Study on Text Similarity Measure Algorithms For English Language

Sultana Kowser
Student Id: 011141116

Shuchita Rahman
Student Id: 011143047

Dewan Ashikuzzaman Shojol
Student Id: 011143050

Moumita Kabir
Student Id: 011143063

A thesis in the Department of Computer Science and Engineering presented
In partial fulfillment of the requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

United International University
Dhaka, Bangladesh
January, 2019

Declaration

We, Sultana Kowser, Moumita Kabir, Shuchita Rahman and Dewan Ashikuzzaman Shojol declare that this **Study on Text Similarity Measure Algorithms for English Language**, Thesis Title and the work presented in it are our own. We confirm that:

- This work was done wholly or mainly while in candidature for a BSc degree at United International University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at United International University or any other institution, this has been clearly stated.
- Where we have consulted the published work of others, this is always clearly attributed.
- Where we have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely our own work.
- We have acknowledged all main sources of help.
- Where the thesis is based on work done by ourselves jointly with others, we have made clear exactly what was done by others and what we have contributed ourselves.

_____

Sultana Kowser
011141116
Department of Computer Science & Engineering
United International University

_____

Shuchita Rahman
011143047
Department of Computer Science & Engineering
United International University

_____

Dewan Ashikuzzaman Shojol
011143050
Department of Computer Science & Engineering
United International University

_____

Moumita Kabir
011143063
Department of Computer Science & Engineering
United International University

Certificate

I do hereby declare that the research works embodied in this thesis entitled "**Study on Text Similarity Measure Algorithms for English Language**" is the outcome of an original work carried out by Sultana Kowser, Shuchita Rahman, Moumita Kabir and Dewan Ashikuzzaman Shojol under my supervision.

I further certify that the dissertation meets the requirements and the standard for the degree of BSc. in Computer Science and Engineering.

_____

Dr. Mohammad Nurul Huda
Professor & MSCSE Director
Department of Computer Science & Engineering
United International University

Abstract

In this modern era Text Similarity Measurement is one of the most vital concerns. We are constantly driven to improve our system to minimize human error by achieving high accuracy and Machine Learning algorithm is serving the same purpose in many aspects like plagiarism checking, answer script checking, optimized search engine etc. Over last decade there have been significant improvements in this era. This paper is focused to develop a text similarity system which will allow teachers to check a script by comparing with a standard answer registered in the system. The objective is not only limited to reduce the time but also eradicates the possibility of biasness. The book describes the algorithm process and accuracy checking with several examples to ensure a clear understanding.

Acknowledgement

We would like to express our special acknowledgement to the Almighty for enabling us to carry out this research in good health and frame-of-mind.

Our gratitude and appreciation goes to my most venerable supervisor Dr. Mohammad Nurul Huda (Professor & MSCSE Director, Dept of CSE, UIU) who gave us the golden opportunity to carry out our research, which would not come to light without his undivided time, patience and guidance. As a supervisor, his drive towards research and teaching is phenomenal.

We must mention Sheikh Adilina (Lecturer, Dept of CSE, UIU) who has been a continuous inspiration to us and guided us to develop and grow as a better person over the past years.

Using this opportunity, we would like to show my gratitude towards our university United International University (UIU) which is a big milestone in both of our educational and career development. We are also grateful for having a chance to come close to all the finest Faculties, Teaching Assistants (TA), seniors and other working stuffs in the Department of Computer Science and Engineering (CSE), UIU for creating a nice and friendly environment while carrying out our project.

Lastly, we would like to extend a token of gratitude towards my parents, without their supports and prayers, we would not have made as far as we have.

The work is dedicated to Martyrs of Language Movement in 1952.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

The thesis basically outlines a process to check the similarities between two documents in an automated system. The application of this text similarity system can be numerous like checking exam scripts, plagiarism checking etc. In recent world we are constantly driven to improve our system so that less human effort is required at the same time the process itself is efficient enough. To find a continuous improvement process with the assistance of machine learning motivates us to research in this particular field.

1.1 Motivation

Text Similarity Measurement system is an interesting field of machine learning. Students undergo many examinations in their university and the teachers evaluate these students on the basis of their answer scripts. However, this manual process of evaluating exam scripts is not only a time consuming process but also there are some variance exists with one script to another. If there was a process which could give us an instant result by comparing with a standard answer registered in the system, the whole process of answer script checking could be more efficient in terms of accuracy, zero error & biasness. Therefore, this was the motivation for our thesis to establish a system which would provide us with the above mentioned facilities. However, much research has been executed over time in this field to eradicate the human error as much as possible. The main purpose of our study is also to have an effective system to ease the administration task.

1.2 Objectives

The main objective of this thesis study is following:

- To standardize different documents by comparing two text documents by an algorithm process defined system.
- To make future provision of using the same system in Bengali.

- To merge different similarity measure algorithms in order to increase the accuracy between two text documents.

1.3 Organization of the Thesis

The chapter 2 provides a general and fundamental description of Text Similarity Measure Algorithms. Also provides goals and benefits. The most important part of this chapter is Literature Review or Related Works .We explores similarities to these paper which gives inspiration for our work.

The chapter 3 provides each and every methods and techniques we needed for find out Text Similarity, provided specific objectives and an overview of our approach.

This chapter 4 includes Results, findings, discussion of results. (Algorithm, Flowchart, Table for presenting accuracies, bar charts, Pie charts)

This chapter 5 includes Final words and our Further Development or Future Research.

Chapter 2

Background and Related Works

Our main goal is to find out the similarity between two text documents. For every question there is a standard answer. Many students can answer this question in multiple ways. Our goal is to find out the similarity between standard answer and a sample answer. We mainly find out similarity between two documents. Based on the accuracy's percentage the marks for a certain individual shall be determined.

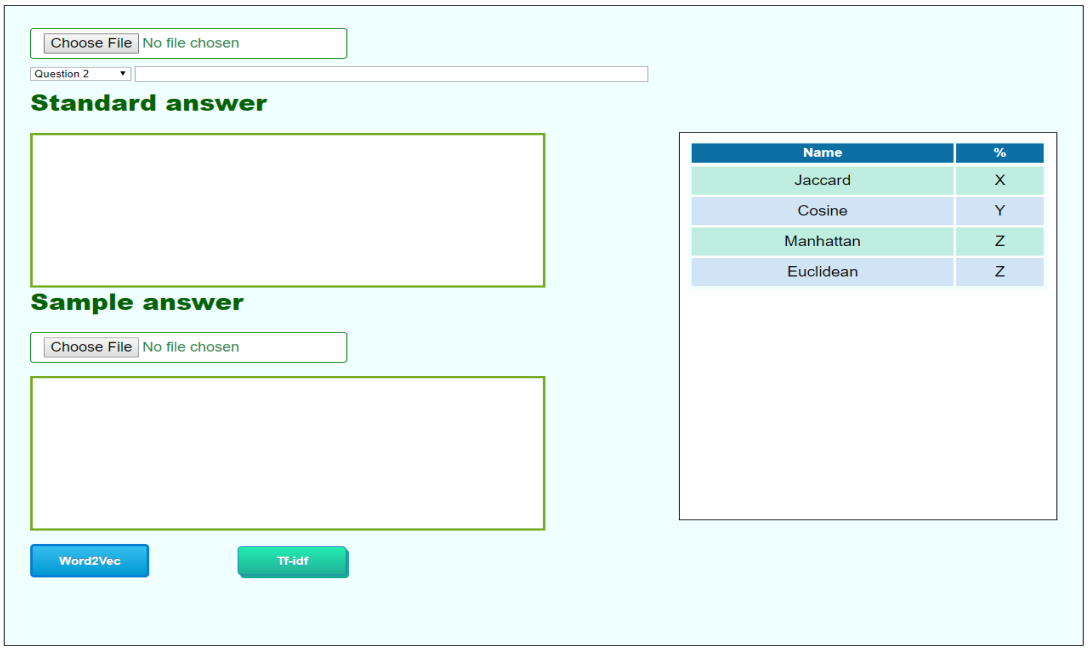

Fig 2.1: Question with standard answer and a sample answer.

From above diagram shows if we choose a question we can automatic find the standard answer. Whenever we select a sample answer and press the button Word2Vector or TF-IDF we will find the similarity of those two text documents. Based on this accuracy's percentage the marks for a certain individual shall be determined. For example, for each question their will be a definite and standard answer that is stored in a text file. Therefore whenever a student will answer the question that will be cross matched with our standard one. Based on those matching the accuracy

of the answer shall be determined. Like of the attempted answer and standard answer match around 50.0%, 90.0%, 65.0% then students will get 5, 9, 6.5 (out of 10) respectively for that particular set of question. We have implemented 5 similarity measure algorithms. So people can use any of them for their desired purpose.

2.1 Fundamental Tools and Techniques of Text Similarity

We communicate using words, a form of unstructured data. Unfortunately, computers suck at working with unstructured data because there is no standard technique to process it. It's not an easy task teaching computers to understand how we communicate. For this reason we have to process this data firstly.

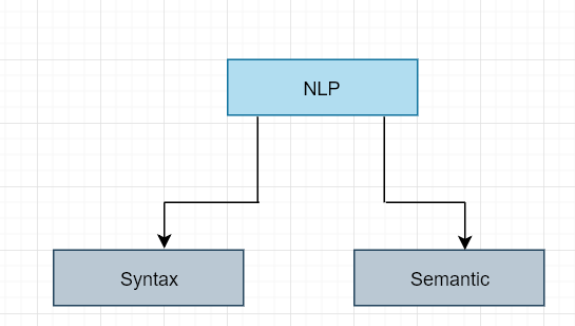

Fig 2.2: Types of NLP

Natural Language Processing is the technology used to help computers to understand the human's structured data. It is a branch of artificial intelligence that deals with the interaction between computers and humans using the natural language. It will process the contents present in standard answer and sample answer. There is lots of way to convert a text to vector. But for our case we have used two methods.

1. TF-IDF
2. Word2Vector

After finding vector we have used 5 similarity algorithms for similarity measurement.

1. **Euclidean Similarity:**

$$d(x,y) = \sqrt{\sum_{i}^{n} (x_i - y_i)^2}$$

2. **Manhattan Similarity:**

|a1-a2|+|b1-b2|

3. **Minkowski Similarity:**

$$\sqrt{\lambda \sum_{b=0}^{n-1} |y_{i,k} - y_{j,k}|^{\lambda}}$$

4. **Cosine Similarity:**
   Sim (A, B) =cos (θ) =A.B/||A||||B||

5. **Jaccard Similarity:**
   J(S, M) = | S∩M |/| S∪M |

2.2 Objectives

Main focus is to find out the similarity between two texts. For one specific question there will be a standard answer and several scripts. We have to find out the similarity between standard answer and one of the scripts. Finally based on accuracy we have to give mark to the script.

2.3 Benefits

It can be a very potential tool as a teacher can easily check all the scripts with in very short time. Also save time as a sample script can be checked within few seconds. The result will accurate and exact. We can say unmistakable result. Sometimes many students suffer or we in other word we can say many students get advantages. So it also ensures 'NO Favoritism'.

2.4 Related works

Over the past several years, several research projects related to text similarity analyses have been implemented. This is one of the necessary research areas in which many papers have already been published.

Semantic similarity approaches have been applied in semantic internet associated applications. There is a novel on calculating the similarity between paragraphs, but there is not several work connected to the calculation of similarity between sentences or texts.

The vector-based approaches are basically used in Information Retrieval (IR) systems to find out similarity, in which by representing a document as a word vector, then queries are matched to associated documents in the document store through a similarity metric.

The Semantic Analysis (LSA) [15] [16] and the Hyperspace Analogues to Language (HAL) model [17] are two very well known approaches in the field of corpus-based similarity.

Maguitman et al. [18] suggested a solution filled gap of how text observations could be used to formulate the related measure accordance with semantic similarity. From information theoretic similarity the semantic similarity between two texts is the degree of the number of meaning shared by meaning of logics.

LSA analyses a huge document storage of natural language information and it provides basically a representation that simply finds the terms and text documents similarity.

Li et al. [19] proposed a very well known hybrid approach that generates text similarity from not only syntactic but also semantic approaches in compared texts. Their technique basically forms a joint word set using all of the specific words in two couple phrases dynamically. For each of the phrase, it basically derived an unrefined semantic vector.

A very well known method proposed by Sahami et al. [20] to measure semantic similarity between two text documents using very small text snippet collections returned for those texts by a search engine. This method solves the problem that existed in earlier methods foe example Cosine similarity that measure the similarity between small texts.

Sanghamitra Bandyopadhay and Koushik Mallick [21] established a new shortest path based hybrid determine of ontological correspondence between couple words which combine the structure of information content of the terms and the Gene Ontology graph. Gene Ontology consists of vocabulary of terms. Here the similarity between two terms X1 and X2 is achieved from the common relations of X1 and X2. Another is generally from remaining relations.

A double-checking method proposed by Chen et al. [22] by applying the term snippets achieved from Web search engine for computing semantic similarity between terms.

Yuncheng Jiang et al. [23] proposed a new similarity computation approach which is based on feature.

Hung Chim and Xiaotie Deng [24] developed a simple way for computing document similarity.

Francine Chen et al. [25] develped Story Link Detection methods that easily determine whether two stories are about the same relations which are generally depends on the cosine similarity measure between this two stories.

Mohamed Ali Hadj Taieb et al. [26] developed a method for calculating semantic similarity. Gomaa and Fahmy [27] presented a survey on multiple methods of semantic similarity.

Short texts similarity has several publications Aminul Islam and Diana Inkpen [28]. This system introduces a method for measuring the semantic similarity of information content using a corpus-proposed a based method of semantic word similarity using both normalized and modified version of the LCS.

Peter D. Turney [29] proposed Latent Relational Analysis (LRA) for calculating semantic similarity between words.

Anna Huang [30] propossed a method to figure out the efficiency of similarity measures in partitioned clustering for text document collection.

Paul Vitanyi [31] suggested a measure for figure out semantic similarity distance without any parameter.

Rudi L. Cilibrasi and Paul M.B. Vitanyi [32] have presented a very new concept. That is similarity of words and sentences using information distance and Kolmogorov complexity.

Danushka Bollegala et al. [33] developed a way to measure semantic similarity between words with the help of search engines.

Lixin Han et al. [34] proposed a way to calculate the semantic similarity from heterogeneous ontology Peipei Xia et al. [35] proposed a method to learn similarities that generally called as cosine similarity ensemble.

Danushka Bollegala, Yutaka Matsuo and Mitsuru Ishizuka [36] proposed an developed method to compute almost semantic similarity using text snippets cut from a search engine for two terms.

Chapter 3

Techniques of Similarity Measurement

Standard Sentence: Mathematics is my favorite subject. There are 3 different sentences inside the table. From following example we can see the standard sentence. A1, A2, A3 are all sample sentences. Standard sentence is about mathematics and favorite subject. There is a big difference between standard sentence and A1. Because A1 is about a pet whose name is Michel. But in A3 there is a similarity. Both sentences are about Mathematics. Last sentence A3's meaning is quite similar to standard sentence.

Table 3.1: Similarity of sentences corresponding to standard sentence

ID	Sample Sentence	Similarity (%)
A1	I love my pet Michel very much.	**0%**
A2	He hates mathematics	**25%**
A3	I always consider mathematics as my favorite subject.	**87.88%**

3.1 Techniques

We can't communicate in "structured data" or we cannot speak binary. We communicate using words, a form of unstructured data. But, computers suck at working with these kind of unstructured data because there is no technique to process it. It's not an easy task learn computers to understand how we generally express or communicate.
For this reason we have to process this data firstly.

3.1.1 Natural Language Processing (NLP)

Natural Language Processing is the technology that is generally used to help computers to understand our structured data. It is a section of artificial intelligence that deals with the interaction between computers and humans using the natural language (NLP). NLP encompasses applying algorithms for extracting the NLP rules so that unstructured data converted into a form that computer can easily understand. After providing text documents, computer will execute the algorithms for extracting the meaning associated with the sentence and find out the important data from them. Syntactic analysis and semantic analysis are the two main techniques used to complete Natural Language Processing tasks.

3.1.2 Techniques used in NLP

There are two type of techniques basically used in NLP.

- Syntax
- Semantic

Syntax Analysis

Syntax refers to the organization of words in a sentence so that these words make only grammatical sense. In NLP, syntactic analysis is used to assess how the natural language regulate with the grammatical rules.

Semantic Analysis

We have used semantic analysis because it refers to the meaning conveyed by a text. Semantic analysis is the most difficult demeanor in the field of NLP. It basically involves algorithm to understand the words. That the reason for using it.

3.2 Conversion of sentence to vector

Whenever computer gets the sentence it's very necessary to find a vector format from this sentence. Whenever we find vector we have to apply similarity measure algorithms. There are lots of way or conversion methods for converting a sentence to vector. Such as:

1. TF-IDF
2. Word to vector
3. Doc to vector
4. Wiki to vector
5. LSTM-RNN
6. Skip gram and so on.

We have used IF-IDF and Word to vector from above methods.

3.2.1 TF-IDF

The term frequency-inverse document frequency (**TF-IDF**) is a very well know method for finding how actually important is a word in a document. In the method a text is represented as a vector. For doing that, we have selected all of text from a specific document called standard document. Convert it to a dimension in the vector space. Similarly we also get a vector from a sample document. But there are some kind of words which are called stop words that are present in both standard document and sample documents, and we're doing is extracting important features from documents, features do identify them among other similar documents, so using terms like "the, is, are, that, at, on", etc.. These aren't going to help us, so simply we'll just ignore them. An example:

A1: I am very glad to know that you got GPA 5:00.

Suppose A1 is selected text from standard document.

A2: I am very happy to know that you got golden A+.

Suppose A2 is selected text from sample document.

Term Frequency (TF)

The terms like “is”, “that”, and “to” will be ignored as cited before. Because this is stop words and we cannot extract any kind of meaningful puss from it. Now that we have a simple index vocabulary, we can convert the standard document set into a vector space where each term of the vector is indexed as our index vocabulary, so the first term of the vector represents the “I” term of our vocabulary, the second represents “very” and so on. Now, we’re going to use the **term-frequency** to represent each term in our vector space; the term-frequency is a measure of how many times the terms present in our vocabulary.

For A1-> (I, very, glad, know, you, got, GPA, 5.00)

For A2-> (I, very, happy, know, you, got, golden, A+)

TF rules:

$$tf(t,d) = \sum fr(x,t)$$

$$fr(x,t) = 1, \text{if x} = \text{t}$$
$$fr(x,t) = 0, \text{otherwise}$$

So it will words as, tf(I, A2), (very, A2) and so on. It is called term-frequency.

Inverse Document Frequency (IDF)

The IDF is a measure of how much information the word provides. It's common or rare across all documents. It is the logarithmically inverse fraction of the documents that contain the word.
IDF rules:

$$\text{idf}(t, D) = \log \frac{N}{|\{d \in xD : t \in d\}|}$$

TF-IDF

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

So by using this TF-IDF method we get a vector which can help to apply similarity algorithm. Final TF-IDF algorithm is given below:

Algorithm: TF-IDF

```
from sklearn.feature_extraction.text import TfidfVectorizer
from mosestokenizer import MosesDetokenizer
from nltk.corpus import stopwords
from math import sqrt
from decimal import Decimal
#open to file in this directory
f1= open('/Users /Desktop/thesis/1.f1', 'r')
f2= open('/Users /Desktop/thesis/2.f2', 'r')
#put file data in data variable
mainData = f1.read()
checkSimilarityData = f2.read()
#remove stop word from this data(am, an etc.)
stop = set(stopwords.words('english'))
mainData = [word for word in mainData.split() if word not in stop]
checkSimilarityData= [word for word in checkSimilarityData.split() if word not in stop]
print(mainData)
print(checkSimilarityData)
#data are tokenized to convert it to vector we must detokenize data or make it a string
detokenize = MosesDetokenizer('en')
mainData = detokenize(mainData)
checkSimilarityData = detokenize(checkSimilarityData)
#printing data whinch are now string
print(mainData)
```

```
print(checkSimilarityData)
#changing sentence to vector
tfidf = TfidfVectorizer()
response = tfidf.fit_transform([mainData, checkSimilarityData])
feature_names = tfidf.get_feature_names()
print(response)
print(feature_names)
#for every word how much similar they are
for col in response.nonzero()[1]:
    print(feature_names[col], ' - ', response[0, col],'-',response[1, col])
#seperate standard answer and sample answer
x=[response.nonzero()[1]][0]
y=[response.nonzero()[0]][0]

standard=[]
sample=[]
j=0
k=0
l=0
for i in y:
   if i == 0:
      standard.insert(j,response[0,x[k]])
    elif i== 1:
      sample.insert(l,response[0,x[k]])
   k=k+1
```

Fig 3.1: Algorithm of TF-IDF

3.2.2 Word to Vector

Vector space mode simply represents words in a vector space where semantically similar type words are mapped to nearby words so easily.

Word to Vector is a bundle of models which helps us to derive interaction between a word and its neighbor words. Suppose the text is: I am very glad.

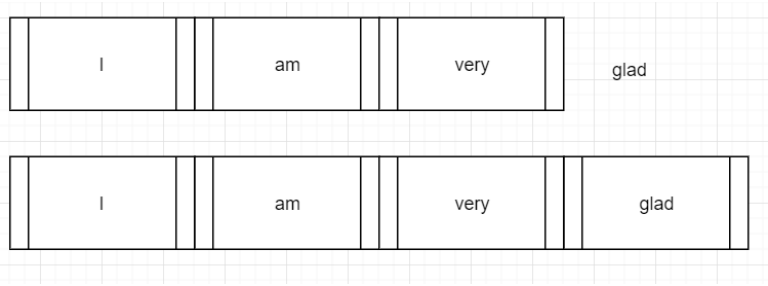

Fig 3.2: Diagram of Word2Vector

So for,

'I' it will check (I, am) and (I, very)

'Am' will check (am, I), (am, very) and (am, glad)

Final Word2Vector algorithm is given below:

Algorithm: Word2Vector

```
#from mosestokenizer import MosesDetokenizer
from nltk.corpus import stopwords
from gensim.models import Word2Vec
import numpy as np
from math import sqrt
from decimal import Decimal
#open to file in this directory
f1= open('/Users /Desktop/thesis/1.f1', 'r')
f2= open('/Users /Desktop/thesis/5.f2', 'r')
#put file data in data variable
mainData = f1.read()
checkSimilarityData = f2.read()
```

```
#remove stop word from this data(am, an etc.)
stop = set(stopwords.words('english'))
mainData = [word for word in mainData.split() if word not in stop]
checkSimilarityData= [word for word in checkSimilarityData.split() if word not in stop]
print(mainData)
print(checkSimilarityData)
model=Word2Vec(mainData,min_count=1);
model2=Word2Vec(checkSimilarityData,min_count=1);
#vocab=model.vocab.keys()
#wordInVocab=len(vocab)
def sent_vector (sentence,model):
    sent_vec=np.zeros(100)
    num_of_word=0
    for w in sentence:
        try:
            sent_vec=np.add(sent_vec,model[w])
                num_of_word+=1
        except:
            pass
    return abs(sent_vec/np.sqrt(sent_vec.dot(sent_vec)))  #normalize sent_vec
  sample=sent_vector(mainData,model)
standard=sent_vector(checkSimilarityData,model2)
for i in range(len(mainData)):
    print(mainData[i],'-',sample[i])
print('----------------------------')
for i in range(len(checkSimilarityData)):
    print(checkSimilarityData[i],'-',standard[i])
```

Fig 3.3: Algorithm of Word2vector

3.3 Similarity Measure Algorithms

Similarity measure means basically how much close two texts are. It is a distance with dimension generally representing features of these kinds of texts. The degree will be high if the distance is

very small but if the distance is large then the degree of similarity will be very low. That's the main concept of similarity algorithms.

After finding vector we have used 5 similarity algorithms for similarity measurement.

1. Euclidean Similarity
2. Manhattan Similarity
3. Minkowski Similarity
4. Cosine Similarity
5. Jaccard Similarity

3.3.1 Euclidean Similarity

Many measures of similarity and also dissimilarity are called Euclidean distance. Suppose X and Y are two vectors. Distance of these two vectors is shown as follows:

$$d(x,y) = \sqrt{\sum_i^n (x_i - y_i)^2}$$

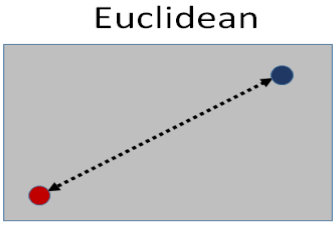

Fig 3.4: Diagram of Euclidean Similarity

So we can easily understand that this distance is the square root of the sum of square differences between tow vectors. Suppose X and Y are two vectors.

3.3.2 Manhattan Similarity

Manhattan distance is the distance between two vectors is the sum of the absolute differences of their Cartesian coordinates. Actually it is the sum of the difference between x and y coordinates. Here, a and b are two vectors. So, a and b are varying in X coordinates and y coordinates.

Manhattan distance=|a1-a2|+|b1-b2|

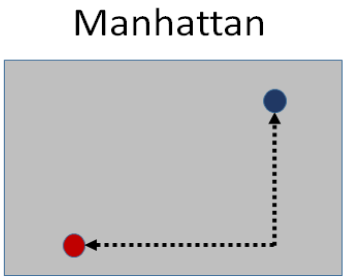

Fig 3.5: Diagram of Manhattan Similarity

3.3.3 Minkowski Similarity

Minkowski is a distance that is a generalized from of Manhattan and Euclidean distance.

$$\sqrt{\lambda \sum_{b=0}^{n-1} |y_{i,k} - y_{j,k}|^{\lambda}}$$

Here, d^MKD = minikowski distance

i, j, k = index of the variable

n= total number of variables y

λ= order of Minkowski metric

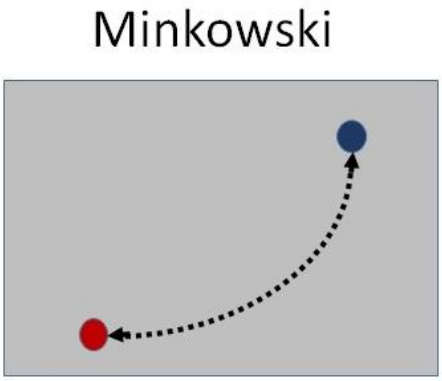

Fig 3.6: Diagram of Minkowski Similarity

3.3.4 Cosine Similarity

Cosine similarity mainly finds the normalized dot products of two vectors. From cosine similarity we find the cosine of the two angles between tow vectors. We know cos0=0 and less than 1 generates other angles. Two vectors which have same orientation similarity of 1, tow vectors which have 90 a similarity of 0.

Sim (A, B) =cos (θ) =A.B/||A||||B||

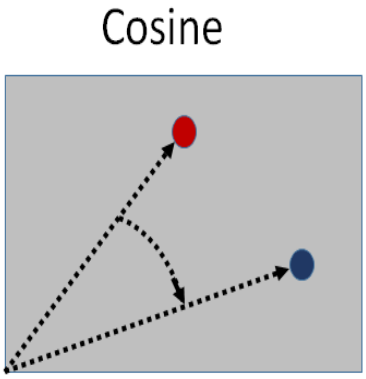

Fig 3.7: Diagram of Cosine Similarity

3.3.5 Jaccard Similarity

For understanding the Jaccard algorithm we need to know set concepts.

Sets

Set: Basically a set is collection of objects.

S = {a, s, d, f, g}

M = {z, x, d , v}

Cardinality: counts how many elements are in S.

|S|=5

|M|=4

Intersection: reveals all objects which are in both sets.

S∩M = {d}

Union: reveals all the items which are in two sets.

S∪M = {a, s, d, f, g, z, x, v}

So the jaccard similarity is

J(S, M) = | S∩M |/| S∪M |

3.4 Using Vector to Implement Similarity Measurement Algorithms

Algorithm is given below:

Algorithm: Similarity Measurement

```
# Euclidean distance:
def euclidean_distance(standard,sample):
  return sqrt (sum(pow(a-b,2) for a,b in zip(standard,sample)))


a=euclidean_distance(standard,sample)
print('euclidean')
print(1-a)
#mahanttan distance
def manhattan_distance(standard,sample):
    return sum(abs(a-b) for a,b in zip(standard,sample))
b=manhattan_distance(standard,sample)
print('manhattan')
print(1-b)
```

```
#minkowski distance
def nth_root(value,n_root):
    root_value=1/float(n_root)
    return round(Decimal(value)** Decimal(root_value),3)
def minkowski_distance(standard,sample):
    return nth_root(sum(pow(abs(a-b),3) for a,b in zip(standard,sample)),3)
b=minkowski_distance(standard,sample)
print('minkowski')
print(1-b)

#cosine distance
def square_rooted(x):
    return round(sqrt(sum(a*a for a in x)),3)
def cosine_distance(standard,sample):
    numerator=sum(a*b for a,b in zip(standard,sample))
    denumerator=square_rooted(standard)*square_rooted(sample)
    return round(numerator/float(denumerator),3)
b=cosine_distance(standard,sample)
print('cosine')
print(b)
#jaccard distance
def jaccard_distance(stndard,sample):
    intersection=len(set.intersection(*[set(standard),set(sample)]))
    union=len(set.union(*[set(standard),set(sample)]))
    return intersection/float(union)
b=jaccard_distance(standard,sample)
print('jaccard')
print(b)
```

Fig 3.8: Algorithm of Similarity Measurement

Chapter 4

Similarity Measurement and Comparison of Accuracy

4.1 Similarity Measure

We have implemented five algorithms using two conversion method TF-IDF and word2vector for similarity measurement.

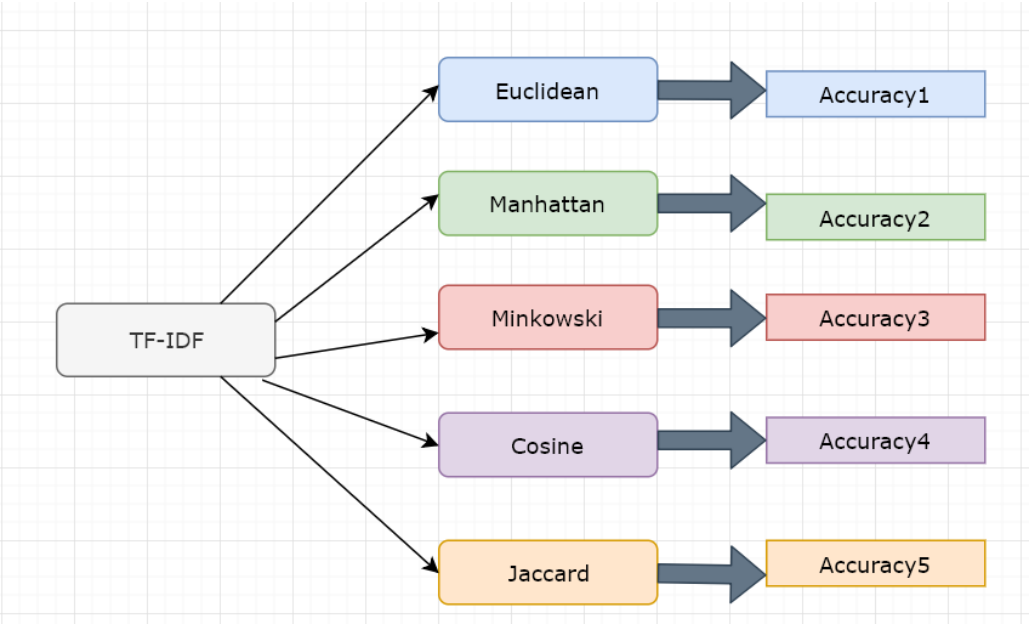

Fig 4.1: Implemented five algorithms using TF-IDF

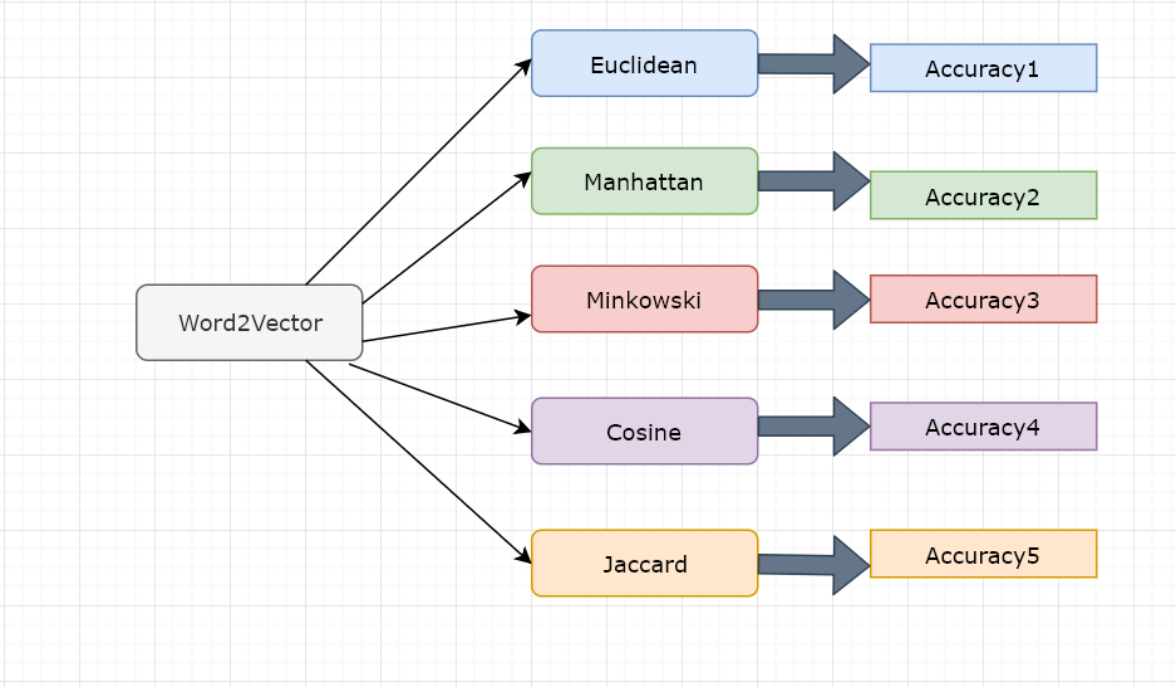

Fig 4.2: Implemented five algorithms using Word2Vector

Basically we input two files. One is Standard document and another is Sample document. Each algorithm gives a different result based on Sample document which depends on Standard document.

4.2 Execution of Algorithm

As we input two files. This algorithm outputs five accuracies.

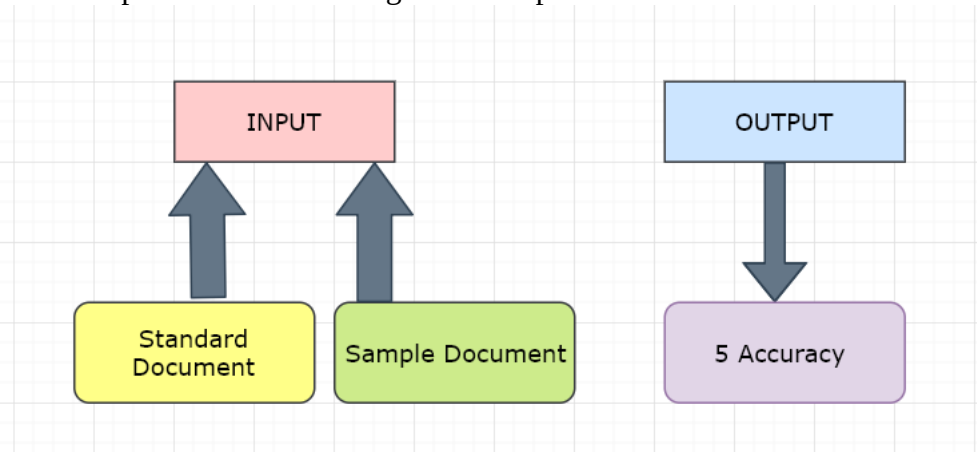

Fig 4.3: Algorithm's Input and Output

4.2.1 Input

As we know we input two documents which contain text. We have to find out how much similar these two documents. Standard document which contains standard text or sentence based on the specific topic. Sample document has various types of text. It can be similar to Standard document or not be similar to the document. Our task is to find out the similarity and show it as a percentage form. The main motive of our thesis is to find out the similarity between different files.

For an Example:

A teacher has a standard answer of a specific question. He has two answer scripts which given by his two students (X and Y).

File 1=> Standard Document (Standard answer)

File 2=> Sample Document1 (similar to Standard document which is X's script)

File 3=> Sample Document2 (not similar to Standard document which is Y's script)

```
A teacher is the person who shapes the future of everyone by providing best education
to her/his students.
Teacher plays a great role in the education of every student.
A good teacher has many qualities and fully able to make his/her students successful
in life.
A teacher is very intelligent and know well that how to draw attention of students
towards study.
She uses creativity while teaching students so that students may concentrate.
Teacher is a good conductor of knowledge having lots of patience and confidence who
take responsibility of the future of students.
Teacher knows the ability of each and every student and tries for them accordingly.
```

Fig 4.4: File1 Standard Document

```
Teacher is a person in our life who provides many important things together
with the good education.
A teacher means a lot to his/her students.
He/she plays extraordinary roles in our lives from the beginning years of
development till we get mature.
They mould us and our future accordingly in order to make us a responsible
citizen of the country.
We have provided below some paragraphs, short essays and long essays on my
favourite teacher to help students in completing their essay writing task.
All the my favourite teacher essay are written very simply.
```

Fig 4.5: File2 Sample Document1

```
A village doctor is a man who treats the patients in the village.
He is a familiar figure.
His work is very important. The village people consult with him for
their disease-related causes.
```

Fig 4.6: File3 Sample Document2

From above three file we can observe very easily that file1 and file2 are quite similar but file1 and file3 are totally different. Now we have to find out similarities.

4.2.2 Executing algorithm

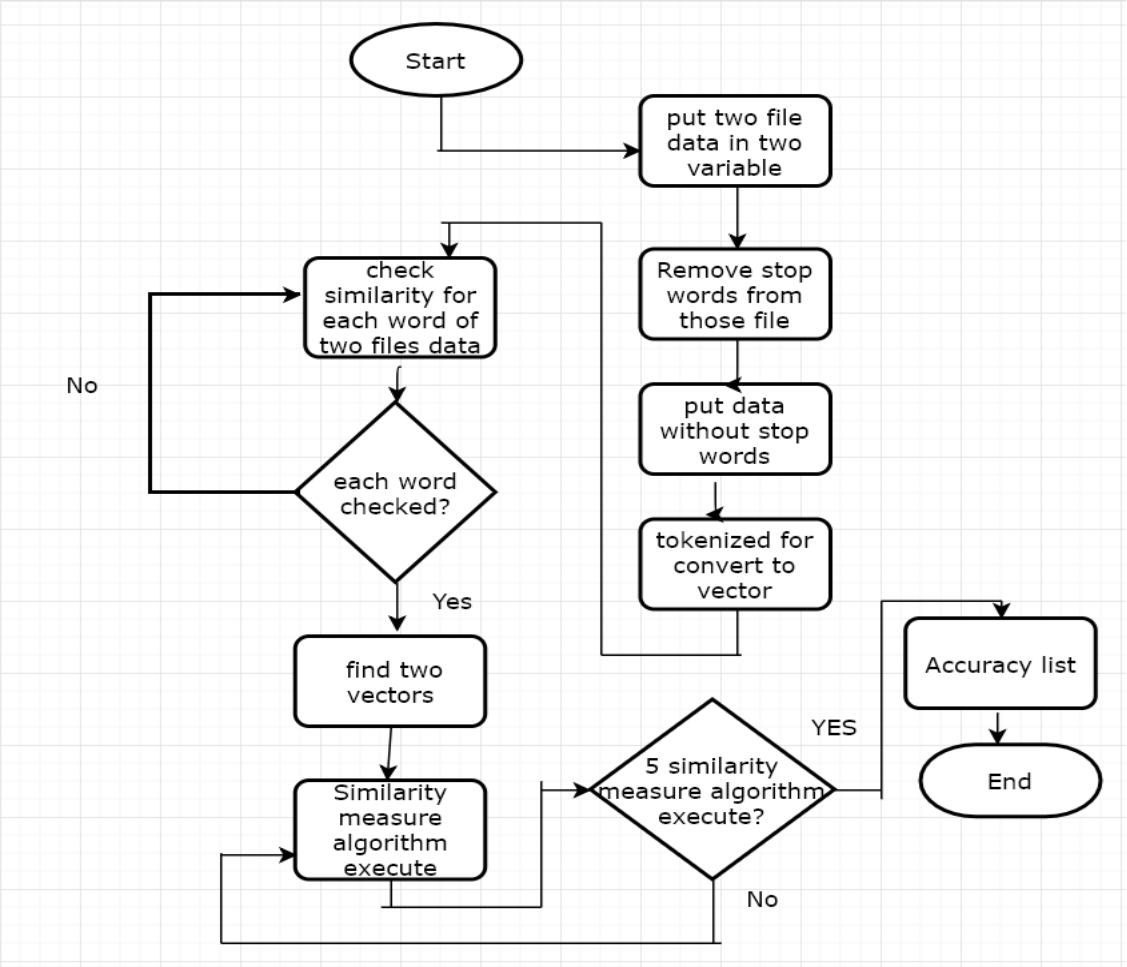

Fig 4.7: Flow chart of the algorithm.

Similarity measurement (File1 and File2)

1. Firstly we take File1, File2 as input. We have to find out the similarity between these two files.
2. All the text or content or data of two files put into two different variables.
3. Remove all the stop words from two files. Such as is, am, are etc. These are not useful for similarity measurement.
4. All data must be tokenized for converting to vector
5. After tokenizing data, we have to de-tokenize data to make it as a string
6. All data converted into a vector form.
7. Check the similarity for each word until all words won't be checked.
8. Now execute 5 similarity measure algorithms.
9. Finally we get 5 accuracies for File2(comparing to file1).

Above way we can also find out or observe the similarity between File1 and File3.

4.2.3 Output

After executing algorithm we found this accuracy chart. This chart gives us the clear and specific idea of similarity measurement.

Table 4.1: Accuracy table for similar type of document

File 2=> Sample Document1	
Name of the Algorithm	Accuracy (%)
Euclidean Similarity	94.77%
Manhattan Similarity	54.44%
Minkowski Similarity	60.21%
Cosine Similarity	27.54%
Jaccard Similarity	50.00%

As we can see Standard document and Sample document1 are quite similar. So the resultant accuracies are high. The accuracy is very high as the distance is very small. From the Table 4.1 we can see the accuracies of all algorithms. The highest accuracy is 94.77% and lowest accuracy is 27.54%. We can use any of the any of the similarity algorithm for our desired purpose.

Table 4.2: Accuracy table for dissimilar type of document

File 3=> Sample Document2	
Name of the Algorithm	Accuracy (%)
Euclidean Similarity	34.36%
Manhattan Similarity	14.40%
Minkowski Similarity	21.6%
Cosine Similarity	8.50%
Jaccard Similarity	2.00%

As we can see Standard document and Sample document2 are much different. So there resultant accuracies are very low. The distance is large then the similarity is very low. From the Table 4.2 we can see the accuracies of all algorithms. The highest accuracy is 34.36% and lowest accuracy is 2.00%. So we can see the actual difference between two documents. One of them is very similar to the standard document and another is quite dissimilar from standard document. It's because of the distance.

4.3 Graphical views

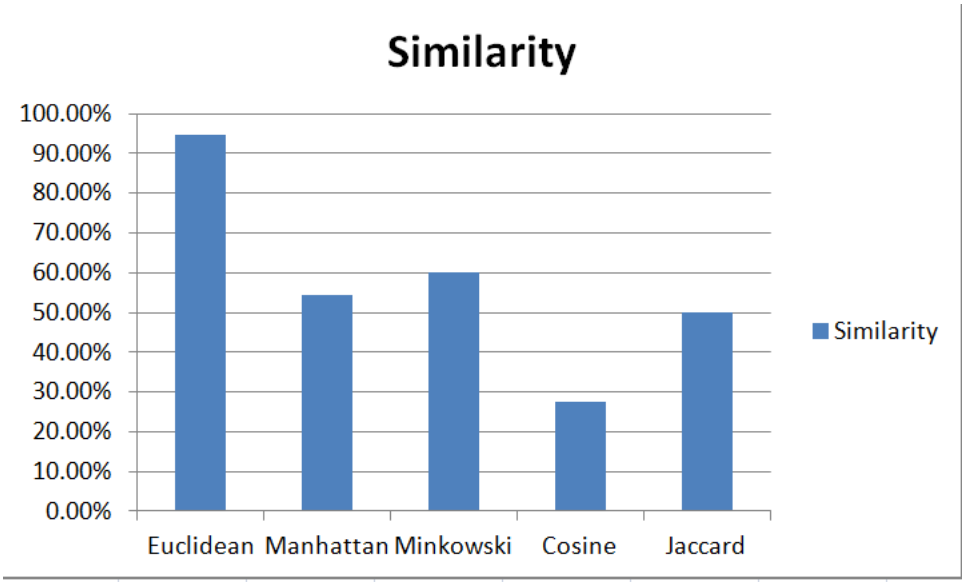

Fig 4.8: Bar chart for Sample Document1. X axis presents name of the algorithm and Y axis presents similarity.

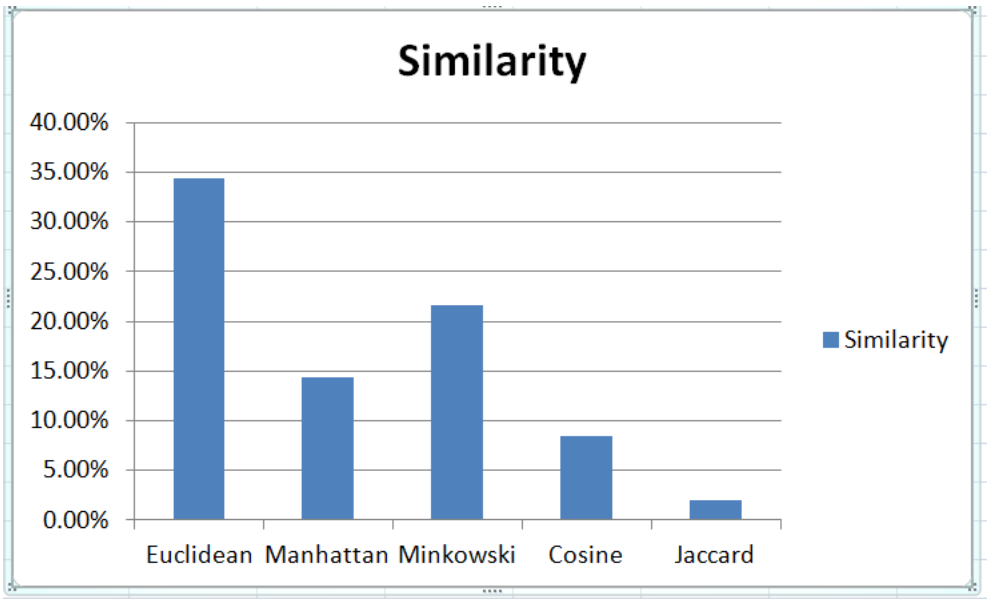

Fig 4.9: Bar chart for Sample Document2. X axis presents name of the algorithm and Y axis presents similarity.

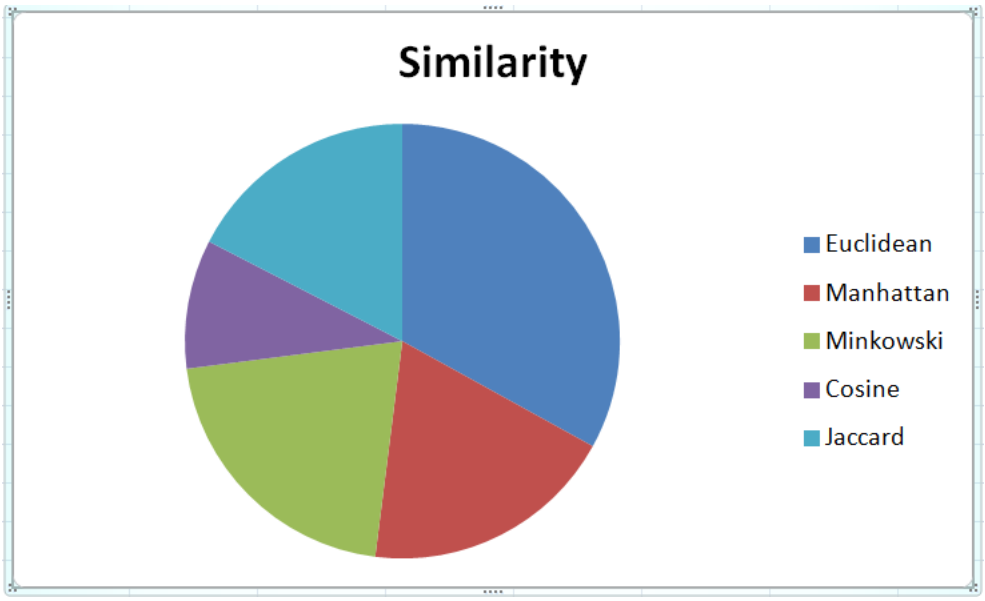

Fig 4.10: Pie chart for Sample Document1

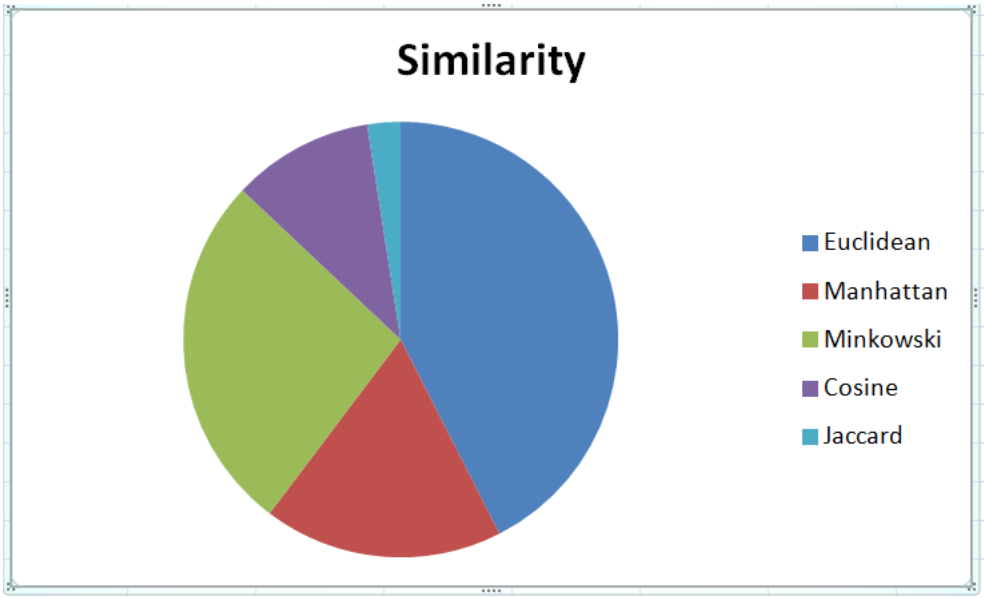

Fig 4.11: Pie chart for Sample Document2

Chapter 5

Conclusion and Future Development

5.1 Conclusion

The concept of similarity measurement is very important and vital concern in this new era. It is also widely used concept. We have tried to introduce a way to learn sentence representations for semantic textual similarity. This article contains mainly five algorithms that we have implemented for similarity measurement using two conversion methods (TF-IDF and Word2Vector). We always provide a standard document that can be as trained document. And also provide a/some sample documents that can be used as text document. After executing our five most popular similarity measure algorithms we will get accuracy easily. These five algorithms basically provide five types of accuracy. We think it will be very useful in the field if computer science. Therefore, future efforts are very essential to improve the ability for increasing the accuracy using Machine Learning tools and methods.

5.2 Future Development

As we all know similarity measurement is a very vast area in the field of computer science especially in Artificial Intelligence and Machine learning. So it's our desire to implement many more concerns with this similarity measurement.

5.2.1 Our further research

Following concerns could be suitable for future research:

1. Firstly, we will implement other conversion method (text to vector) to observe the accuracy whether it increases or not.
2. Secondly, we will merge the result of similarity measure algorithms to obtain a maximum accuracy.

3. Throughout the thesis we have worked with English context only. As Bengali is our mother tongue we want to implement those algorithms for Bengali. We think we will very essential for our country.

References

[1] Lin, D.: Information Theoretic definition of similarity. In: Proc. 15th International Conf. on Machine Learning (1998)

[2] Fellbaum, C.: WordNet, An Electronic Lexical Database. MIT press, Cambridge (1999) [3] Francis, Kucera: Computational Analysis of present day American English. Brown University press (1967)

[4] Resnik, P.: Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. Journal of Artificial Intelligence Research (JAIR) 11, 95–130 (1999)

[5] Berry, M.W., Dumais, S.T., O'Brien, G.W.: Using Linear Algebra for Intelligent Information Retrieval. SIAM Review 37(4) (1995)

[6] Jiang, Conrath: Semantic similarity based on Corpus statistics and lexical Taxonomy. In: Proceedings of International Conference Research on Computational Linguistics (1997)

[7] Scott, S., Matwin, S.: Text classification using WordNet hypernyms. In: Proc. of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems (1998)

[8] Scott, S., Matwin, S.: WordNet improves text document clustering. In: Proc. of the Semantic Web Workshop at SIGIR-2003 (2003)

[9] Rada, R., Milli, H., Bicknell, E., Blettner, M.: Development and Application of a Metric on Semantic Nets. IEEE Transactions on Systems, Man and Cybernetics 1(9), 17–30 (1989)

[10] Lesk, M.: Automatic sense disambiguation: How to tell a pine cone from an ice-cream cone. In: Proc. of the 1986 ACM SIGDOC conference, New York, pp. 24–26 (1986)

[11] C. M. Lyon, J. A. Malcolm, and R. G. Dickerson (2001). Detecting short passages of similar text in large document collections. In Proceedings of Conference on Empirical Methods in Natural Language Processing. SIGDAT Special Interest Group of the ACL.

[12] C. D. Manning and]H. Schutze (1999). Foundations of Statistical Natural Language Processing. Cambridge, MA: MIT Press.

[13] Ai.intelligentonlinetools.com/ml/text-vectors-word-embeddings-word2vec/

[14] Jesús Oliva et al., "SyMSS: A_Syntax-Based Measure for Short-Text Semantic Similarity", Data & Knowledge Engineering, Vol.70, pp.390–405, 2011.

[15] T. K. Landauer and S. T. Dumais, "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of the Acquisition, Induction, and Representation of Knowledge", Psychological Review, Vol. 104, Nos. 2, pp. 211-240, 1997.

[16] M. Lesk, "Automatic Sense Disambiguation using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone", Proc. of 5th Annual International conference on Systems documentation, pp.24-26, USA, 1986.

[17] Burgess, K. Livesay and K. Lund, "Explorations in Context Space: Words, Sentences, Discourse", Discourse Processes, Vol.25, No.2-3, pp. 211-257, 1998.

[18] A. Maguitman, F. Menczer, H. Roinestad, and A. Vespignani, "Algorithmic Detection of Semantic Similarity", Proc. of 14th International World Wide Web Conference, USA, 2005.

[19] Y. Li, D. McLean, Z. Bandar, J. O'Shea, and K. Crockett, "Sentence Similarity Based on Semantic Nets and Corpus Statistics," Knowledge and Data Eng., Vol.18, No.8, pp. 1138-1149, Aug. 2006.

[20] Mehran Sahami and Timothy D. Heilman, "A Web-based Kernel Function for Measuring the Similarity of Short Text Snippets", Proc. of International World Wide Web Conference, Scotland, 2006.

[21] Sanghamitra Bandyopadhay and Koushik Mallick, "A New Path Based Hybrid Measure for Gene Ontology Similarity", Computational Biology and Bioinformatics, Vol.11, No.1, pp 116-127, 2014.

[22] H. Chen, M Lin, and Y. Wei, "Novel Association Measures Using Web Search With Double Checking", Proc. of 21st International Conference on Computational Linguistics, pp1009-1016, Sydney, 2006.

[23] Yuncheng Jiang et al., “Feature-based approaches to Semantic Similarity Assessment of Concepts using Wikipedia”, Information Processing and Management, Vol. 51, pp. 215–234, 2015.

[24] Hung Chim and Xiaotie Deng, “Efficient Phrase-Based Document Similarity for Clustering”, Knowledge and Data Engineering, Vol.20, No.9, pp.1217-1229, 2008.

[25] Francine Chen, Ayman Farahat and Thorsten Brants, “Multiple Similarity Measures and Source-Pair Information in Story Link Detection”, Proc. of Human Language Technology Conference, pp.313-320, Chicago, 2004.

[26] Mohamed Ali Hadj Taieb et al, “Ontology-Based Approach for Measuring Semantic Similarity”, Engineering Applications of Artificial Intelligence, Vol. 36, pp. 238–261, 2014.

[27] Wael H. Gomaa and Aly A. Fahmy, “A Survey of Text Similarity Approaches”, International Journal of Computer Applications, Vol.68, No.13, pp.0975 – 8887, 2013

[28] Islam A and Ink pen, “Second order co-occurrence PMI for determining the semantic similarity of words”, Proc. of International Conference on Language Resources and Evaluation, pp.1033-1038, Genoa, Italy, 2006.

[29] Peter D. Turney, “Measuring Semantic Similarity by Latent Relational Analysis”, Proc. of 19th International Joint Conference on Artificial Intelligence, pp. 1136-1141, Canada, 2005

[30] Anna Huang, “Similarity Measures for Text Document Clustering”, Computer Science Research Student Conference, pp.49-56, New Zealand, 2008.

[31] Paul Vitanyi, “Universal Similarity”, Proc. of International Theory Workshop, New Zealand, 2005.

[32] Rudi L. Cilibrasi and Paul M.B. Vitanyi, “The Google Similarity Distance”, Knowledge and Data Engineering, Vol.19, No.3, pp.370-383, 2007.

[33] Danushka Bollegala et al., “Measuring Semantic Similarity between Words Using Web Search Engines”, Proc. of 16th International Conference on World Wide Web, pp. 757-766, Canada, 2007.

[34] Lixin Han, Linping Suna, Guihai Chenb and Li Xieb, “ADSS: An Approach to Determining Semantic Similarity”, Advances in Engineering Software, Vol.37, pp.129–132, 2006.

[35] Peipei Xia, Li Zhang and Fanzhang Li, “Learning Similarity with Cosine Similarity Ensemble”, Information Sciences, Vol.307, pp.39–52, 2015.

[36] Danushka Bollegala,Yutaka Matsuo and Mitsuru Ishizuka , “A Web Search EngineBased Approach to Measure Semantic Similarity between Words”, Knowledge and Data Engineering,Vol. 23, No.7, pp.977-990, 2011.