



**TEXT CATEGORIZATION BASED ON SEMANTIC SIMILARITY WITH
WORD2VECTOR**

ATHER ABDULRAHEM MOHAMMEDSAED

ALSAMURAI

SEPTEMBER 2017

TEXT CATEGORIZATION BASED ON SEMANTIC SIMILARITY WITH
WORD2VECTOR

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ÇANKAYA UNIVERSITY

BY

ATHER ABDULRAHEM MOHAMMEDSAED
ALSAMURAI


IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF COMPUTER ENGINEERING
INFORMATION TECHNOLOGY PROGRAM

SEPTEMBER 2017


Title of Thesis: Text Categorization Based on Semantic Similarity with Word2Vector

Submitted by Ather Abdulrahem Mohammedsaed Alsamurai


Approval of the Graduate School of Natural and Applied Sciences, Çankaya University


Prof. Dr. Can ÇOGUN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.


Prof. Dr. Erdoğan DOĞDU
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.


Assist. Prof. Dr. Abdül Kadir GÖRÜR
Supervisor

Examination Date: 15.09.2017

Examining Committee Members

Assist. Prof. Dr. Abdül Kadir GÖRÜR (Çankaya Univ.)

Assist. Prof. Dr. Özgür Tolga PUSATLI (Çankaya Univ.)

Assist. Prof. Dr. Hassan SHARABATY (THK Univ.)





STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname : Ather Alsamurai

Signature : 

Date : 15.09.2017



ABSTRACT

**TEXT CATEGORIZATION BASED ON SEMANTIC SIMILARITY WITH
WORD2VECTOR**

ALSAMURAI, Ather

M.S., Information Technology Department

Supervisor: Assist. Prof. Dr. Abdül Kadir GÖRÜR

September 2017, 58 pages

With an increase in online information, which is mostly in the form of a text document, there was a need to organize it so that management and retrieval by the search engine became easier. It is difficult to manually organize these documents, therefore, machine-learning algorithms can be used to classify and organize them. Mostly, they are faster,

more accurate and less expensive than manual classification. Most traditional approaches of machine learning algorithms depend on the term frequency in determining the importance of the term within a document and neglect semantically similar words. For this reason, we proposed to build a classifier based on semantically similar words in text classification by using the Word2Vector model as a tool to compute the similarity between documents and capture the correct topic. So we built two models by applying three phases: the first phase, we applied preprocessing steps and the second phase, we created a dictionary for top ten categories of Reuters 21578 datasets and the final phase we trained Word2Vector model on the Wikipedia English dataset and use it to compute similarity

between documents. Depending on the results of our study, we found that the second model (the most similar predicted topic) is better than the first model (average based predicted topic) in all categories. When we compare the results of our study with other studies, we found that result of our study is a parallel to the results of other studies, but not overcome them, although these studies use feature selection in the improvement of their results while we use feature extraction in explaining of our results.

Keywords: Text Categorization, Semantic Similarity, Word2Vector Model



ÖZ

WORD2VECTOR İLE SEMANTİK BENZERLİĞE DAYANAN METİN KATEGORİZASYONU

ALSAMURAI, Ather

Yüksek Lisans, Bilgi Teknolojileri Anabilim Dalı

Tez Yöneticisi: Yrd. Doç. Dr. Abdül Kadir GÖRÜR

Eylül 2017, 58 sayfa

Çoğunlukla bir metin belgesi biçiminde olan çevrimiçi bilginin artmasıyla birlikte, belge erişimi ve yönetimi kolay hale gelmesi için bir organizasyona ihtiyaç duyulmaktadır. Bu belgelerin el ile organize edilmesi zordur, bu nedenle makine öğrenme algoritmaları, belgeleri sınıflandırmak ve organize etmek için kullanılabilir. Çoğunlukla, manuel sınıflandırmadan daha hızlı, daha doğru ve daha az maliyetlidir. Makine öğrenme algoritmalarının geleneksel yaklaşımlarının çoğu, terimlerin bir belgedeki önemini belirlerken kullanılan terim sıklığına ve anlamsal olarak benzer kelimeleri ihmal etmesine bağlıdır. Bu nedenle, belgeler arasındaki benzerliği hesaplamak ve doğru konuyu yakalamak için bir araç olarak Word2Vector modelini kullanarak, metin sınıflandırmasında anlambilimsel olarak benzer kelimelere dayalı bir sınıflandırıcı oluşturmayı önerdik. Bu nedenle, üç aşamalı yaklaşım uygulayarak iki model oluşturduk: Birinci aşama, ön işleme adımlarını uyguladık ve ikinci aşama, Reuter 21578 derleminin ilk on kategorisi için bir sözlük hazırladık ve Wikipedia İngilizce veri setinde Word2Vector modelini eğittiğimiz son aşama ile sınıflayıcı oluşturarak belgeler

arasındaki benzerliđi hesapladık. alıřmamızın sonularına bađlı olarak, ikinci modeli (en benzer belgenin categorilerini tahmin edilen kategoriler olark belirledik), ikincisinde ise tm kategorilerdeki ilk modelden (ortalama bazlı) daha iyi bulduk. alıřmamızın sonularını diđer alıřmalarla karřılařtırdığımızda, sonularımızın diđer alıřmaların sonularına paralel olduđunu ancak kategorilerin bazılarında iyi sonular alırken bazılarında daha kt sonular alındığını tespit ettik.

Anahtar Kelimeler: Metin Sınıflandırma, Anlamsal Benzerlik, Word2Vector Model

ACKNOWLEDGEMENTS

I would firstly like to thank my thesis advisor, Dr. Abdül Kadir GÖRÜR of the Computer Engineering Department at Çankaya University, without whose helpful advice, valuable comments and guidance this thesis could not be completed. His door was always open for me whenever I needed his help. His immense knowledge was an asset that was always readily available for me to explore, which turned out to be the best source of information that was available to me.

Special thanks to my dissertation committee members, Dr. Abdül Kadir GÖRÜR, Dr. Özgür Tolga PUSATLI, Dr. Hassan SHARABATY. I also thank all my colleagues and friends for their support of my studies.

Most of all, I am grateful for the constant support from my parents, my wife, my brothers, and the rest of my family, without whom none of this work would have been possible.

TABLE OF CONTENTS

STATEMENT OF NON PLAGIARISM.....	iii
ABSTRACT.....	iv
ÖZ.....	vi
ACKNOWLEDGEMENTS.....	viii
TABLE OF CONTENTS.....	ix
LIST OF FIGURES.....	xi
LIST OF TABLE.....	xii
LIST OF ABBREVIATIONS.....	xiii

CHAPTERS:

1. INTRODUCTION.....	1
1.1 Aim of the study.....	2
1.2 Significant of the study.....	2
1.3 Related works.....	3
2. BACKGROUND AND TEXT CATEGORIZATION.....	5
2.1 Text Categorization.....	5
2.2 Semantic meaning.....	8
2.3 Word2Vector Model.....	9
2.3.1 Continuous Bag of Worde (CBOW)	10

2.3.2 Continuous Skip-Gram Model.....	13
3. METHODOLOGY	16
3.1 Buildings of Classifier.....	16
3.2 Reuters21578dataset.....	20
3.3 Preprocessing phase (Noise Removal)	22
3.3.1 Parsing the Documents and Case-Folding.....	23
3.3.2 Removing Stop words.....	23
3.3.3 Tokenization.....	23
3.3.3.1 Difference between token and type.....	24
3.2.3.2 Penn Treebank. Tokenize.....	24
3.3.4 Stemming.....	25
3.4 Vector Space Model (Document Representation).....	26
3.4.1 Term Frequency (TF)	27
3.4.2 TF-IDF weighting.....	28
4. RESULTS	30
4.1 Performance Measurement Metrics.....	30
4.2 Results for the Reuters-21578 Data Set.....	32
4.2.1 Method 1(Average based predicted topic.....	32
4.2.2 Method 2 (The Most Similar Predicted Topics)	34
5. CONCLUSION	38
REFERENCES.....	40
APPENDIX.....	43
CURRICULUM VITAE.....	43

LIST OF FIGURES

FIGURES

Figure 1.	A linear vs non-linear problem space.....	7
Figure 2.	CBOW and Skip-Gram model architecture.....	10
Figure 3.	CBOW model architecture for context consist of a single word.....	12
Figure 4.	Input layer and target for context of a single word.....	13
Figure 5.	Architecture skip-gram model.....	15
Figure 6.	Simple form taken from cats.txt file of Reuters21578.....	17
Figure 7.	Subdirectories of Wikipedia English after preprocessing.....	18
Figure 8.	Schematic of Model 1(average based predicted topic)	19
Figure 9.	Schematic of Model 2 (the most similar predicted topic)	20
Figure 10.	Stages of the preprocessing.....	22
Figure 11.	List of stop words that used widely.....	23
Figure 12.	Output tokenization by tokenizer.....	24
Figure 13.	Precision, Recall, and F1 measure for Model 1.....	33
Figure 14.	Precision and recall, F1 measure for Model 2.....	35

LIST OF TABLES

TABLES

Table 1:	Representation corpus into a training set a CBOW model with a context window of 1.	11
Table 2:	Matrix for context single word	11
Table 3:	Architected training data of skip-gram model.	14
Table 4:	Documents training and test for Reuters 21578	21
Table 5:	Stemming Rules	25
Table 6:	Term frequency matrix of various of term in two	27
Table 7:	A Sample group Representing Term Frequency of Different Terms in a Set of Documents.	28
Table 8	Documents training and test for Reuters 21578 of model 1	32
Table 9:	Precision, recall and F1 score for model 1	33
Table 10	Documents training and test for Reuters 21578 of model 2	34
Table11:	Precision, recall and F1 score for model 2	35
Table 12:	Comparison measurements between two methods.	36
Table 13:	Model 2, F1 values comparison with kNN, SVM, and DAN2	37

LIST OF ABBREVIATIONS

FN	False negative
TC	Text Classification
TF-IDF	Term Frequency- Inverse Document Frequency
TN	True negative
TP	True positive
VSM	Vector Space Model
W2V	Word2Vector



CHAPTER I

INTRODUCTION

In recent years, text classification has received importance because of the great growth of the text in the digital form of various sources across the World Wide Web (www), also most persons, corporation, and Institutions have their own site on the web. When we want to look for some information, we often search for it by search engines because most of the information has become publicly available. Also, there is always constantly increasing information, this huge information require to be classified in order to get a benefit of it. If the information size is small or written by us, it is classified easily, but because of the rapid growth information on the online, it is impossible to classify these documents manually and it is necessary to classify these documents automatically into proper categories by their content.

Most resources on the internet involve semi-structured and unstructured information such as ‘news article’, ‘online forums’, ‘digital libraries’, ‘e-mail’, ‘blog repositories’ and etc [1], in order to access information required easily from this huge amount of information we need the correct classification for these resources. For this reason, the text classification field has earned great importance in the research community. In order to extract information from various text resources on online and deal with it through the processes of classification and retrieval and summarization, it is required to be properly explained and represented and categorized to achieve proper generalization and prevent over-fitting [2].

Human have the capability to understand unstructured data and to overcome problems that computers cannot deal with them, but humans do not have the computer's ability to handle huge amount of documents at the same high speed [2]. Therefore, in our study, we make computers mimic human behavior by measuring the similarity between words in text categorization task through comparing the new document with a document under a certain category belongs to the training set and assigning labels for for the new document if the value of measured similarity is great.

1.1 Aim of the study

The machine learning algorithms are based on a word as a basic structure for text classification. Statistical analysis of words by using a term frequency determines the importance of a word in a document only. This is considered as a traditional classification which does not care about the semantic meaning of the word. So, the main aims of this study are building a classification model which depends on semantically similar documents and evaluating the accuracy of classifier performance by using performance measurement metrics.

1.2 Significance of the study

Mostly, a human can handle textual document easily, but when we have millions of documents that were generated in a day, thus making the task difficult or impossible for us. So that, we need to categorize text automatically, this task needs software tools that can classify documents within predefined classes. Automated text categorization has a significant importance because of rapid growth of textual documents in the online. It was necessary to categorize them so that organization and retrieval by search engine became simple.

Automated text classification has importance in many fields such as in the business world field, automated classification of users can make marketer's life simple. Marketers can monitor and classify users based on how they talk about a product or brand online.

1.3 Related Works

Many research groups are working on automated text categorization and provide an efficient algorithm due to the importance of the research topics. The work proposed by L. Bing, L. Xiaoli, Wee Sun Lee, and S. Philip [3] propose a method within semi supervised learning depends on an extracting group of words for every class. That used to extract set of documents which are unlabeled of each class for constructing an initial training set. For extracting representative words, they proposed integrated feature selection with clustering and then ask the user about important words. The results of the approach have shown its effectiveness.

Similar work proposed by B. Tang, H. He, M. Baggenstoss, and K. Kay [4] and they use approach to choose a feature subset for every class. The researchers derived a new naive Bayes rule following Baggenstoss's PDF Projection Theorem. The benefit of such way is to combine many feature selection standards, such as Information Gain (IG) and Maximum Discrimination (MD). The results improve accuracy in relation to update ways.

The work done by B. Tang, H. He, and K. Kay [5] proposed new measure used for score performance of multi class classification, this measure called Jeffreys-Multi-Hypothesis. They used Jeffreys-Multi-Hypothesis metric for improving two methods to choose feature selection, termed maximum discrimination and methods in order to classify text, the results of many experiments illustrated the efficacy of this approach.

A similar study was done by Xiao-Bing Xue and Zhi-Hua Zhou [6] which proposed a new type of value that is called feature distribution represent a distribution of a word in the text. Distribution feature involves a minimum number of a word and the first presence

of the word. Distributed feature was used by tf-idf model and various features and integrated with ensemble learning techniques. The experiment shows that distributional features have an importance when the text are extended and the writing style is easy.

Y. Ping, Y. Zhou, Y. Yang, and P. Peng [7] proposed work for addressing many common keyword between the classes of a dataset, and which effecting classification task. The researchers proposed a new naming scheme called prob_rf that used to correct conventional naïve Bayes Vectorizer of the hybrid categorization method to get a good description of the information, it joined term weighting scheme with the distribution factor to get a good accuracy.

CHAPTER II

BACKGROUND AND TEXT CATEGORIZATION

2.1 Text Categorization

Text classification is a task allocating pre-defined labels of category to new documents that depend on the probabilities associated with the training examples. The important part of text mining is the text classification. In order to automate Text classification we need a training data set for building models. The training set requires manual systems that come with the assistance of knowledge-engineering techniques, which is based on expert knowledge to manually write a group of rules that are utilized for categorizing documents within pre-defined categories. A machine learning approach is used to classify documents automatically that work to save time, effort and cost by reducing the number of experts with maintaining the accuracy of classification. For example; let us suppose that a set of logical rules has been manually labeled which assigns labels to documents, that machine learning algorithm need them for the task of classification. Therefore, the model was built within a supervised approach that needs two collections of data: training set and test set, so the training set is $D = (d_1, d_2, \dots, d_n)$, which they are documented back to predefined labels within classes C_1, C_2 (e.g. Sciences, sports, economics, ..., c_n). The test set is a collection of documents have not been labeled before and used for verifying from the performance of the model. A machine learning algorithm aimed to build a model that is capable to select the right class (d) from a set of classes (c), depending on the features of the training set, for assigning one label or more. The single-label means a text that is beneath single class, for instance; e-mail

messages which are classified to spam or not spam, this classification is called single-labeled classification, where every text must be allocated to exactly single class [7]. Multi-label means a document that is beneath more than one class, for example; a part of news related to in what way the Prime Minister spent the holiday, this piece of news can be putted in the social column or politics column, so this classification is called multi-labeled classification.

The main problem in Machine learning algorithms is how they can understand the data which is mostly in the text configuration and this data has been written in a natural language, how they picked the particular features which are associated with the classification task.

There are three essential kinds of machine learning algorithms for dealing with data sets that have been described in natural language and process them:

- **Supervised learning**

Supervised learning are machine-learning techniques that are related to the inferring function or learning a classifier from the training dataset in order to find the correct label for a novel document. There is a broad range of supervised methods such as nearest neighbor classifiers and probabilistic classifiers and act [8].

- **Unsupervised learning**

Unsupervised learning are techniques that try to find hidden structure out of unlabeled data. They do not need the training phase, therefore, could be implemented on every text data with no exertion.

Clustering and topic modeling are the two widely used unsupervised learning algorithms in the context of text data. Clustering aims to divide a collection of documents into partitions where documents, in the similar group are identical to each other than those in other groups. In topic modeling a probabilistic model is

applied to determine a soft clustering, in which each document has a probability distribution across all the clusters opposed to static clustering [8].

- **Semi-supervised learning**

This combines supervised and unsupervised learning which produces a way that make use of unlabeled data for training-typically with a small amount of labeled data and a large amount of unlabeled data.

Machine learning algorithms are considered as supervised learning when assigning documents to one or more classes based on inferring of classification function, in which these methods of learning based on experts that describes the collection of categories and labels in the training set and considered as an instructor for the learning algorithms [9].

$$\Gamma(D) = \gamma$$

Γ refers to supervised learning.

D refers to the training set.

γ refers to learned classification function.

In fact, there are two kinds of classifiers: linear classifiers and non linear classifiers, as shown in the figure below.

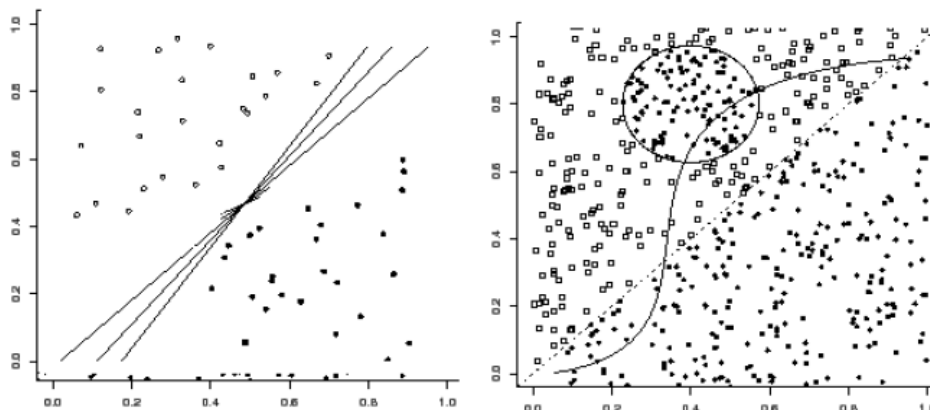


Figure 1. A linear vs non-linear problem space.

The linear classifier is a two-class classifier, which can determine a boundary of classes by relating a linear collection of the features with a threshold. Linear classifier gives us a good result when a problem is linear, while if we cannot determine the borders of category to linear hyperplanes, this case is nonlinear and the results of it will be better than Linear Classifiers [9].

2.2 Semantic Meaning

Text classification is considered as a task to classify documents automatically under the predetermined one category or more depend on their contents. Many text categorization algorithms that based on the semantic meaning measure similarity between documents. Therefore, semantic meaning has a significant role in document categorization. The text of documents involves semantic information, for this reason, machine-learning algorithms can employ semantic information to obtain good results. In the area of text categorization, the basic structure in representation documents are terms and their frequencies. This feature of representation is called Bag of Words and it is widely used. In this approach, every word creates its own dimension in a vector space, apart from other words surrounding it in the same text [10]. BOW is considered as a way that is easy and widely utilized and it has many limitations. So that it is regarded as words that independent of one to another. Words are represented in the vector space model with ignoring their location in the document and also ignoring their semantic meaning associated with other words. Therefore, it is neglecting the multiple meanings of the word, for example: the word “apple” may refer to fruit when it comes in a context related to eatables, or it refers to a company when it comes in a context that refers to technology. In principle, of analyzing and argue [11], every category has two kinds of vocabulary: one is “core” vocabulary which is closely related to the topic of that category, the other kind is “general” vocabulary, and this can have analogous distributions on diverse categories. So, there may be 2 documents from different categories which can participate in several general words and may be reflect the similarity in the BOW approach.

In order to address these problems several methods have been proposed which use a measure of relatedness between term on Word Sense Disambiguation Many approaches have been suggested for dealing with these challenges that they use relationship between words on explain word meaning. So the distance between words in WordNet as mentioned in [12] is used to capture semantic similarity between English words. The semantic relatedness computations among the words are corpus-based systems that certain statistical analysis depending on the associations between words in group of training documents is conducted in order to expose latent similarities among them. In this in text categorization, semantic relatedness can be categorized into three approaches: knowledge base systems, statistical approaches and hybrid ways which integrate both ontology-based and statistical information study, we propose a method for building a classifier based on semantic meaning by using the Word2Vector model as a tool for measuring similarity between documents.

2.3 Word2Vector Model

Word2Vector is a tool that related to deep learning and it is produced by Google [13], Word2Vector turns a text into numerical form and is called vector, the purpose of that is the deep neural net can understand it. It is a two-layer neural net, its input is a sequence of words and its output is a collection of vectors. These vectors can be used as a feature in the classification problems.

Word2Vector consist of 2 famous models, Continuous Bag Of Words (CBOW) and Continuous Skip-Gram Model (Mikolov et al.). Continuous Bag Of Words (CBOW) predicts target word from the context of the text, while Continuous Skip-Gram model predicts context of text from a certain word. In the below figure 2 explain the architecture of the two models [13].

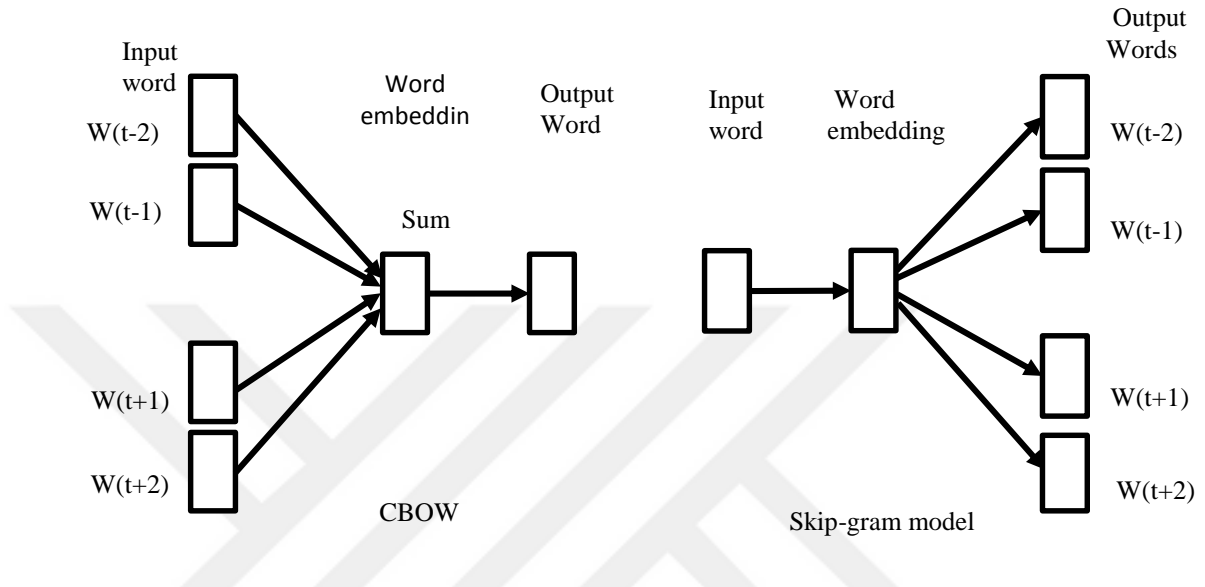


Figure 2. CBOW and Skip-Gram model architecture.

2.3.1 Continuous Bag Of Word (CBOW)

The CBOW model depends on predicting the word from a given context. A context is a set of words or maybe one word. Neural network architecture is adjusted and this adjustment made the input to a hidden layer is repeated according to C times, for determining C from the number of context words, every word is coded using 1-out-of- V , this means that the weighting between hidden layer and output layer is the word vectors that correspond to a word in the context as input to hidden layer. So that CBOW model consist of a single hidden layer and completely connected to neural network [14]. The hidden layer has neurons in the linear form. Hidden layer size represents the number of neuron network. The input layer size is the number of words in the vocabulary that utilized

for training and output layer is the target word. For example for simplicity, we will use a single context word for predicting a single target word. Let us suggest that we have corpus C=“Hey, this is a sample corpus using only one context word.” and we determine a context window of 1. This corpus can be translated into a training set for a CBOW model as follows:

Table 1: Representation corpus into a training set a CBOW model with a context window of 1.

Input	Output		Hey	Thi s	is	Sample	Corpu s	using	Only	One	context	Word
Hey	This	Datapoint1	1									
This	Hey	Datapoint2		1								
This	Is	Datapoint3		1								
Is	This	Datapoint4			1							
Is	Sample	Datapoint5			1							
Sample	Is	Datapoint6				1						
Sample	Corpus	Datapoint7				1						
Corpus	Sample	Datapoint8					1					
Corpus	Using	Datapoint9					1					
Using	Corpus	Datapoint10						1				
Using	Only	Datapoint11						1				
Only	Using	Datapoint12							1			
Only	One	Datapoint13							1			
One	Only	Datapoint14								1		
One	Context	Datapoint16								1		
Context	One	Datapoint17									1	
Context	Word	Datapoint18									1	
Word	Context	Datapoint19										1

In the figure below we show a single data point.

Table 2: Matrix for context single word.

Hey	This	Is	Sample	Corpus	using	only	one	Word
0	0	0	1	0	0	0	0	0

As explained in the above figure the matrix is passed into a shallow neural network composed of 3 layers: an input layer, a hidden layer and an output layer. Softmax layer exists in the output layer that is used to sum the probability appeared in the output layer to 1. We see how the forward propagation will be used to compute the hidden layer activation [14].

The figure below show a diagrammatic representation of CBOW model for context involved from a single word.

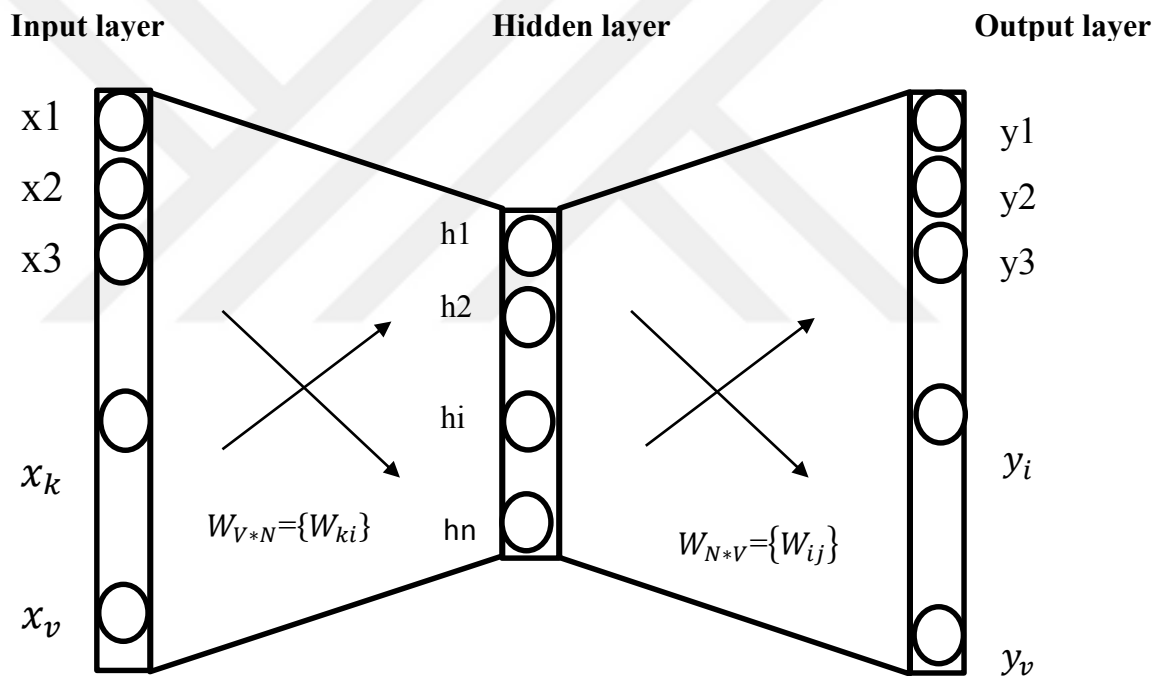


Figure 3. CBOW model architecture for context consist of a single word.

There are collections of weights between input and hidden layer, this set is represented as a matrix which its size = $[V * N]$, and another matrix between hidden layer and output layer Hidden activation that its size = $[N * V]$, the figure below represent both matrix of weights for input – hidden layer and hidden – output layer.

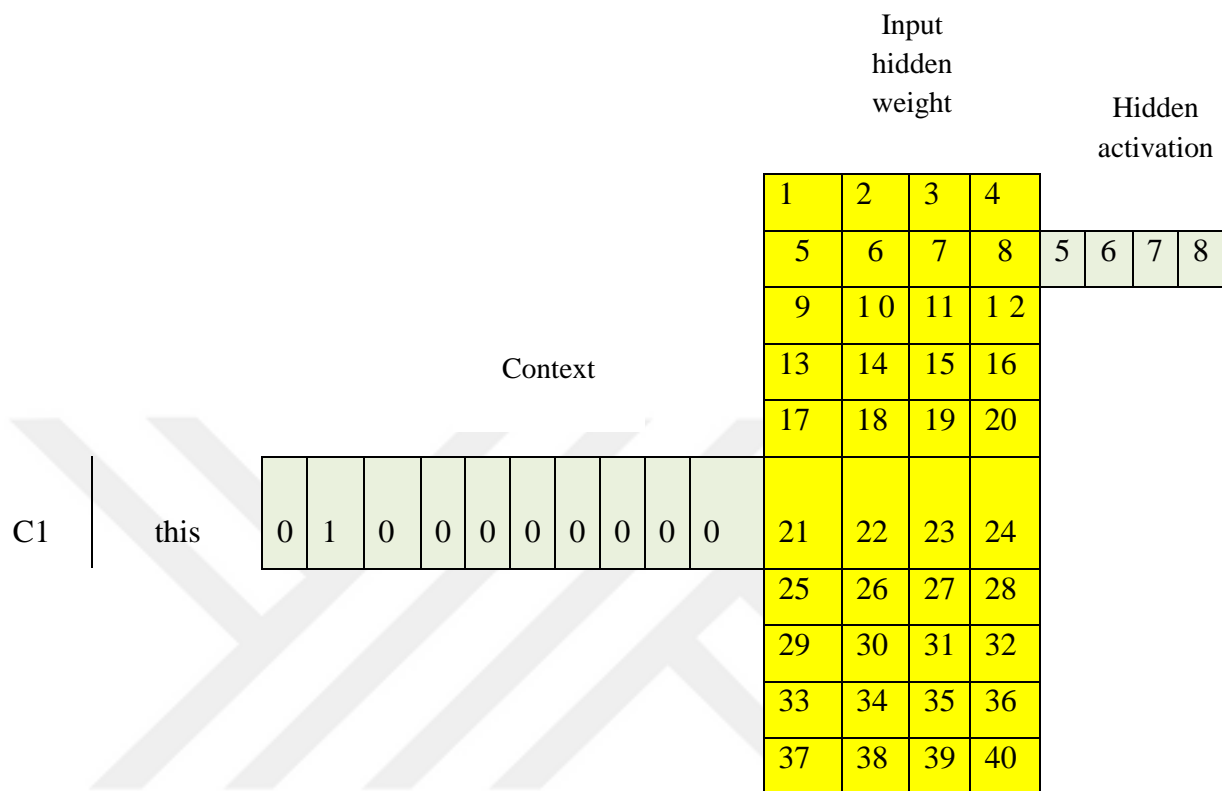


Figure 4. Input layer and target for context of a single word.

For a simplicity, activation function is copying the corresponding row in the input-hidden weight for 1-out-of-V, and hidden-output is multiplied by weight. The word vector indicates of the word which is utilized by weight between the hidden and output layer.

2.3.2 Continuous Skip-Gram Model

Continuous Skip-gram model is the same CBOW model, but it just reverses architecture on its head. The aim of continuous skips-gram model is to use target word for feeding input layer of the skip-gram model for predicting context. So we will use the same corpus

that we used it in CBOW. C="Hey, this is a sample corpus using only one context word."
 As in the table below.

Table 3: Architected training data of skip-gram model.

Input	Output(Context)	Output(Context2)
Hey	This	<padding>
this	Hey	Is
is	This	sample
sample	Is	corpus
corpus	Sample	corpus
using	Corpus	Only
only	Using	One
one	Only	context
context	One	word
word	Context	<padding>

The input of skip-gram will be the same to a 1-context CBOW model. Also, the computations of hidden layer activations will be similar. The variation is going to be in the target variable. Because we have determined a context window of 1 on both the sides, we will have 'two' one-hot encoded target variables. 'two' outputs. The word vector presentation of the word is used as a weight among the input and the hidden layer.

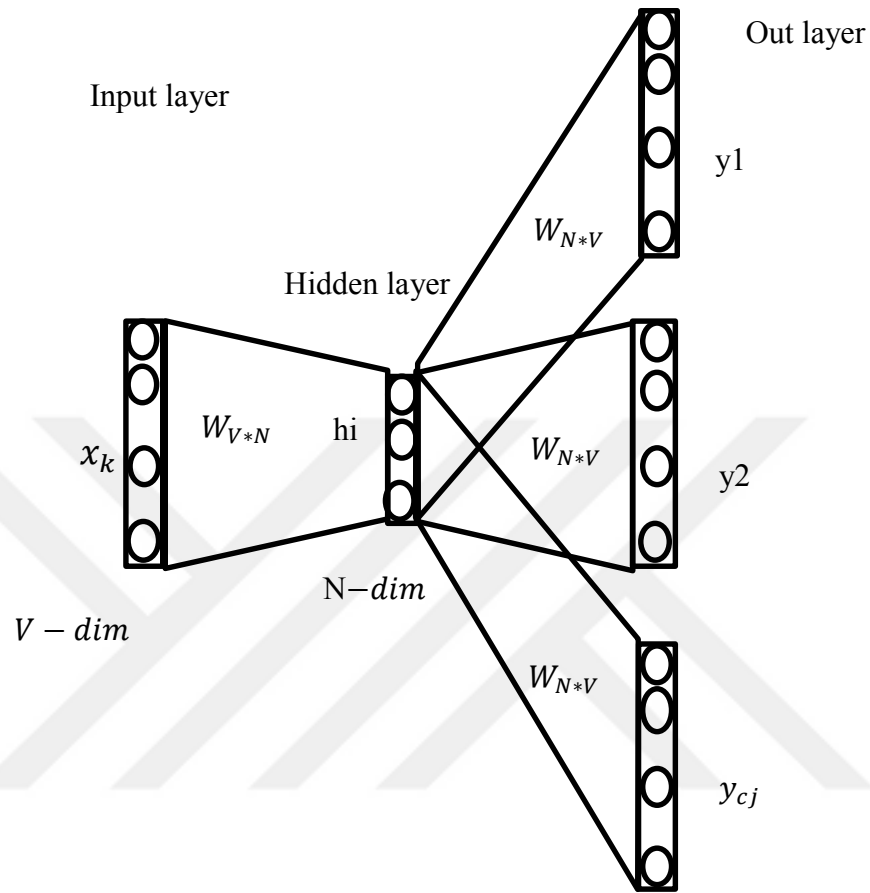


Figure 5. Architecture skip-gram model

The Skip-gram model predicts the context of words by using a specific word that provides the input layer, the hidden layer stays unchanged. We use every word present in an input layer to a log-linear classifier together with a continuous projection layer to produce words in a particular scope in front and behind the current word. So we accommodate the chosen number of context words that are produced through repetition, multiple times for the output layer of the neural network [14].

CHAPTER III

METHODOLOGY

3.1 Buildings of Classifier

Here, we present an independent overview in our framework that related to text classification through building model by using Word2Vector model, we used a python interpreter as the environment and used Reuters21578 as a sample study.

We download Reuters21578 from the natural language toolkit (NLTK) that its size (42.9 MB) and composes of training folder, test folder, text file (category) and text file (stopwords). NLTK is a platform utilized for constructing code for text analysis.

For building classifier we started with a training dataset that composes from two things topics and contains, so the topics are Id numbers of documents and the cantons are the texts. We removed the noise from all documents of the training set through using preprocessing stages. We convert all characters of words from the upper into lower form and splitting all texts into tokens by using tokenizer algorithm, and removed all stop words from the training set by using text file (stop words) of the Reuters corpus to shrink size training set and reduce mislead of classifier, we applied stemmer by using a porter stemmer algorithm and save all documents in new text file. After finishing preprocessing stages of the training set, we extract a feature set that includes all unique words or tokens

from the training set and save them in another new text file and used it as a basis for a classification model.

Text file (category) of Reuters is regarded the important part in the experiment because it's through we can control on an entire corpus. It includes all (Id) document numbers of the entire corpus, topics or (classes) and type of documents (training or test) as shown in the figure below.

```
test/21562 trade
test/21565 sugar
test/21567 coffee
test/21568 crude
test/21570 rice grain
test/21571 acq
test/21573 yen dlr money-fx
test/21574 ship
test/21575 ipi
test/21576 gold
training/1 cocoa
training/5 sorghum oat barley corn wheat grain
training/6 wheat sorghum grain sunseed corn oilseed soybean sun-oil soy-oil lin-oil
training/9 earn
training/10 acq
training/11 earn
training/12 acq
training/13 earn
training/14 earn
training/18 earn
training/19 grain wheat
training/22 copper
training/23 earn
```

Figure 6. Simple form taken from cats.txt file of Reuters21578

We used it to extract all categories of training set, in which there are 90 unique categories, then we split training set according to categories by creating dictionary which their keys are categories and their values are set of documents Id that belong to the same category. After that, we have created another dictionary which their keys are Id documents and their values are categories.

We moved to test set, so we applied all preprocessing stages on it and select terms which are in the feature extraction only for representing documents, we also save them in a text file. We completed initializing both training and test data set of Reuters, and then moved to train the Word2Vector model that we will use it as the main tool for classifying texts.

We trained the Word2Vector model on the other data set that it is called Wikipedia English. Wikipedia English is a data set that its size (12GB), the recent version deposit in April, 2017 and downloaded from (<https://dumps.wikimedia.org/enwiki/latest/>), and we used Wiki Extractor for removing noise from the text of Wikipedia dataset which is written by python code, split and save it on sub directories, and we added training dataset for Reuters to the these subdirectories as shown in the figure below.

[textminer@textminer:/opt/wiki/data/enwiki\\$ ls](#)

```
AA  AH  AO  AV  BC  BJ  BQ  BX  CE  CL  CS  CZ  DG  DN  DU  EB  EI
AB  AI  AP  AW  BD  BK  BR  BY  CF  CM  CT  DA  DH  DO  DV  EC  EJ
AC  AJ  AQ  AX  BE  BL  BS  BZ  CG  CN  CU  DB  DI  DP  DW  ED  EK
AD  AK  QR  AY  BF  BM  BT  CA  CH  CO  CV  DC  DJ  DQ  DX  EE  EL
AE  AL  AS  AZ  BG  BN  BU  CB  CI  CP  CW  DD  DK  DR  DY  EF  EM
AF  AM  AT  BA  BH  BO  BV  CC  CG  CQ  CX  DE  DL  DS  DZ  EG  Reuters
AG  AN  AU  BB  BI  BP  BW  CD  CK  CR  CY  DF  DM  DT  EA  EH
```

Figure 7. Subdirectories of Wikipedia English after preprocessing.

We applied a python script “train_Word2Vectorc_with_gensim. Pay” for a training Wikipedia dataset, this process took around 150 hours. We modified this code and added remove stop words and porter stemmer algorithm in order to make its vocabulary similar

to the vocabulary of the training set of Reuters. We have two models for assigning a correct category to the unseen document.

Model 1 is a way that assigns a category to a document through calculating the average similarity between documents of test set and categories of training set, so we select a document from the test set and select category of training set, every category is considered as a clustering which involves a set of documents. We selected a category that it has the highest average similarity allocated to the document.

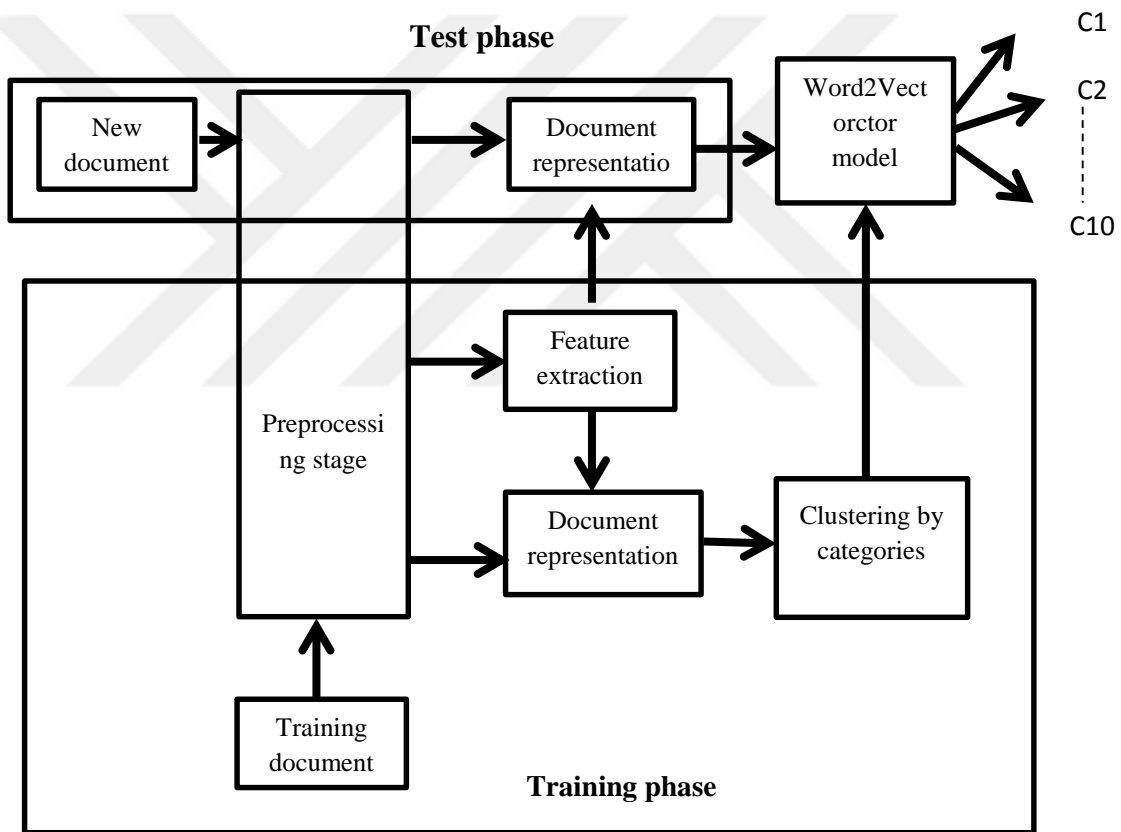


Figure 8. Schematic of Model 1 (average based predicted topic)

Model 2 is a way which is used to calculate the similarity between document of test sets and all documents of training set, and select the highest similarity between them for allocating category to document of the test set.

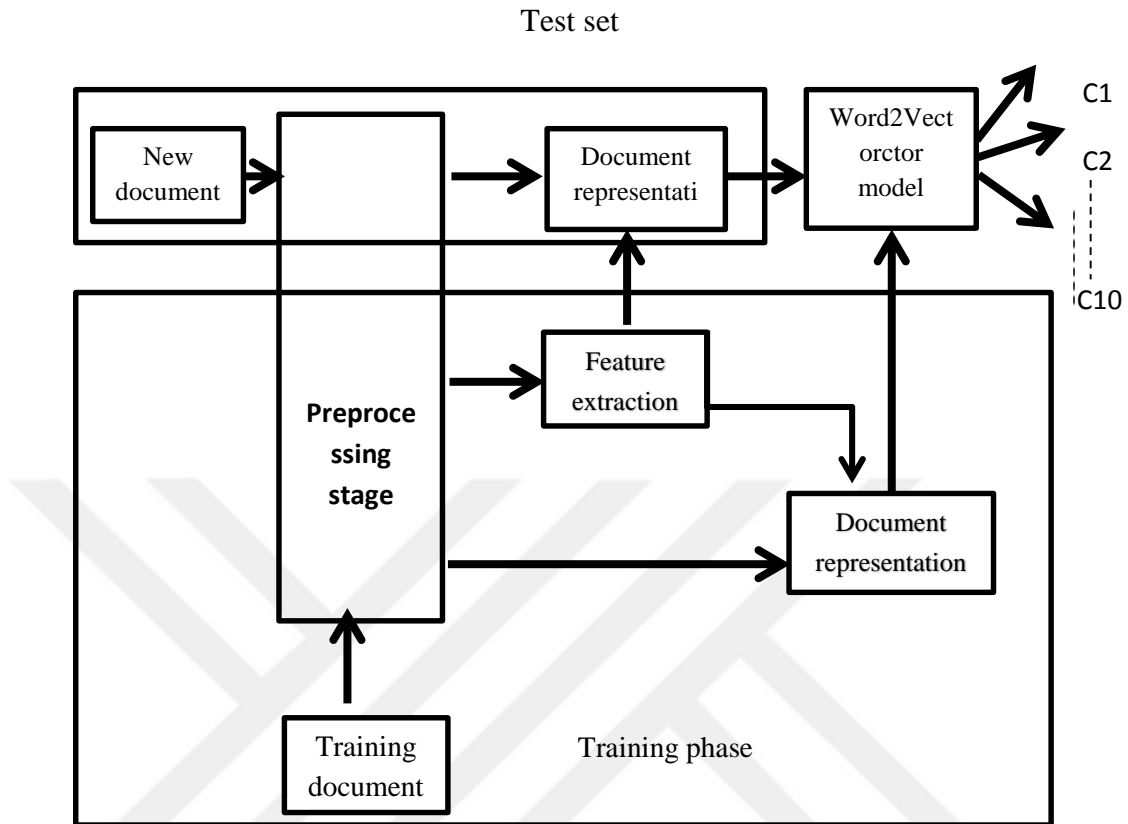


Figure 9. Schematic of Model 2 (the most similar predicted topic)

3.2 Reuters 21578 Data Set

Reuters21578 data set is a collection appeared in 1987 and it is classified manually in Reuters Ltd. This data set is split into training and testing set, adopted on the standard “Mod Apte”. It has been selected as a sample in this experiment and other studies because of following heuristic:

- An aggregate size of the corpus is huge for facility of experiment.
- It contains several predefined categories.
- A lot of their categories contain few documents. It has several documents categorized with multi-category and other documents with a single category.
- Some documents of Reuters are present without any actual textual content.

The Reuters collection involves 21,578 documents assembled from the Reuters Newswire that contains 135 classes which are defined previously. It has 13,476 documents which are categorized with at least one category, 7,059 documents unlabeled within any category, and 1,043 documents labeled as “bypass”. There are 2,308 documents from total 13,476 documents, are classified to be incorrect. So there are 9,338 documents classified with a single-labeled and 1,830 documents with a multi-label.

The version is achieved by removing unlabeled text, then choosing classes with a minimum one document in both the training and test data sets. Despite that dataset is widely used, researchers used various forms of the corpus. Most of them used “unlabeled” examples, while others considered examples with more than two thousand words. This large variety of data set and classes do not permit for a dependable comparison. The table below show the commonly used training and test example for every class. We built categorization models for the top 10 classes.

Table 4: Documents training and test for Reuters 21578

Category	Training examples	Test examples
Earn	2877	1087
Acq	1650	719
Money-fx	538	149
Grain	433	189
Crude	389	118
Interest	347	131
Ship	197	89
Wheat	212	71
Corn	182	56
Total	7194	2788

Text categorization challenges require large computational capabilities. A speedy computer with enough memory and Computer capability and suitable software systems

are mostly required. We would illustrate the hardware and software. The computer system composed of a multi-core server with the following configuration: (a) 2 Intel Quad Core Xeon @ 3.2 GHz, (b) 128 GB of RAM, And the Operating System used on this computer was the 64-bit version of the Ubuntu 16.4. The major modeling environment for preprocessing was the python 2.7 environment, and the Eclipse Neon.3 environment and install libraries of natural language toolkit (NLTK) platform. We found that these tools are very useful in our experiment.

3.3 Preprocessing phase (Noise Removal)

Generally, the importance of applying preprocessing methods on a representation document is to diminish the complication of the document and make them simpler to process, the text documents tuned to proper form for algorithms can handle them. Text representation is the significant attribute in the classification of documents, every document involves some noises that mislead the algorithm and cause increase in errors of classification. A noise is a word or term that it leads to increase classification errors when included in the document representation.

After a preprocessing phase, we start choosing feature extraction. These terms are extracted from a training set and used in document representation. Preprocessing provides two advantages for term extraction; the first it makes training model effective by reducing the volume of an active vocabulary, and the second is to enhance the categorization accuracy through ignoring noises feature [15].

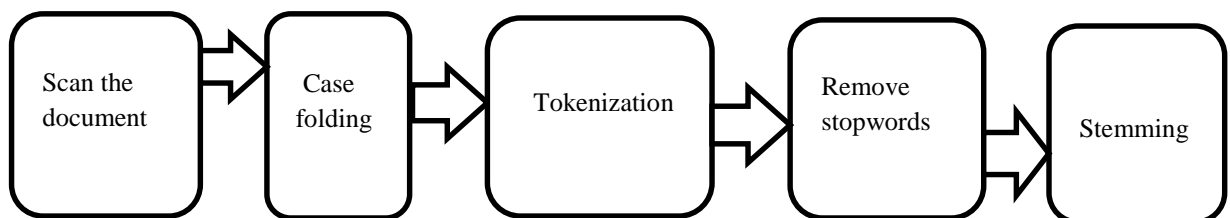


Figure 10. Stages of the preprocessing.

3.3.1 Parsing the Documents and Case-Folding

We removed all the special characters HTML and SGML from the text document for collection of Reuters21578 data set or corpus. In addition, it turns all characters in the text in the same case by using case-folding. Normally, all characters are changed to the lower case [16]. Mostly, case folding is regarded a useful strategy, like; ‘*Book*’ at the beginning of a sentence, identify the ‘book’ which it comes in the middle of a sentence.

3.3.2 Removing Stopwords

Stop words are words that widely occurring and their values seem insignificant in text classification. The stop words are based on their appearing in the entire collection, the aggregate number of every word in entire corpus is calculated, after that defines the common terms depend on specific criteria like (TF-IDF) that are used to demonstrate its relevance to the entire corpus that being classified. The list shown below is called a stop list that most frequently occurs in English [17].

A	an	and	Are	As	at	Be	by	For	from
Has	he	in	Is	It	its	Of	on	That	the
To	was	were	will	With					

Figure 11. List of stop words that used widely

3.3.3 Tokenization

Tokenization is splitting a text into words, sentences, paragraph or other significant features called tokens. The purpose of it is the extracting of terms from a text and know what are special characters that could be eliminated through tokenization [18]. Example:

Input: text classification has many applications;



Figure 12. Output tokenization by tokenizer.

Typically, the meaning of tokenization is not easily defined, so next heuristic assist to determine the token:

1. The most important for tokenization delimiter is the white space that can be in the most cases.
2. Although that Punctuation is regarded as a delimiter in the many cases, but it is removed by the tokenization.
3. Numbers are considered as delimiter tokenization but also it is removed after tokenization.

3.3.3.1 Difference between token and type:

Tokens represent the terms. The token indicates a series of characters in a specific text which they are associating one with the other jointly as a meaningful part for classifying [19], While a type is an abstract category of the entire tokens which they have the identical linguistic item. Type is different from the number of actual occurrences that it is known as tokens.

3.3.3.2 Penn Treebank Tokenizer

PTBTokenizer is a proper tool that used in order to tokenize text. In English language, there are 3 algorithms for Penn Treebank, the PTB tokenizer has a number of functionalities which help the Unicode character set compatibility [19]. PTBTokenizer is

an effective and fast tokenizer [19]. It's able to tokenize approximately million tokens each second. PTBTokenizer has many limitations that identify whether, single quotes, semicolon, colon or exclamation are a portion of the terms. In case periods which it doesn't mean boundaries of sentence, the classifier work well depends on a previously determined group of tokens.

3.3.4 Stemming

Stemming is the way converting words into a word in the basic form. The words appear in the text based on grammatical rules, like: the terms 'develop', 'develops', 'developer' and 'developing' all derived from the same stem or root 'develop'. The derived words can be similar or not to the stem word; but the important thing is mapping words that associated with the same stem, on the other hand it is not important the stem to be a valid root [20]. In the stemming algorithm the most important step is generating a word database by a previously-known group of rules. We mention some of the popular stemming algorithms.

1. Porter Stemmer
2. Lovins Stemmer
3. Paice Stemmer

Example for converting plural forms to singular as follows:

Table 5: Stemming Rules

Suffix	Stemmed to	Example
SES	S	addresses \Rightarrow address
IES	Y	Rallies \Rightarrow rally
SS	S	processes \Rightarrow process

We applied porter stemmer in python on tokens in the entire corpus. The Major challenge of applying stemming is changing the sense of the word significantly for example: the term “operate” in Operational Research, and Operating System could not be stemmed from “operate” because stem term drastically converts sense of them. The stammer has not been active in the preprocessing phase like to remove stop words.

3.4 Vector Space Model (Document Representation)

We applied preprocessing technique on a training dataset, which is converting irregular data into regular data, after that we need a document representation to create an effective text categorization model. Many machine learning algorithms cannot deal with textual information in their raw form due to their need for inputs in the numerical form to perform any sort of job [20]. Consequently, we used vector space model (VSM) which is considered as a model widely used for this issue.

The vector space model considers the document as a multi-dimension vector and a vocabulary extracted from dataset is a dimension of this vector [20]. We take this example for more illustrate:

The sentence =” **Vector space model is Word converted into numbers**”

Now we will extract a list of vocabulary from the sentence.

['vector', 'space', 'model', 'are', 'word', 'converted', 'into', 'numbers']

Vocabulary are every word did not repeat (unique) in the sentence. When we want representation of a vector for a word, 1 stands in the place of word in the sentence and other positions are 0 by using one-hot encoding vector. By sentence above if we want representation of a vector of a word 'number' in the sentence will be [0,0,0,0,0,0,1].

3.4.1 Term Frequency (TF)

The term frequency is considered as an easy method for representing a term in the vector space. So the weight of a term in textual document is equivalent to how many times this term appeared in the text [21]. Let us assume that we have a corpus C of D documents $\{d_1, d_2, \dots, d_D\}$ and we form our dictionary of unique terms which we extract them from corpus C and create term frequency matrixes that its size $(V \times D)$. Each document in corpus corresponds to row in matrix and each row contains the frequency of terms in a document. For example, we have two documents;

d1= He is a lazy boy. She is also lazy.

d2= Neeraj is a lazy person.

According to the example above the corpus $C = \{d_1, d_2\}$, we need to extract vocabulary for creating a dictionary of unique terms.

V = [he, lazy, boy, she, Neeraj, person]

Here we have $D = 2$ and $V = 6$, so the term frequency matrix will be $(D \times V)$ as shown in the table below.

Table 6: Term frequency matrix of various of term in two

	He	lazy	boy	she	Neeraj	person
d1	1	2	1	1	0	0
d2	0	1	0	0	1	1

From the table above every row refer to a document vector and every column refer to word vector.

3.4.2 TF-IDF Weighting

Generally, term frequency (TF) is concerned to measure the occurrence of words within boundary of a document, while inverse document frequency (IDF) is concerned to measure the occurrence of words over entire corpus [21]. The benefit from (IDF) is to reduce important common words occurring in the entire collection and gives importance to words that occur in one or subset of documents. Inverse document frequency processed all documents that have particular term equaled, this is a drawback IDF because it does not care whether the term has occurred one time in a document or more [21]. Let us take the following example; in the table below each column refers to a document d (column) and each row refers to the term t (row) and every intersection between d (i) and t (i) refer to time occurrence in the text.

Table 7: A Sample group Representing Term Frequency of Different Terms in a Set of Documents.

t/d	d1	d2	d3	d4	d5
t1	0	1	6	3	5
t2	2	1	0	2	3
t3	2	50	3	2	4
t4	3	2	3	2	3
t5	0	1	1	0	6
t6	0	1	0	0	3

Now let us a count (TF), (IDF) and (TF-IDF).

TF= (number of times term t occurrences in a document) / (Number of terms in the document) (3.4.1)

So TF (t3, d5) = (4/24) = 1/8

And TF (t1, d4) = (3/9) = 1/3

IDF= logs (N/n) (3.4.2)

Where N is the size of the corpus (number of documents)

n is the number of documents that the term appears in it.

So the IDF (t3) =logs (5/5) =0

And IDF (t2) =logs (5/4) = 0.09691

TF-IDF (t2, d1) = (2/7) *log (5/4) = -0.15545

TF-IDF (t3, d1) = (2/7) * 0 = 0

As seen in the equations above, the TF-IDF method gives importance for term t2, and it has penalized the term t3. It can be understood from that, the term t2 is an important for document d1 from the context of all documents.

CHAPTER IV

RESULTS

We present a report about experimental results. We started the experiment on March 17, 2017, on Information technology, Cankaya University. To evaluate the efficiency of Word2Vector in fix text organization challenges, its performance is estimated by utilizing a data set commonly used in text classification tasks that it is Reuters 21578 corpus.

4.1 Performance Measurement Metrics

The major statistics in terms of effectiveness are precision, recall and F1 measures. We used the same performance statistics in presenting our results to be capable of offering a comparative analysis.

In general, the text classification uses standard measure for scoring a performance, and is called F1score which it depends on two parameters basic that are the precision (p) and the recall (R) for benchmarking performance [19].

Precision (P) is defined as the amount of true positive results divided by the number of all positive results.

$$P = TP / (TP+FP) \tag{4.1.1}$$

Recall (R) is defined as the amount of true positive results divided by the number of positive results that should have been recalled.

$$R = TP / (TP+FN) \quad (4.1.2)$$

TP: Refer to true positive, FP: Refer to false positive, FN: Refer to false negatives.

The higher the value of F1 lead to the higher prediction accuracy, F1 measure could be understood as the mean of weighting precision and recall and it is measured from 0 to 1, as in the equation below:

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (4.1.3)$$

Micro-averaged F-Score is computed globally all results of the class. Precision and recall are calculated by aggregating all single results, and it is calculated as follows:

$$F1_{micro} = \frac{2 \cdot \sum_{c=1}^{NC} TP_c}{2 \cdot \sum_{c=1}^{NC} TP_c + \sum_{c=1}^{NC} FP_c + \sum_{c=1}^{NC} FN_c} \quad (4.1.4)$$

Where (NC) refer to a number of classes. In macro-averaging, F-score is calculated locally for every class initially after that the average is taken from all categories. Precision and recall are calculated for every class. Then F1-score is computed for each class and the macro-averaged F-measure is obtained by aggregating F-measure values for every category and divided by total of classes as follows:

$$F1_{macro} = \frac{\sum_c F1_c}{NC} \quad (4.1.5)$$

It would be important to obtain high accuracy in automated text categorization, so the accuracy would be high when assessing results of the classifiers on a training set, but the performance is not necessary to be of high accuracy when verifies it with the test set or any new data set. So the training and testing data are important to be from the same distribution.

4.2 Results for the Reuters-21578 Data Set

We present the results of our study in this chapter by using two models (Average Based Predicted Topic and The Most Similar Predicted Topics). This will be illustrated with tables and figures through using performance measurement metrics like precision, recall, and F1 scores in order to verify of performance.

4.2.1 Method 1 (Average based predicted topic)

In this model we applied preprocessing steps and created a dictionary from the documents having at least one topic in one of the top ten categories of Reuters 21578 data set as shown in table 8.

Table 8: Documents training and test for Reuters 21578 of model 1

Cat	Test examples	Training examples
Earn	1086	2877
Acq	696	1650
Trade	88	368
Crude	138	389
Interest	109	347
Ship	67	197
money- fx	112	583
Wheat	37	212
Grain	15	433
Corn	26	181
Total	2374	7237

After that the representation of documents in training and test set based on feature extraction is done, and we have selected every document from test set and compute the average similarity with a set of documents that belong to every category of training set,

and choose the category that it has the highest average similarity with the document of the test set, then compute precision, recall and F1score for measuring performance of model 1 as shown in the table below.

Table 9: Precision, recall and F1 score for model 1

Cat	Precision	Recall	F1_Score
earn	0.906077	0.906077	0.906077348
acq	0.892241	0.892241	0.892241379
trade	0.811765	0.784091	0.797687861
crude	0.652174	0.652174	0.652173913
interest	0.587156	0.587156	0.587155963
ship	0.375	0.358209	0.366412214
money-fx	0.357798	0.348214	0.352941176
wheat	0.314815	0.459459	0.373626374
grain	0.25	0.333333	0.285714286
corn	0.212121	0.28	0.24137931

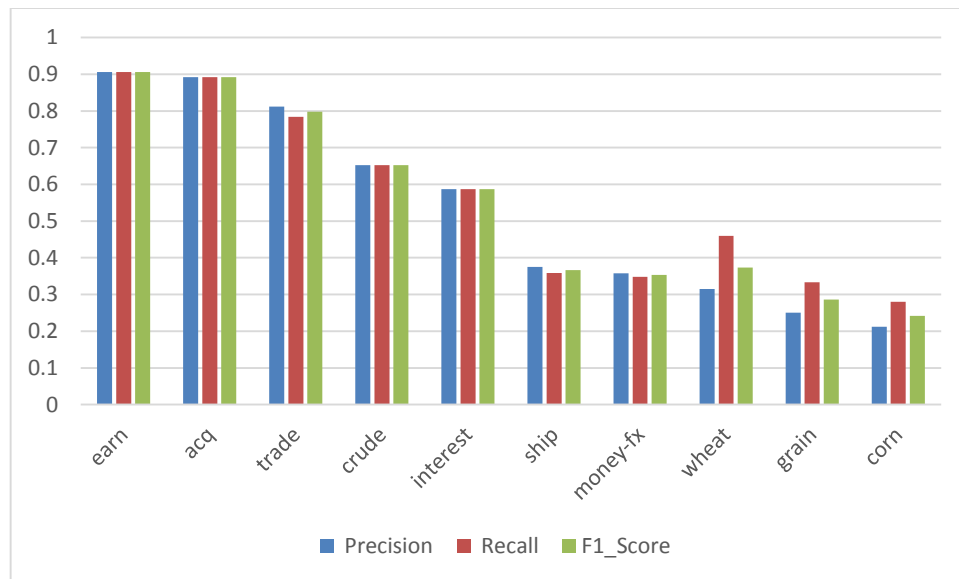


Figure 13. Precision, Recall , and F1 measure for model 1(average based prediction topic)

According to precision and recall as shown in table and figure above the best results were gained for categories that were more training examples (earn, acq, and trade), and the worst results were less training examples (wheat, grain, and corn), as with F1 scores, we found that (earn) is a category with the most accuracy (0.906077348) and (corn) is a category with the worst accuracy (0.24137931).

4.2.2 Model 2 (The Most Similar Predicted Topics)

In most similar prediction topic model for choosing a category for document test set we calculate similarity between the document of the test set with all documents of training set and select category of document of the training set that it has the highest similarity for allocating to document of the test set, we also used the top ten categories of Reuters 21578 as shown in the table below.

Table 10: Documents training and test for Reuters 21578 of model 2

Cat	Test examples	Training examples
Earn	1086	2877
Acq	696	1650
money- fx	112	538
Crude	138	389
Ship	67	197
Interest	109	347
Trade	88	368
Wheat	37	212
Grain	15	433
Corn	26	181
Total	2374	7237

After completely compute similarity for the entire corpus, and allocate categories of the documents of test set we assess the performance of model 2 by using precision, recall, and F1 scores as shown in the table below.

Table 11: Precision, recall and F1 score for model 2

Cat	Precision	Recall	F1_Score
Earn	0.985267	0.985267	0.985267
Acq	0.909613	0.91092	0.910266
money-fx	0.867257	0.875	0.871111
Crude	0.861111	0.898551	0.879433
Ship	0.768116	0.791045	0.779412
Interest	0.754386	0.788991	0.7713
Trade	0.897727	0.897727	0.897727
Wheat	0.75	0.891892	0.814815
Grain	0.8	0.8	0.8
Corn	0.571429	0.769231	0.655738

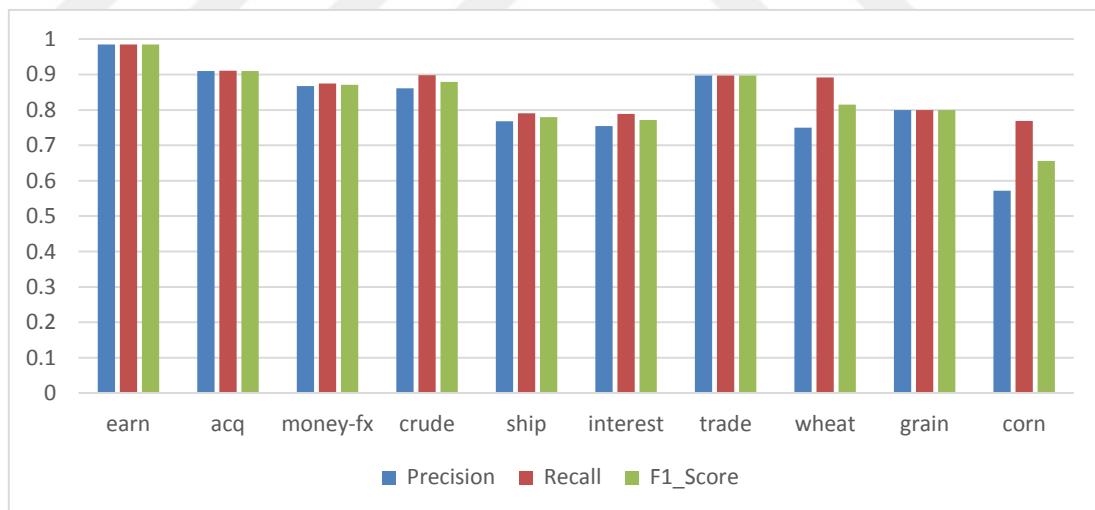


Figure 14. Precision, Recall, and F1 scores for Method 2 (the most similar predicted topic)

The table and figure above illustrate that the best results were categories which had more training examples (earn, acq, and money-fx), and the worst results were less training

examples (wheat, grain, and corn). We found to depend on F1 scores that (earn) is most accuracy (0.985267) and (corn) is the worst accuracy of the categories with (0.655738).

Table 12: Comparison measurements between two methods.

Measures	Model 1	Model 2
Micro_Precision	0.80200501	0.918503
Micro_Recall	0.8091024	0.930497
Macro_Precision	0.53591475	0.816491
Macro_Recall	0.56009555	0.860862
F1_Micro	0.80553807	0.924461
F1_Macro	0.54773841	0.83809

Table 12 shows the micro-averaged and macro-averaged for precision, recall and F1-score results for model 1 (average based predicted topic) and model2 (the most similar predicted topic). We demonstrate that model 2 obtains higher micro-averaged precision, recall, and F-score than model 1. So, model 2 is considered as the most robust way for finding the correct class, while model 1 is considered as a way which less successful in choosing the correct class. We observed that model 1 is more affected than model2 when the number of training examples are few and the performance of model 1 drops significantly as in (money-fx, grain, corn) that shown in table 9, while model2 is the less affected as shown in table 11.

We found that the results of model 2 depend on F1 score and the comparison with other studies that used the same top ten categories of Reuters 21578 and F1score as shown in the table below.

Table 13: Model 2, F1 values comparison with kNN, SVM, and DAN2

Categories	DAN2 F1	kNN F1	SVM F1	Model 2 F1_Score
Earn	99.23%	97.3%	98.5%	98.52%
Acq	93.85%	92%%	95.3%	91.02%
money-fx	83.17%	78.2%	75.4%	87.11%
Grain	96.21%	82.2%	91.9%	87.94%
Crude	94.1%	85.7%	89%	77.94%
Trade	84.07%	77.4%%	78%	77.13%
Interest	83.59%	74%	75%	89.77%
Ship	92.03%	79.2%	86.5%	81.48%
Wheat	88.28%	76.6%	85.9%	80%
Corn	98.53%	77.9%	85.7%	65.57%

In the table above, we demonstrate that the results of model 2 is parallel to the results of previous studies [22], [23], although the number of training examples is few as in (ship, wheat, corn) but the performance accuracy of F1 score is parallel to the result of previous studies [22], [23].

CHAPTER V

CONCLUSION

In our study on automatic text categorization, we found that most classifications approaches not depend on semantic meaning in the representation of documents, so we developed a model that depends on semantic meaning for text classification based on a Word2Vector model by measuring the similarity between documents as an alternative method to solve problems of document classification.

The results of our study show that the performance of model 2 (the most similar predicted topic) is better than the performance of model 1 (average based predicted topic) according to F1 measure, micro average F-score and macro average F-score which used for measuring the results of our experiment. We found that model 2 is better than model 1 and it is considered as the best way of classifying according to this study.

When we compare the results of our study with results of other studies which are conducted on the same data set (Reuters 21578),and used the same metrics as in Naïve Bayes (NB), k-Nearest-Neighbor (KNN), Support Vector Machines (SVM) [22] and dynamic architecture for artificial neural networks (DAN2) [23], we demonstrate that DAN2 and SVMs obtained the best results in most cases because both of them based on feature selection in improving their results, while the results of our study are parallel to them although it did not overcome other studies. So we can conclude from our study that it is a scalable

way because it depends on semantic meaning in text classification and has the capability to handle large vocabularies in comparison with other studies.

We used only Reuters 21578 dataset for measuring the performance of our study that depends on semantic meaning in the text classification. It would be significant in the future to realize whether the results which obtained from this dataset is similar to that obtained from other data sets. A Study could be done by applying part of speech instead of stemmer algorithm which is used in our study and know whether the results differ from results in our study.



REFERENCES

1. Falguni N. Patel, Neha R. Soni, "Text mining: A Brief survey", International Journal of Advanced Computer Research, Volume-2 Number-4 Issue-6 December-2012.
2. J. Rajni, M. Ruchika and Abha Jain, "Techniques for text classification: Literature review and current trends", Webology, Volume 12, Number 2, December, 2015.
3. L. Bing, L. Xiaoli, Wee Sun Lee and Philip S. Yu, "Text classification by labeling words.", American Association for Artificial Intelligence, AAAI. Vol. 4. 2004.
4. B. Tang, H. He, M. Baggenstoss and K. Kay, "A Bayesian Classification Approach Using Class-Specific Features for Text Categorization.", IEEE Transactions on Knowledge and Data Engineering, Issue Date: June 1. 2016.
5. B. Tang, H. He and K. Kay, "Toward Optimal Feature Selection in Naive Bayes for Text Categorization", IEEE Transactions on Knowledge and Data Engineering, Issue Date: Sept. 1 2016.
6. Xiao-Bing Xue and Zhi-Hua Zhou, "Distributional Features for Text Categorization", IEEE Transactions on Knowledge and Data Engineering, Volume: 21, Issue: 3, March 2009.
7. Y. Ping, Y. Zhou, Y. Yang, and W. Peng, "A novel term weighting scheme with distributional coefficient for text categorization with support vector machine

- information” Computing and Telecommunications (YC-ICT), 2010 IEEE Youth Conference on, Issue Date: 28-30 Nov. 2010.
8. A. Mehdi, P. Seyedamin, A. Mehdi, S. Saied, B. Gutierrez, and K. Krys,” A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques”, KDD Bigdas, arXiv: 1707.02919v1 [cs.CL] 10 Jul 2017.
 9. A. McCallum and K. Nigam, “A Comparison of Event Models for Naive Bayes Text Classification,” ,in Proc. AAAI Workshop On Learning For Text Categorization,1998, pp. 1-42.
 10. G. SALTON and C. S. YANG,” On the Specification of term values in automatic indexing”, THE Journal of Documentation VOLUME 29 NUMBER 4 DECEMBER 1973.
 11. Miller, G. Beckwith, R. Fellbaum, C. Gross, and D. Miller,” Five Papers on WordNet Technical report”, Cognitive Science Laboratory, Princeton University, 1993.
 12. A. Berna, Murat Can Ganiz , and D. Banu,” A corpus-based semantic kernel for text classification by using meaning values of terms”, Engineering Applications of Artificial Intelligence 43 (2015) 54–66.
 13. Deeplearning4j Development Team. Deeplearning4j: Open-source distributed deep learning for the JVM, Apache Software Foundation License 2.0. <http://deeplearning4j.org>.
 14. David Meyer,” How exactly does Word2Vector work”, 3.pdf, July 31, 2016, pp.1-9.
 15. J. Batiz-Benet, Q. Slack, M. Sparks, and A. Yahya, “Parallelizing Machine Learning Algorithms,” in Proc. ACM Symp., 2012, pp. 12-81.

16. D. Beeferman, A. Berger and J. Lafferty, "Statistical Models for Text Segmentation," In Proc. AIT, 2013, pp. 4-15.
17. M. Berenjkoob, R. Mehri, H. Khosravi, and M. Ali, "A Method for Stemming and Eliminating Common Words for Persian Text Summarization," published in Natural Language Processing and Knowledge Engineering (NLP-KE), 2009, pp.1-6.
18. G. Filip, J. Krzysztof, W. Agnieszka and W. Mikołaj, "Text Normalization as a Special Case of Machine Translation," In Proc. Int. Conf. on Computer Science and Information Technology, 2006, pp. 51–56.
19. H. Schütze, D. Manning, and P. Raghavan, "StanfordTokenize," [http://nlp.stanford.edu / software/tokenizer.shtml](http://nlp.stanford.edu/software/tokenizer.shtml)", 2009, pp. 3-14.
20. J. Dean "Experiences with MapReduce, an Abstraction for Large-Scale Computation," in Proc. 15th Int. Conf. on Parallel architectures and compilation techniques, 2006, pp. 1-7.
21. H. Trevor, T. Robert, and F. Jerome, "The Elements of Statistical Learning: Data Mining", Inference, and Prediction, 2nd ed. Springer-Verlag, New York, 2001.
22. I. Feinerer, "A framework for text mining applications within R", <http://cran.rproject.org/web/packages/tm/index.html>., Accessed 7 June 2009.
23. M. Ghiassi , M. Olschimke, B. Moon, and P. Arnaudo, "Automated text classification using a dynamic artificial neural network model", Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053-0388, USA, Expert Systems with Applications 39 (2012) 10967–10976.

APPENDIX A

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Ather Abdulrahem Mohammedsaed
ALSAMURAI

Nationality: Iraq

Marital status: Married

Phone: 009647700038799

E-mail: snrs001@gmail.com



EDUCATION

Degree	Institution	Year of Graduation
M.Sc.	Çankaya University computer engineering	2017
B.Sc.	Al-Yarmouk university	2002
High School	Al Sharif Alradi	1997

FOREIGN LANGUAGES

English