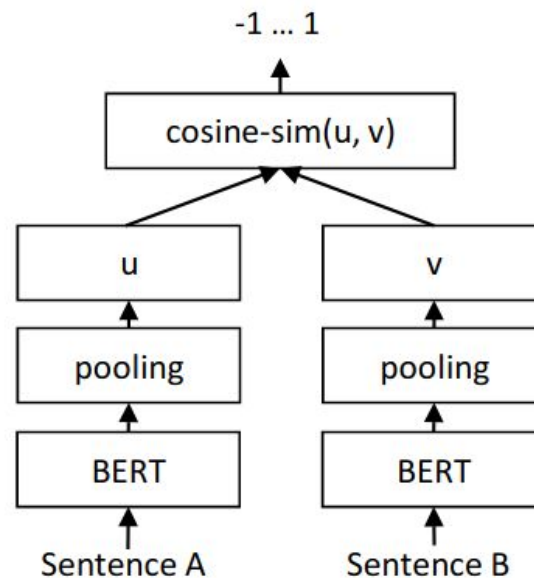


Figure 2: SBERT architecture at inference, for example, to compute similarity scores. This architecture is also used with the regression objective function.

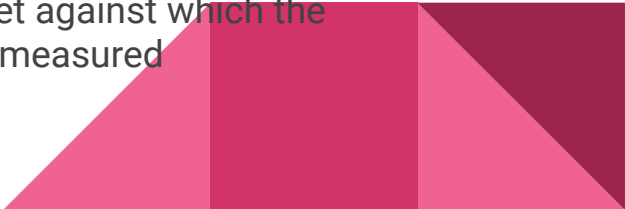
# Datasets Used

## **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 Image Flickr 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.

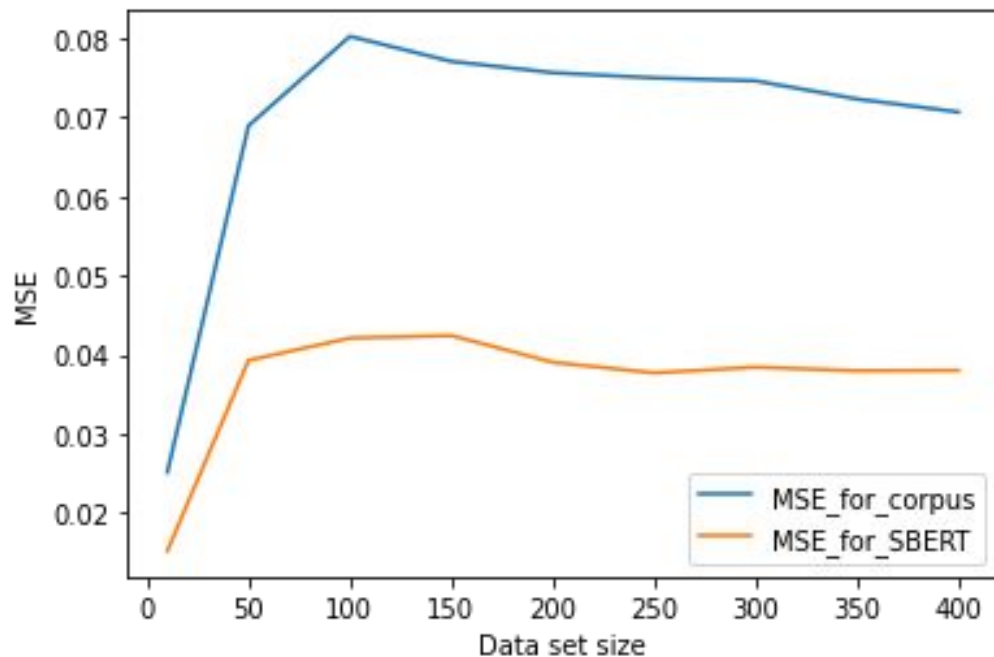
## **STS DataSet**

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



# Experimental Results

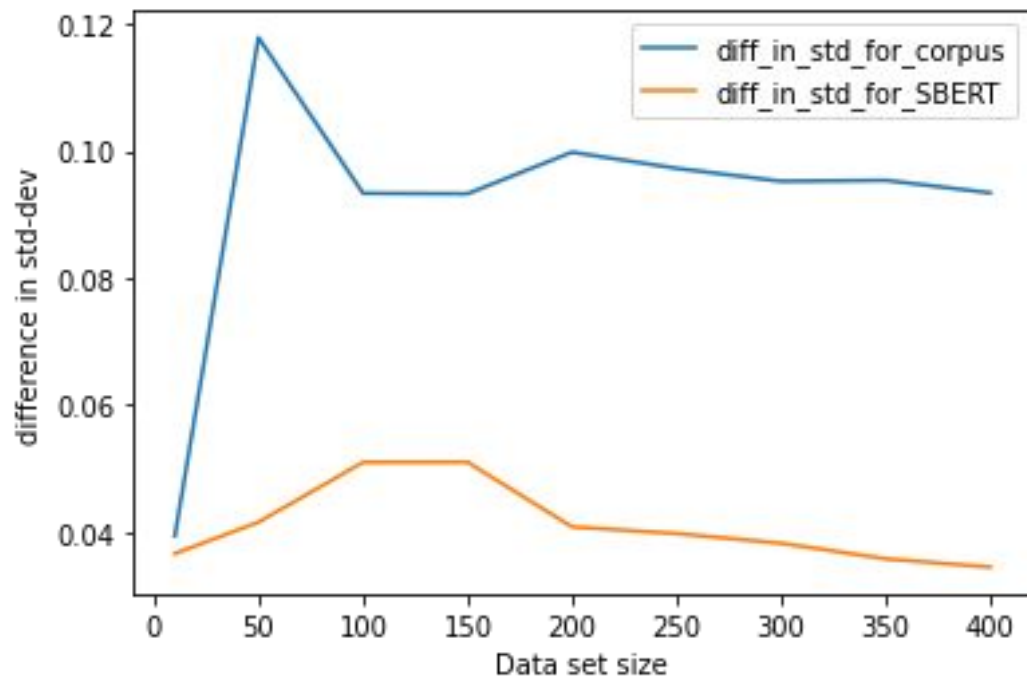
# Mean Squared Error



For test dataset of size 400 **MSE** are observed as below

Corpus Based	SBERT
0.0706958	0.03804137

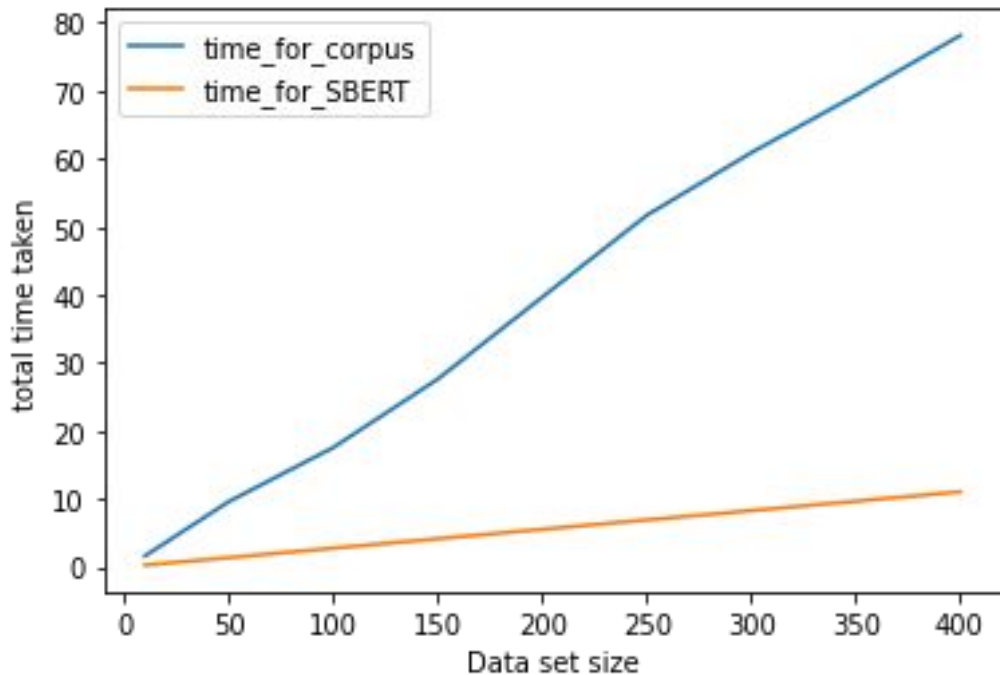
# Difference in Standard Deviation



For test dataset of size 400 STD difference 's are observed as below

Corpus Based	SBERT
0.0933976	0.0345011

# Time for Operation



For test dataset of size 400 Time of operation are observed as below

Corpus Based	SBERT
78.126484	11.055418



# Dev progress

## Front-End

- Home Page
- Authentication
- Comparison Page
- API Docs Page


## Back-End

- DB Layer
- REST API Layer
- Comparison Engine Integration
- Comparison APIs
- Front-end Interactions

## Comparison Engine

1. Corpus Based Algorithm - Embedding generator
2. Sentence Embeddings using Siamese BERT-Network

# References

1. Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea and K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," in IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 8, pp. 1138-1150, Aug. 2006, doi: 10.1109/TKDE.2006.130.
  2. Miller, G.A., 1995. WordNet: a lexical database for English. Communications of the ACM, 38(11), pp.39-41.
  3. Reimers, N. and Gurevych, I., 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.
  4. Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- 

# Thank You.

BTP Project By:

Hrishabh Pandey S20180010064

Ayush Gairola S20180010020

Rakesh Muchimari S20180010109

BTP Code : **B21AP01**

Mentor : Dr. Amit Praseed

