**DJ SYNAPSE: TASK 1(a)**

**NATURAL LANGUAGE PROCESSING**

**TITLE:** Comparing any two given texts and provide a similarity score (Using Cosine Similarity and Word Embeddings).

**WORK PLAN:**

1. Choose the text documents to be compared by the model. They can be two articles describing the same event or reviews for some car model etc.
2. Read the files into two separate lists.
3. Clean the text, i.e remove all digits, special characters and extra spaces.

Also remove all the stopwords.

1. Perform lemmatization on the words.
2. The words are each converted into an vector of m dimensions where each value of the vector contain a score from -1 to +1. We can use google’s Word2Vec open source library for python.
3. Performing SIF on the m dimensional vector.
4. Calculate the Cosine distance between the vectors of the two texts.

Where cosine\_distance = 1 – cosine\_similarity.

**PROJECT PLANNING:**

1. Import all the libraries:

import nltk

import genism

import re

1. Read the text file using manipulation.
2. Remove all the special characters like “\_”, “@” etc from the texts.

Remove the extra spaces and all the stop words.

1. Create a list for all the sentences in the text. Each sentence will be split into a list of words.
2. Perform lemmatization on all the words.

Lemmatization is the process of producing morphological variants of a root/base word of the language. The root word is called lemma. A Lemmatization algorithm reduces the words ‘chocolates’ to the root word ‘chocolate’. NLTK library provides a WordNetLemmatizer.

1. Use the Word2Vec model to convert the words in the cleaned 2D list

The big idea is that you represent words as vectors of features, and compare documents by measuring the distance between these features. There are multiple ways to compute features that capture the semantics of documents and multiple algorithms to capture dependency structure of documents to focus on meanings of documents.

1. Smooth Inverse Frequency:

Taking the average of the word embeddings in a sentence tends to give too much weight to words that are quite irrelevant, semantically speaking. Smooth Inverse Frequency tries to solve this taking weighted average of the word embeddings in the sentence. Every word embedding is weighted by a/(a + p(w)), where a is a parameter that is typically set to 0.001 and p(w) is the estimated frequency of the word in a in corpus(document).

1. Calculating Cosine Distance between the two vectors of the text documents. This will tell us how similar the two texts are.

Mathematically speaking, Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any angle in the interval (0,π] radians. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a cosine similarity of 1, two vectors oriented at 90° relative to each other have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude.