The Islamic University – Gaza

Denary of Higher Studies

Faculty of Information Technology

Arabic Text Classification Using Learning Vector Quantization

By

Mohammed N. Azarah

Supervisor

Dr. Alaa El-Halees

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master In Information Technology

2012 – 1433 H

Acknowledgements

First and foremost, I could never forget Dr. Alaa El-Halees who walked with me on the first steps with this work and from whom I have learned a lot, due to his supervision, guidance, support and advising till this work come to light.

I would like to thank Mr. Motaz Saad, for his valuable scientific and technical notes.

Also, I would like to express my gratitude to all staff in the faculty of Information Technology at Islamic University of Gaza whose gave me the help and idea of this thesis with his great experience.

Finally, my deepest thanks go to my parents and my wife for their unconditional love, and to my friends for their support.

This thesis would have been much different (or would not exist) without these people.

Abstract

Text classification aims to automatically assign document in predefined category. In our research, we used a model of neural network which is called Learning Vector Quantization (LVQ) for classifying Arabic text. This model has not been addressed before in this area. The model based on Kohonen self organizing map (SOM) that is able to organize vast document collections according to textual similarities. Also, from past experiences, the model requires less training examples and much faster than other classification methods.

In this research we first selected Arabic documents from different domains. Then, we selected suitable pre-processing methods such as term weighting schemes, and Arabic morphological analysis (stemming and light stemming), to prepare the data set for achieving the classification by using the selected algorithm. After that, we compared the results obtained from different LVQ improvement version (LVQ2.1, LVQ3, OLVQ1 and OLVQ3). Finally, we compared our work with other most known classification algorithms; decision tree (DT), K Nearest Neighbors (KNN) and Naïve Bayes. The results presented that the LVQ's algorithms especially LVQ2.1 algorithm achieved high accuracy and less time rather than others classification algorithms and other neural networks algorithms.

Keywords

Arabic Text Mining, Arabic Text Classification, Arabic Text Categorization, Learning Vector Quantization, LVQ, Artificial Neural Network, ANN.

الملخص

عملية تصنيف النصوص تهدف إلى تعيين النص في الفئة المناسبة المحددة مسبقاً، في هذا البحث سأستخدم نموذج يعتمد على الشبكات العصبية لتصنيف النصوص العربية يسمى التعلم بتكميم المتجهات (LVQ) الذي لم يستخدم من قبل في مجال النصوص العربية. هذه الخوارزمية تعتمد على خرائط التنظيم الذاتي (SOM) التي تمتلك القدرة على ترتيب كميات ضخمة من المستندات النصية إلى فئات متشابهة، ومن مميزات هذه الخوارزمية تحتاج لعدد قليل من الأمثلة لعمل تدريب للشبكة العصبية وأيضاً سرعة أكبر من طرق تصنيف أخرى.

في البداية قمت باختيار مجموعة من النصوص العربية من أكثر من مجال، بعد ذلك قمت باختيار طرق مختلفة لتجهيز النص مثل توزين الكلمات و معالجة النصوص المعالجة المصرفية (التجذير والتجذير الخفيف للكلمات العربية). بعد ذلك قارنت بين النتائج التي حصلت عليها من عملية تجريب الخوارزمية وكافة النسخ المعدلة على الخوارزمية (LVQ2.1,LVQ3,OLVQ1,OLVQ3). في النهاية تمت مقارنة النتائج مع خوارزميات تصنيف مشهورة مثل (decision tree (DT), K Nearest Neighbors (KNN) , Naïve Bayes) . أظهرت النتائج أن الخوارزمية LVQ2.1 قد حققت نتائج أكثر دقة وأقل وقت عند مقارنتها بكافة النسخ المعدلة للخوارزمية وأيضا خوارزميات أخرى للشبكات العصبية.

كلمات مفتاحية

تنقيب النصوص العربية، تصنيف النصوص العربية، الشبكات العصبية، LVQ.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

TM	Text Mining
TC	Text Classification/ Text Categorization
ANN	Artificial Neural Network
NN	Neural Network
DM	Data Mining
KNN	K Nearest Neighbors
DT	Decision Tree
OSCA	Open Source Arabic Corpus
LVQ	Learning Vector Quantization
SOM	self organizing map
NLP	Natural Language processing
SVM	Support Vector Machine
MLP	Multilayer Perceptron
BP	Back propagation
RBF	Radial Basis Function
BPNN	Back Propagation Neural Networks
SVD	Value Decomposition
IDF	Inverse Document Frequency
TF-IDF	Term Frequency-Inverse Document Frequency
WEKA	Waikato Environment for Knowledge Analysis
HMM	Hidden Markov Models

CHAPTER 1: Introduction

Text classification (TC) is an intersection of Natural Language Processing (NLP) field and data mining field. It can be defined as the assignment of unclassified document to one or more predefined categories based on their content. Automatic TC is very useful in terms of time and expense that it saves. Many methods and algorithms have been applied to the problem of text classification. These methods vary in their accuracy and computation efficiency [1] .

Text classification have been used in many applications such as real time sorting of files into folder hierarchies, topic identifications, dynamic task-based interests, automatic meta-data organization, text filtering and documents organization for databases and web pages [2]. In Arabic where there are huge amount of documents in an organization as textual. We need text classification to categorize collection of Arabic text document, which finding the correct topic for each document [3].

There are several methods used to classify text such as Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Artificial Neural Networks, Naïve Bayes Classifier, and Decision Trees. Artificial Neural Networks (ANN) is one of methods used for classification. ANN structure contains input layer, output layer and with or without hidden layer based in type of ANN. The ANN widely applied by many researchers to classify text documents with different types of algorithms. In this research, Linear Vector Quantization (LVQ) one of the neural network models used for classification [4].

The LVQ model is a classification method based on Kohonen 1995. The original of Kohonen model is known as Self Organizing Map networks (SOM) [5]. It uses a competitive unsupervised learning algorithm [4]. The LVQ is a supervise version of Kohonen model. The architecture of LVQ model is simple and does not include any hidden layer, so the neural network consists only of one input layer and one output layer [4]. In this model, the output layer has a neurons equal to the number of classes [5]. The weight vector associate to each output unit are known as codebook vector. Each class of input space is represented by its own codebook vector. Codebook vectors are defined according to the specific task. For the categorization task we use one codebook vector per class. The category label for each codebook vector is the class name [4].

The advantage of LVQ is that [4, 6, 7]:

1) It requires less training examples.

2) Much faster than other classification methods.

3) It has a better accuracy results than other methods

There are different type or algorithms for LVQ algorithm which used as optimization of original LVQ [6, 8]:

- LVQ1 (LVQ),
- LVQ2.1,
- LVQ3, and
- Optimized learning rate algorithms OLVQ1,
- OLVQ3 for the classification task.

More details for LVQ's algorithms will be discussed in chapter 2.

1.1 Arabic Language

Arabic Language is one of the widely used languages in the world. Arabic language is a Semitic language that has a complex and much morphology than English, it is a highly inflected language and that due to this complex morphology [3, 9-12].

Arabic Language consist of 28 alphabet characters: ا ب ت ث ج ح خ د ذ ر ز س ش ص ض ط ظ ع غ ف ق ك ل م ن ه و ي . In addition to the hamza (ء) which is considered as a letter by some Arabic linguistics. Arabic is written from right to left. Arabic letters have different styles when appearing in a word depending on the letter position at beginning, middle or end of a word and on whether the letter can be connected to its neighbor letters or not [13]. For example the position for letter (ص) has the following style (صـ) when appears at the beginning of word such as صبر which means patience; (ـصـ) if the letter appears in the middle of a word such as يصبر which means patient; (ـص) if the letter appears at the end of a word such as the word قفص which means cage. Arabic language has diacritics which are signals placed below or above letters. Arabic diacritics include: shada, dama, fatha, Kasra, sukon, double dama, double fatha, double kasra.

Arabic words have two genders, feminine and masculine; three numbers, singular, dual, and plural; and three grammatical cases, nominative, accusative, and genitive. A noun has the nominative case when it is subject; accusative when it is the object of a verb; and the genitive when it is the object of a preposition. Words are classified into three main parts of speech, nouns (including adjectives and adverbs), verbs, and particles. All verbs and some nouns are morphologically derived from list of roots.

Words are formed by the following fixed patterns, the prefixes and suffixes are added to the word to indicate its number, gender and tense.

1.2 Arabic Language Challenges

Arabic is a challenging language for a number of reasons [9]:

1. Orthographic with diacritics is less ambiguous and more phonetic in Arabic, certain combinations of characters can be written in different ways. For example, sometimes in glyphs combining HAMZA with ALEF (أ) the HAMZA is dropped (ا). This makes the glyph ambiguous as to whether the HAMZA is present.

2. Arabic has a very complex morphology recording as compared to English language. For example, to convey the possessive, a word shall have the letter (ى) attached to it as a suffix. There is no disjoint Arabic-equivalent of "my".

3. Arabic words are derived: Arabic words are usually derived from a root (a simple bare verb form) that usually contains three letters. In some derivations, one or more of the root letters may be dropped. In such cases tracing the root of the derived word would be a much more difficult problem.

4. Broken plurals are common. Broken plurals are somewhat like irregular English plurals except that they often do not resemble the singular form as closely as irregular plurals resemble the singular in English. Because broken plurals do not obey normal morphological rules, they are not handled by existing stemmers.

5. In Arabic we have short vowels which give different pronunciation. Grammatically they are required but omitted in written Arabic texts.

6. Arabic synonyms are widespread. Arabic is considered one of the richest languages in the world. This makes exact keyword match is inadequate for Arabic retrieval and classification.

From previous challenges Arabic language needs a set of preprocessing routines to be suitable for manipulation which will be handled through chapter 4.

1.3 Research Motivation

Over the year, the electronic text information increasingly through organization and internet. Arabic text document is one of the most famous documents and needs to apply

text mining methods as any other languages. Text classification can be applied in importance operations such as real time sorting of files into folder hierarchies, topic identifications, dynamic task-based interests, automatic meta-data organization, text filtering and documents organization for databases and web pages. There are a lot of researches for classification English language text but in Arabic language text is limited. LVQ algorithm proved that it required less training examples and are much faster than other classification methods. Also, it has a better accuracy results than other methods. LVQ algorithms not used previously for Arabic language.

1.4 Research Problem

The main research problem is the works on classification for Arabic text are limited. These works consume time especially on training and can have more accurate results.

1.5 Research Objectives

1.5.1 Main objective

The main objective of this research is to classify Arabic text by using neural network algorithms called Linear Vector Quantization (LVQ) for better performance in term of time and accuracy.

1.5.2 Specific objectives

The specific objectives of the research are:

- Find the best pre-processing applicable to LVQ method.
- Study whether LVQ's algorithms as suitable model for classifying Arabic text.
- Find which version of LVQ algorithms gets the best result for classifying Arabic text document.
- Applicability of the approach to work with different domains.
- Evaluate Arabic Text classification using LVQ approach with respect to accuracy, recall, precision , f-measure and training time needed to build model.

1.6 Research Scope and limitations

The research has the following Scope and limitations:

1- It will not propose any new pre-processing method.

2- The research will not modify any LVQ algorithm.

3- The results will be compared to only the most famous data mining methods which are decision tree (DT), K Nearest Neighbors (KNN) and Naïve Bayes.

1.7 Thesis Structure

The rest of the research is organized as follows: chapter 2 literate view; chapter 3 related work; chapter 4 methodology; chapter 5 presents experimental results and analysis; and finally, chapter 6 we draw future work and conclusion.

CHAPTER 2: Literature Review

This chapter introduces the theoretical foundation of the research. It includes the following topics: Data Mining (DM) ,Text Mining (TM) which are the main topics of the research. Then we introduce classification, and the famous classification methods. During these methods I will compare my proposed method with them. Then, Artificial Neural Network (ANN) algorithms and Learning Vector Quantization (LVQ) will be introduced which are my proposed method.

2.1 Data Mining (DM)

Data Mining (DM) or knowledge discovery, is the computer-assisted process of digging through and analyzing enormous sets of data and then extracting the meaning of the data. Data mining tools predict behaviors and future trends, allowing businesses to make proactive, knowledge-driven decisions [14].

The knowledge discovery goals are defined by the intended use of the system. We can distinguish between two types of goals: (1) verification and (2) discovery. With verification, the system is limited to verifying the user's hypothesis which is mainly used in statistics. With knowledge discovery, the system autonomously finds new patterns. We can further subdivide the discovery goal into prediction, where the system finds patterns for predicting the future behavior of some entities, and description, where the system finds patterns for presentation to a user in a human-understandable form. In this article, we are primarily concerned with discovery-oriented data mining [14-16].

Data mining involves fitting models to, or determining patterns from, observed data. The fitted models play the role of inferred knowledge: Whether the models reflect useful or interesting knowledge is part of the overall, interactive knowledge discovery process where subjective human judgment is typically required. Two primary mathematical formalisms are used in model fitting: (1) statistical and (2) logical.

The statistical approach allows for nondeterministic effects in the model, whereas a logical model is purely deterministic [14].

The most important tasks in data mining are *supervised learning*, with a known output variable in the dataset, including (a) classification: class prediction, with the variable typically coded as an integer output; (b) fuzzy classification: with gradual memberships with values in-between 0 and 1 applied to the different classes; (c) regression: prediction of a real-valued output variable, including special cases of predicting future values in a time series out of recent or past values.

The *unsupervised learning* which means without a known output variable in the dataset, including (a) clustering: finds and describes groups of similar examples in the data using crisp of fuzzy clustering algorithms; (b) association learning: finds typical groups of items that occur frequently together in examples. The last method is *semisupervised learning*, whereby the output variable is known only for some examples [14, 15].

Each of these tasks consists of a chain of low level tasks. Furthermore, some low-level tasks can act as stand-alone tasks; for example, by identifying in a large dataset elements that possess a high similarity to a given example. Examples of such low-level tasks are:

- data cleaning (e.g., outlier detection).
- data filtering (e.g., smoothing of time series).
- feature extraction from time series, images, videos, and graphs (e.g., consisting of segmentation and segment description for images, characteristic values such as community structures in graphs).
- feature transformation (e.g., mathematical operations, including logarithms, dimension reduction by linear or nonlinear combinations by a principal component analysis, factor analysis or independent component analysis);
- feature evaluation and selection (e.g., by filter or wrapper methods);
- computation of similarities and detection of the most similar elements in terms of examples or features (e.g., by k-nearest-neighbor methods and correlation analysis);
- model validation (cross validation, bootstrapping, statistical relevance tests and complexity measures);
- model fusion (mixture of experts); and
- model optimization (e.g., by evolutionary algorithms).

2.2 Text Mining (TM)

Text Mining (TM), also known as text data mining or knowledge discovery from textual databases, refers generally to the process of extracting interesting and non-trivial patterns or knowledge from unstructured text documents. It can be viewed as an extension of data mining or knowledge discovery from (structured) databases [17].

Text mining is well motivated, due to the fact that much of the world's data can be found in text form (newspaper articles, emails, huge documents, web pages, etc.).

Text mining has the same goals as data mining including, text categorization, clustering, document summarization, and extracting useful knowledge. Text mining must overcome a major difficulty that there is no explicit structure [9].

Text mining doing some preprocessing to convert unstructured document to structural document then doing data mining operations. It usually involves the process of structuring the input text (parsing, along with the addition of some derived linguistic features and the removal of others), deriving patterns within the structured data, and finally evaluation and interpretation of the output. High quality in text mining usually refers to some combination of relevance, novelty, and interestingness [9].

In this thesis we are interested to discover new knowledge from text by using classification algorithms which is described in the next section.

2.3 Classification

Text classification or categorization is one of the important tasks in text mining process [18].This is because of the significance of natural language text, the huge amount of text stored on the internet, and the available information libraries and document corpus. Text classification refers to the process of grouping unclassified text document into predefined group or categories, giving it one or more category label, in machine learning approaches to text categorization, this is done by learning the system how to classify through examples [19]. This type of categorization is called supervised learning because target class based on predefined classes.

2.4 Classification Methods

Classification involves a lot of methods. Thus, a large number of algorithms used for document. Major kinds of classification method used are: decision tree (DT), K Nearest Neighbors (KNN) ,Naïve Bayes and Artificial Neural Networks.

In the following section we introduce several classification algorithms used during experiments results.

2.4.1 Naïve Bayes

Naïve Bayes is one of the most widely used classifiers. Naïve Bayes is a type of statistical classification based on Bayesian theorem. Among others classifiers Naïve Bayes is the simplest and used in any research related of classification [20]. Naïve

Bayesian classifier assumes (naively) that features of the input feature vector (usually word distribution) are statistically independent [20].

2.4.2 Decision Tree (DT)

This type of classifiers builds trees by sorting them based on feature values. The tree contains root and nodes each branch represents a value that the node can assume. Instances are classified starting at the root node and sorted based on their feature values. In the last DT, classifier distributes each instance to suitable branch based on target class [14].

The most well-know algorithm in the literature for building decision trees is the C4.5. C4.5 which is an extension of Quinlan's earlier ID3 algorithm. The decision tree generated by C4.5 and can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier [13].

2.4.3 K-Nearest Neighbors (K-NN)

K-NN is a simple efficient example based approach for text categorization, the K-NN algorithm can work as follows: given a test document to be classified, the algorithm searches for the k nearest neighbors among the pre-classified training documents based on some similarity measure, and ranks those k neighbors based on their similarity scores, the categories of the k nearest neighbors are used to predict the category of the test document by using the ranked scores of each as the weight of the candidate categories, if more than one neighbor belongs to the same category then the sum of their scores is used as the weight of that category, the category with the highest score is assigned to the test document provided that it exceeds a predefined threshold, more than one category can be assigned to the test document.

The draw back in k-NN is the difficulty to determine the value of k, a series of experiments with different k values should be conducted to determine the best value of k, another disadvantage of k-NN is the complexity of computation time needed to traverse all the training documents [19].

2.5 Artificial Neural Network (ANN)

Artificial Neural Networks (ANNs) is one type of network sees the nodes as ‘artificial neurons’. An artificial neuron is a computational model inspired in the natural neurons. Natural neurons receive signals through *synapses* located on the dendrites or

membrane of the neuron. When the received signals are strong enough (surpass a certain threshold), the neuron is activated and emits a signal through the axon. This signal might be sent to another synapse, and might activate other neurons [21]. The complexity of real neurons is highly abstracted when modeling artificial neurons. These basically consist of inputs (like synapses), which are multiplied by weights (strength of the respective signals), and then computed by a mathematical function which determines the activation of the neuron. Another function (which may be the identity) computes the output of the artificial neuron (sometimes in dependence of a certain threshold). ANNs combine artificial neurons in order to process information [9,10]. In practice, there are two kinds of NN architectures, the feed forward NNs and the feedback or recurrent ones applied in totally different problem domains [22].

Multilayer Perceptron (MLP) is a common NN architecture applied in many areas of research, when solutions to diverse and difficult problems are required, typically, an MLP is a structure trained to map specific inputs to outputs through nonlinear functionality. This mapping is performed by passing inputs to outputs through one or more strongly connected hidden layers of computation nodes and a final layer of output nodes. The number of the hidden layers is critical parameter and effects on performance for NNs, and also their corresponding neurons, the NN's weights and the hidden layers' transfer functions. Regarding the network weights, the MLPs use the error back propagation algorithm to train their values on the supervised learning phase [22, 23].

Artificial Neural Networks (ANNs) are well-established tools used successfully in many problems such as pattern recognition, classification problems, regression problems differential equations, etc [24]. Further, neural network is a popular classification method, it can handle linear and nonlinear problems for text categorization, and both of linear and nonlinear classifier can achieve good results [25]. Several of research used neural network for solving text classification problem. There are many ANN models used for classification namely back propagation (BP) networks, radial basis function (RBF) networks, kohonen networks and learning vector quantization (LVQ) networks. In this thesis we used LVQ for text classification and compare with others NNs.

2.5.1 Back Propagation Neural Networks (BPNN)

Back Propagation (BP) is the famous ANN model. The artificial neurons are organized in layers and send their signals "forward", and then the errors are propagated backwards. When using BP in text classification the inputs are the components of the document vector, and the outputs are the document categories [21, 26].

The structure of the three layered back propagation neural network (BPNN) is shown in Figure 2.1. The network receives inputs by neurons in the *input layer*, and the output of the network is given by the neurons on an *output layer*. There may be one or more intermediate *hidden layers*. The back propagation algorithm uses supervised learning, which means that we provide the algorithm with examples of the inputs and outputs we want the network to compute, and then the error (difference between actual and expected results) is calculated. The idea of the back propagation algorithm is to reduce this error, until the ANN *learns* the training data. The training begins with random weights, and the goal is to adjust them so that the error will be minimal [21].

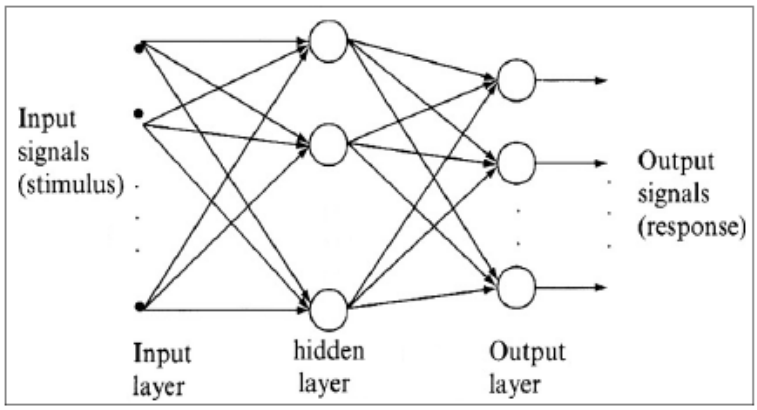

Figure 2.1: Typical three layered back propagation neural network [25]

2.5.2 Radial Basis Function Neural Network (RBF-NN)

A Radial Basis Function Neural Network (RBF-NN) is a special type of ANN with several distinctive features [27]. The radial basis function (RBF) network is a feed-forward artificial neural network that uses radial basis functions as activation functions. It is based on linear combination of radial basis functions. It has been shown that RBF-NN had a simple structure and many excellent performances. Therefore, RBF-NN has been widely used for pattern classification, functional approximation, signal processing, mixture models and many other fields [22].

A RBF-NN consists of three layers namely the input layer, the hidden layer, and the output layer. Each RBF is a fixed two layer NN that hides all nonlinearities in its

special hidden layer and performs a linear combination in the output layer show Figure 2.2. The parameters that determine the output values are the centers of the hidden RBF units and the weights of the synapses from the hidden to the output layer. One important difference is that MLP is most of the times a multi-layer network, whereas each RBF-NN consists of only one hidden layer with radial basis function neurons. In a typical implementation, the hidden nodes and the output nodes of a MLP-NN share a common computation and neuronal model, while the RBF-NN neurons of the RBF hidden layer plays a totally different role in comparison to the output nodes which are most of the times linear and perform a linear combination of the hidden layer neurons' responses [22, 23].

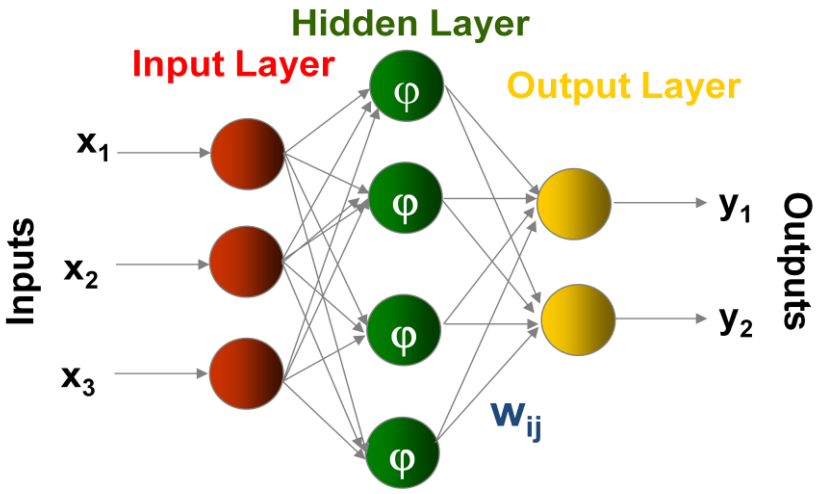

Figure 2.2 Architecture of Radial Basis Function (RBF-NN)

2.5.3 Kohonen network

The Kohonen neural network differs considerably from the feed forward back propagation neural network. The Kohonen neural network differs both in how it is trained and how it recalls a pattern. The Kohohen neural network, does not use any sort of activation function. Further, the Kohonen neural network does not use any sort of a bias weight [8, 28, 29].

Output from the Kohonen neural network does not consist of the output of several neurons. When a pattern is presented to a Kohonen network one of the output neurons is selected as a "winner". This "winning" neuron is the output from the Kohonen network. Often these "winning" neurons represent groups in the data that is presented to the Kohonen network.

The most significant difference between the Kohonen neural network and the feed forward back propagation neural network is that the Kohonen network trained in an unsupervised mode. This means that the Kohonen network is presented with data, but the correct output that corresponds to that data is not specified. Using the Kohonen network this data can be classified into groups. We will begin our review of the Kohonen network by examining the training process.

The Kohonen neural network contains only an input and output layer of neurons. There is no hidden layer in a Kohonen neural network. First we will examine the input and output to a Kohonen neural network. The input to a Kohonen neural network is given to the neural network using the input neurons. These input neurons are each given the floating point numbers that make up the input pattern to the network. A Kohonen neural network requires that these inputs are normalized to the range between -1 and 1. Presenting an input pattern to the network will cause a reaction from the output neurons.

The output of a Kohonen neural network is very different from the output of a feed forward neural network. For example, if we had a neural network with five output neurons we would be given an output that consisted of five values. This is not the case with the Kohonen neural network. In a Kohonen neural network only one of the output neurons actually produces a value. Additionally, this single value is either true or false. When the pattern is presented to the Kohonen neural network, one single output neuron is chosen as the output neuron. Therefore, the output from the Kohonen neural network is usually the index of the neuron that fired. The structure of a typical Kohonen neural network is shown in Figure 2.3.

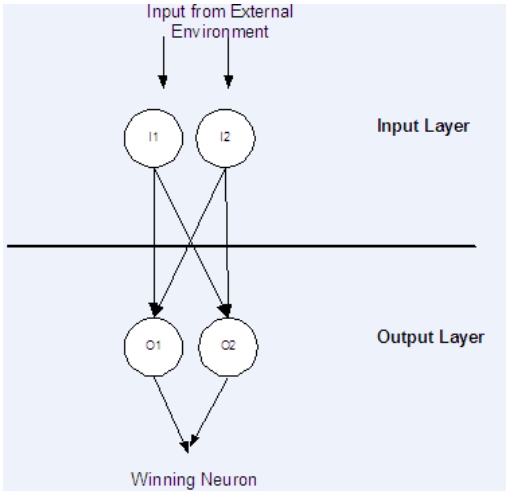

Figure 2.3: A Kohonen Neural Network [29]

2.6 Learning Vector Quantization (LVQ)

Learning Vector Quantization (LVQ) is one of ANNs models. LVQ focuses on classification [28]. This models based on Kohonen model. The original of Kohonen model is known as Self Organizing Map networks (SOM) which used for unsupervised learning [30-32]. The supervised-training algorithm suggested by Kohonen is known as the Learning Vector Quantization (LVQ). Further, LVQ is a supervised version of Kohonen model [4].

A comparison study between unsupervised (SOM) and supervised (LVQ) kohonen models showed that the LVQ algorithm performs slightly better than the SOM model. The experiments applied into Time Magazine collection and, again, from Yahoo financial news, the best results were obtained with the LVQ algorithm. Also, some differences will be discussed in next sections [4].

2.6.1 SOM and LVQ models Architecture

The SOM contains two layers input layer and competitive or output layer Figure 2.4. The input data, that can be represented in the general case in the form of vectors with N dimensions, are converted into classes which self-organize according to a two-dimensional structure of neurons on which neighborhood relations are preset. The process of SOM topographic classification thus combines a stage of classification with one of data projection. In this connectionist model, two layers of neurons are used: the first encodes the inputs and the second computes the outputs (classes). The two layers are entirely connected. After training, a neuron in this layer can be activated by input vectors corresponding to various classes, which is a problem in decision making and neuron labeling process. To overcome this limitation, a second training phase supervised this time is applied. This phase allows a readjustment of the distribution probability and of the labels attributed to neurons in the output map, in such a way that only one class is attributed to each neuron [32].

The second model known as the Learning Vector Quantization (LVQ). The architecture of the LVQ is similar to that of the SOM map, without lateral connections on the neurons of the second layer Figure 2.4. With its various alternatives, this algorithm improves the separation in classes from the solution suggested by the unsupervised training. For a given input, the method consists of bringing closer the most activated neuron if it is in the right class (supervised training), and to push it back in the

opposite case. The losers neurons remains unchanged. Each neuron thus becomes class representative [32].

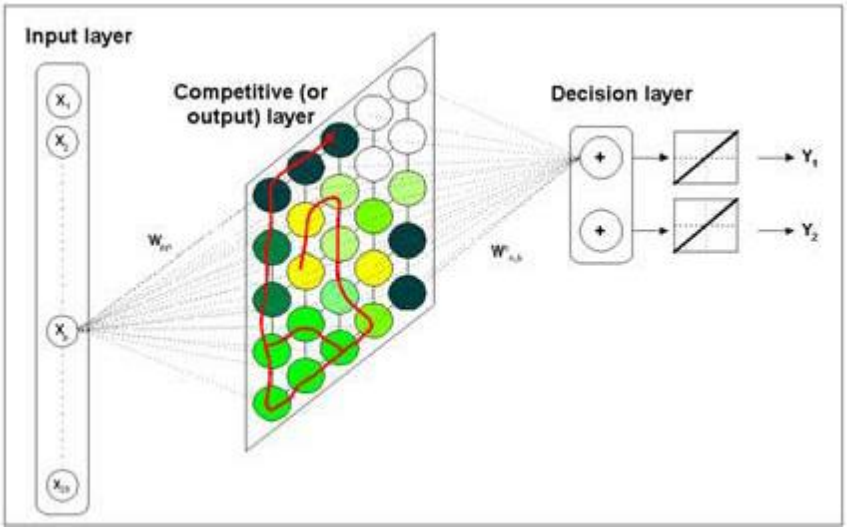

Figure 2.4: From left to right, the first two layers represent the general architecture of the Self Organizing Map (SOM) The other two layers indicate the principle of Learning Vector Quantization (LVQ) [32].

2.6.2 LVQ Algorithms

The motivation for the algorithm for the LVQ network is to find the output unit *w* that is closest to the input vector *x*. If *x* and *w* belong to the same class, then we move the weights toward the new input vector; if *x* and *w*, belong to different classes, then we move the weights away from this input vector [7]. The weight vectors associated to each output unit are known as codebook vectors. Each class of input space is represented by its own set of codebook vectors. Codebook vectors are defined according to the specific task. For the categorization task we use one codebook vector per class. The category label for each codebook vector is the class name [4].

The LVQ algorithm is a competitive network, and thus, for each training vector, output units compete among themselves in order to find the winner according to some metric. The LVQ algorithm uses the Euclidean distance to find the winner unit. Only the winner unit (i.e., the output unit with the smallest Euclidean distance with regard to the input vector) will modify its weights using the LVQ learning rule. The basic LVQ algorithm is the following [4, 32]:

1. Initialize the codebook vectors *Wi* and the learning rate α
2. Randomly select an input vector *X*

3. Find the winner unit closest to the input vector (i.e., the codebook vector *Wc* with the smallest Euclidean distance with regard to the input vector *X*):

$$||X - W_c|| = \min_k ||X - W_k|| \quad (1)$$

i.e.,

$$c = \arg\min_k ||X - W_k|| \quad (2)$$

4. Modify the weights of the winner unit:
   - If *Wc* and *X* belong to the same class (the classification has been correct)

$$Wc(t+1) = Wc(t) + \alpha(t)[X(t) - Wc(t)] \quad (3)$$

   - If *Wc* and *X* belong to different classes (the classification has not been correct)

$$Wc(t+1) = Wc(t) - \alpha(t)[X(t) - Wc(t)] \quad (4)$$

5. Reduce the learning rate α
6. Repeat from step 2 until the neural network is stabilized or until a fixed number of iterations have been carried out.

The learning rate $\alpha(t)$ $(0 < \alpha(t) < 1)$ is a monotically decreasing function of time which controls how quickly the weight vector is allowed to change. It is recommended that $\alpha(t)$ should already initially be rather small, say, smaller than 0.3 and it continues decreasing to a given threshold very close to 0.

2.6.3 Advantages of LVQ Algorithms

(1) They are easy to be implemented for multi-class classification problems.
(2) The algorithm complexity can be adjusted during training as needed.
(3) LVQs have less computationally expensive.
(4) Ability to correctly classify results even where classes are similar.
(5) The model is trained significantly faster than other neural network techniques like Back Propagation.
(6) It is able to summaries or reduce large datasets to a smaller number of codebook vectors suitable for classification or visualization.
(7) Able to generalize features in the dataset providing a level of robustness.
(8) Can approximate just about any classification problem as long as the attributes can be compared using a meaningful distance measure
(9) Not limited in the number of dimensions in the codebook vectors like nearest neighbor techniques.

(10) Normalization of input data is not required (normalized may improve accuracy if attribute values vary greatly).

(11) Can handle data with missing values.

(12) The generated model can be updated incrementally.

[7, 28, 31]

2.6.4 LVQ Versions

There are number of versions of LVQ algorithm [6]. This version based on the following basic algorithms:

- A learning sample consisting of input vector x_i together with its correct class label c_i is present to the network.
- A suitable number of codebook vectors are selected for every class label c_i.
- Using distance measures between codebook vectors and input vector d_i, the winner is determined. In some cases, the second best winner is also determined.

The version of LVQ algorithms described as follow:

- **Type One Learning Vector Quantization (LVQ1)** was the original of learning vector quantization algorithm developed by Kohonen in 1989 as a classification method using vector quantization architecture in combination with labeled vectors and supervised training based on a system of rewards and punishments [28]. The algorithm is designed to approximate the optimal Bayesian decision boundary by approximating a function derived from the conditional probabilities of the differently labeled data vectors.

- **Optimized LVQ1 (OLVQ1)**: The Optimized LVQ1 is essentially the same as LVQ1 except that there are independent learning rates for different codebook vectors [20,22,23]. The following discrete time learning process is obtained. If c is defined by $c=\text{argmin}_i \{\| x - m_i \|\}$. Then

$$M_c(t+1) = M_c(t) + \alpha_c(t) [X(t) – M_c(t)] \quad (7)$$

if x is classified correctly,

$$M_c(t+1) = M_c(t) + \alpha_c(t) [X(t) – M_c(t)] \quad (8)$$

if x is classified incorrectly,

$$M_i(t+1)=M_i(t) \qquad \text{for } i \neq c \quad (9)$$

Next, the problem of whether the $\alpha_i(t)$ can be determined optimally for fastest possible convergence of (3) was addressed. If equation (1) is expressed in form of

$$M_c(t+1) = [1-s(t)\,\alpha_c(t)]\,m_c(t) + s(t)\,\alpha_c(t)\,X(t) \qquad (10)$$

where:

s(t)= +1 if the classification is correct and

s(t)= -1 if the classification is incorrect,

it is noted here that $m_c(t)$ is statistically independent of X(t). It might also be noted how optimal the statistical accuracy of the learned codebook vector values, if the effects of the corrections made at different times are of equal weight. Notice that $M_c(t + 1)$ contains a "trace" from X(t) through the last term in Eq. (2), and "traces" from the earlier X(t'), where t' = 1, 2, . . . , t-1 through $M_c(t)$. The (absolute) magnitude of the last "trace" from X(t) is scaled down by the factor $\alpha_c(t)$, and, for instance, the "trace" from X(t-1) is scaled down by: $[1-s(t)\,\alpha_c(t)] \bullet \alpha_c(t-1)$.

It was made certain that these two scaling factors are identical

$$\alpha_c(t) = [1-s(t)\,\alpha_c(t)]\,\alpha_c(t-1) \qquad (11)$$

Now if the condition is fulfilled to hold for all t, it can be shown that the "traces" accumulated up to time t from all the earlier x will be scaled down by an equal amount at the end, and thus the "optimal" values of $\alpha_i(t)$ are determined by the recursion:

$$\alpha i(t) = \frac{\alpha_i(t-1)}{1+s(t)\alpha_i(t-1)} \qquad (12)$$

we noticed that precaution must be made when choosing $\alpha_c(t)$ so that it does not rise above the value 1, the interesting thing about learning algorithm OLVQ1 that it never allows any α_i to rise above its initial value. It is proved that the initial values of the α_i can be selected rather high, say, 0.3, whereby learning is considerably speeded up, especially in the beginning, and the m_i quickly finds their approximate asymptotic values.

- **LVQ2:** This algorithm has already been used by many researchers [33]. While in the LVQl only one codebook vector was updated at a time, this algorithm updates two vectors at each step, namely, the "winner" and the "runner-up". The purpose is to shift the midplane of these two vectors directly to the zone where the Bayes border is supposed to lie. An algorithm that can easily be seen to work in that direction is the following. First define a "window" show Figure 2.5 around the midplane of neighboring codebook vectors M_i and M_j.

- **LVQ2.1**: is the second improvement of LVQ2. The LVQ2.1 is assume that two codebook vectors M_i and M_j that belong to different classes and are closest neighbors in the vector space are initially in a wrong position. The (incorrect) discrimination surface, however, is always defined as the mid-point of M_i and M_j. Let us define a symmetric windows of nonzero width around the mid-point, and stipulate that corrections to M_i and M_j shall only be made if X falls into the window on the wrong side of the mid-point Figure 2.5.

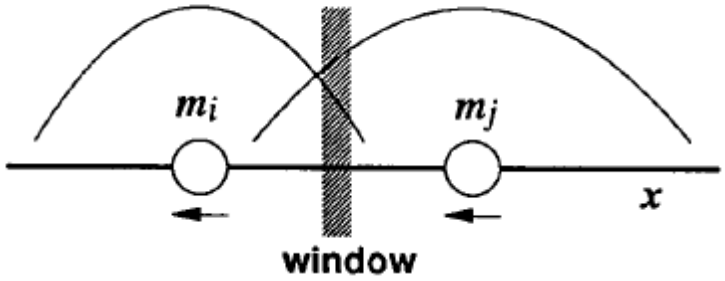

Figure 2.5: Illustration of the "window" used in LVQ2, The curve represent class distribution of x samples [8]

- **Type three LVQ3**: Essentially the same as LVQ2.1 but updating takes place only if both the vectors belong to the desired class. With LVQ2.1 the codebook vector are not guaranteed to continue approximating the class distributions and the algorithm does not continue over time to approximate the optimal Bayesian boundaries. LVQ3 addresses this fact by adding an update to LVQ2.1 algorithm in the case where the two reference vectors lie within the window centered around the mid-point between m_i and m_j whose width is defined by w, but where they both belong to the same class as the data vector [28].
- **Optimized LVQ3 (OLVQ3)**: is same as LVQ3 except each codebook vector has its own learning rate. The algorithms Allow the two closest vectors to learn if the following condition is met:

$$\min\left[\frac{d_{c1}}{d_{c2}}, \frac{d_{c2}}{d_{c1}}\right] > (1-\varepsilon)(1+\varepsilon) \tag{13}$$

where a typical value for ε is 0.2.

Summary

This chapter gave the basic theoretical foundation about the topic of data mining and the LVQ algorithms which will be used in our research. Next chapter will discuss the related work done in Arabic classification.

CHAPTER 3: Related Works

Many researchers have been working on text classification for English and other European languages. However, researches on text classification for Arabic language are limited [18, 25]. In this chapter we introduce state of the arts about Arabic text classification.

Arabic text classification applied by researchers using several traditional machine learning methods. Some others proposed new methods for classification. Artificial Neural Network (ANN) algorithms are used in many researches and compared its results with others traditional algorithms. In the next sections, we will introduce Arabic text classification by using traditional algorithms and we will focus on using ANN algorithms since it is our main concerns in this research.

3.1 Arabic Text Classification Using Traditional Algorithms

Al-Harbi et. al. in [3] experiment is to evaluate the performance of two popular classification algorithms (SVM and C5.0) on classifying Arabic text using the seven Arabic corpora. The SVM average accuracy is 68.65%, while the average accuracy for the C5.0 is 78.42%. The dataset comprised 17,658 texts with more than 11,500,000 words. It is obvious that their work accuracy is too low.

Naive Bayes used by EL Kordi et. al. in [20]. They applied Arabic text classification. The used data set includes 300 web documents for each of five categories from the Aljazeera website. The five categories used for this work were: sports, business, culture and art, science, and health. The average accuracy over all categories is: 68.78%. The higher performance for the Sports and the Business categories with a classification accuracy is higher than 70%. The performance of other categories ranges from 40% to 60%. Also, their performance is low.

Maximum entropy model used by El-Halees in [10] for Arabic text classification. The experiments trained the system using Arabic documents collected from the Internet which collected from Aljazeera Arabic news channel. The documents categorized into sex domains: politics, sports, culture and arts, science and technology, economy and health. The best classification accuracy was 80.40%. Although, the accuracy was better than the previous ones, but still it could be better.

Sawaf et. al. in [34] applied document classification and cluster on the Arabic NEWSWIRE corpus using statistical methods. The domains in the articles covered politics, economy, culture and sports. The best classification accuracy was 62.7%.

El-Halees in [35], and Al-Zoghby in [36] applied Arabic text classification using association rules. The accuracy achieved by El-Halees was 80.4%. Al-Zoghby used CHARM algorithm and showed the excellence of soft-matching over hard big O exact matching.

Mesleh in [37] applied Support Vector Machines (SVMs) based text classification system for Arabic language articles. The classifier used CHI square method as a feature selection method in the pre-processing step of the Text Classification system design procedure. They have used an in-house collected corpus from online Arabic newspaper archives, including Al-Jazeera, Al-Nahar, Al-hayat, Al-Ahram, and Al-Dostor as well as a few other specialized websites. The collected corpus contained 1445 documents that varied in length. These documents fell into nine classification categories. The result showed that high classification effectiveness for Arabic data set in term of F-measure achieved 88.11%.

Harrag et. al. in [38] classifying Arabic text documents using a decision tree algorithm. Experiments were performed over two self collected data corpus and the results showed that the suggested hybrid approach of Document Frequency Thresholding using an embedded information gain criterion of the decision tree algorithm was the preferable feature selection criterion. The study concluded that the effectiveness of the improved classifier was very good and gave generalization and they also concluded that the effectiveness of the decision tree classifier was increased as they increased the training size, and the nature of the corpus had such a influence on the classifier performance.

Khreisat in [11] applied classification for Arabic text documents using the N-gram frequency statistics. The technique employs a dissimilarity measure called the "Manhattan distance", and Dice's measure of similarity, for the purposes of classification. The Dice measure was used for comparison purposes. Results showed that N-gram text classification using the Dice measure gave better classification results compared to the Manhattan measure. A corpus of Arabic text documents was collected

from online Arabic newspapers. The corpus consisted of text documents covering 4 categories: sports, economy, technology and weather. The best result for the tri-gram method using the Manhattan measure was achieved for the sports category with a recall value of 88% and the worst result was for the economy category with a recall value of 40%.

Habib et. al**.** in [39] hybrid approach of global and local feature selection technique was proposed and compared with both local and global feature selection techniques. Results reported on a set of 1132 documents of six different topics showed that the proposed hybrid feature selection surpassed the local and global feature selection in selecting effective terms and improves the Arabic documents categorization efficiency. Experiments performed on their data set showed that the proposed hybrid feature selection gave a high classification rate equivalent to that of global feature selection besides reducing the number of empty documents.

3.2 Arabic Text Classification Using ANN Algorithms

Harrag and El-Qawasmah in [25] applied Back-propagation ANN algorithm for Arabic text classification. The experimental results showed that ANN model using Singular SVD achieves (88.33%) which is better than the performance of basic ANN which yields 85.75% on Arabic document classification. The dataset collected from *Hadith encyclopedia* from nine books which includes 453 documents distributed over 14 categories.

Harrag et. al**.** in [23] presented a model based on the Neural Network (NN) for classifying Arabic texts. they proposed using Singular Value Decomposition (SVD) as a preprocessor of NN with the aim of further reducing data in terms of both size and dimensionality. The processing via the NN method involved three basic steps, namely data pre-processing، training and testing. In their case, the Feature Extraction phase referred to data pre-processing using SVD method. For the training dataset, once we obtain the selected features, they feed them into the neural network and generated a text classifier. For each test text, they used the classifier to verify the efficiency of NN model. They used a specific corpus of Prophetic Traditions or "Hadiths' (sayings and doings of the Prophet 'Peace Be Upon Him') collected from the Prophetic Traditions Encyclopedia Alkutub Altissâa - "The Nine Books". The domain included 453 documents distributed over 14 categories. The dataset had 5743 tokens from 14 major

categories. The number of words collected from all the categories after the preprocessing step is 1065 word. The average F1 measure value improved with higher number of input factors. The average F1 with all features (1065) was 49% for MLP and 38% for RBF. The average F1 with SVD (530) is between 22% and 53% for MLP and between 15% and 30% for RBF.

3.3 Text Classification for Different Languages Using LVQ Algorithm

Umer and Khiyal [40] evaluates the Learning Vector Quantization (LVQ) network for classifying text documents. In the LVQ method, each class is described by a relatively small number of codebook vectors. These codebook vectors are placed in the feature space such that the decision boundaries are approximated by the nearest neighbor rule. The experimental results show that the Learning Vector Quantization approach outperforms the k-NN, Rocchio, NB and Decision Tree classifiers and is comparable to SVMs. The Reuters-21578 data set used for evaluation and used tf-idf term weighting scheme. The results presents that the accuracy for five versions of LVQ algorithm achieved similar accuracies also take similar time for building the training model but the OLVQ1 performs well than its counterparts. It gives F1-measure over 90% for weat and trade category.

Pilevar et. al. [7] presented classification of Persian documents by using the Learning Vector Quantization network. They used 1050 news stories are chosen from Hamshahri2 text collection as training data set. Hamshahri2 corpus is a Persian test collection that consists of 345 MB of news texts from this newspaper since 1996 to 2002 (corpus size with tags is 564 MB). This corpus contains more than 160,000 news articles about variety of subjects (82 categories like politics, literature, art and economy and includes nearly 417,000 different words. Hamshahri2 articles vary between 1 KB and 140 KB in size. The categories are however overlapping and non-exhaustive, and there are relationships among the categories. The experimentation of LVQ present that F1-measure results are much higher than KNN's results (i.e. compare 0.868 of LVQ with 0.686 of KNN). According to our experiments, the LVQ performs better than the SVM (average F1-measure of 0.757) which is believed to be the best classifier around.

Summary

In this chapter we gave an overview about works done in Arabic classification. We can notice that the accuracy of the classification is not good as in English. That means it could be a better accuracy which we expected by using LVQ algorithms. Also, LVQ has the advantage of less training time which is important in this area. Training time did not recorded by any of the previous works.

Next chapter will give the methodology and the experiment settings of using LVQ in Arabic classification, while chapter 5 will give and discuss the results of the experiments.

CHAPTER 4: Methodology

This chapter explains the methodology deployed in this study and at the research methods which informed my choice of methods.

4.1 Methodology Steps

The methodology used in this thesis simplified in the following steps also show Figure 4.1.

1. Collect Arabic text document from different domains.
2. Using neural network algorithm LVQ (LVQ 1) and another LVQ algorithms such as LVQ2.1, LVQ3, OLVQ1 and OLVQ3 and see if there is any improvement.
3. Perform pre-processing to convert documents from unstructured data to structure data. Evaluate the method using Accuracy, recall , precision , f-measure and training time.
4. Compare our results with results comes from other methods such as K-NN, naïve bays and decision trees.

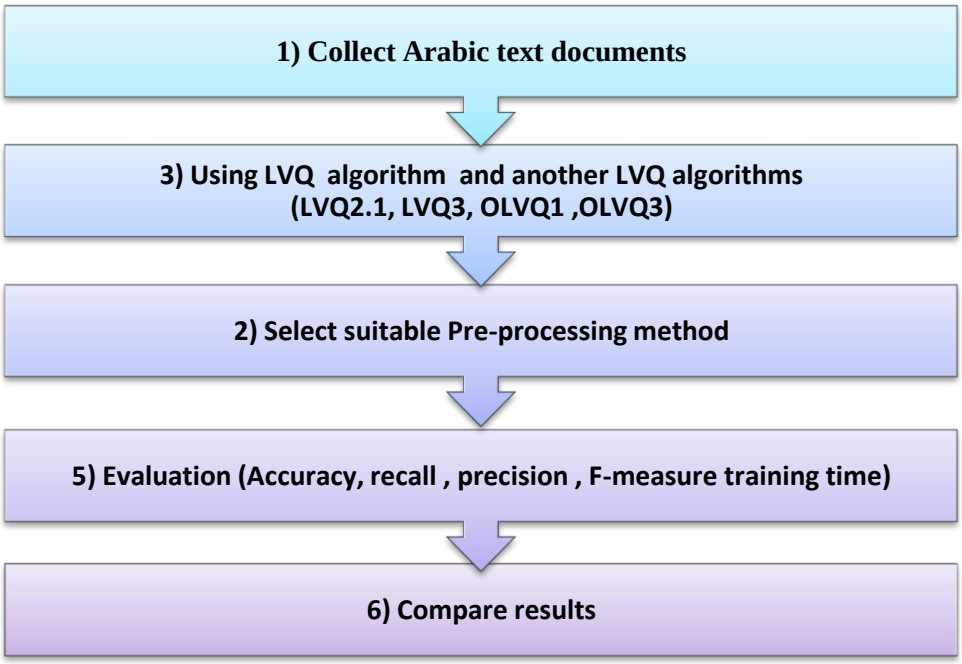

Figure 4.1: Methodology Steps

The next sections will discuss the steps, excepts 4 and 6 which they already described in chapter 3.

4.2 Arabic Text Preprocessing

Text mining process required several steps to structuring text data that depicted in Figure 4.2. The process starts by tokenizing string to words, after that word normalizing tokenized words, the stop word removed and applying stemming algorithm (stemming / light stemming / raw text stemming), and finally term weighting for each words.

Arabic language is a complex and more rich morphology than other languages like English. Holy Quran adds great value to Arabic which is widely studied and known throughout the Islamic world.

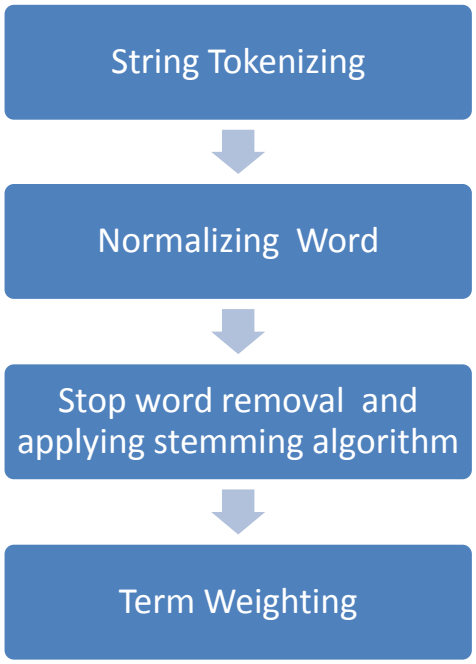

Figure: 4.2 Text preprocess structure

In the next section we will describe text preprocess structure steps and concentrate on Arabic text especially.

4.2.1 String Tokenize

Tokenization is a necessary and non-trivial step in natural language processing [41]. Token is one of the first steps of processing any text corpora is to divide the input text into proper units. These units could be characters, words, numbers, sentences or any other appropriate unit [42]. The definition of a word here is not the exact syntactic form, that is why we call it a 'token'. In the case of Arabic, where a single word can comprise up to four independent tokens, morphological knowledge

needs to be incorporated into the tokenize. However, Tokenization closely related to the morphological analysis [41]. The tokenize process is responsible for defining word boundaries such as white spaces and punctuation marks, demarcating words or (clitics), multiword expressions, abbreviations and numbers.

One of the most useful features in detecting sentences boundaries and tokens is punctuation marks. However, the total number of punctuation marks and symbols used in Arabic corpus was 134, while in the corresponding English corpus only 54 punctuations and symbols were used [42]. There are several methods to apply tokenization, the simplest way we used is extracting any alphanumeric string between two white spaces.

4.2.2 Stop Words

Stop words are frequently occurring, insignificant words that appear in an article or web page (i.e. pronouns, prepositions, conjunctions, etc.) [10, 20, 23, 43]. In Arabic words like (هذه ، انه ،تكون،قد ،بين،لن،نحو،في ، على ، أين ، كان) are considered stop words. These words carry no information. Stop words are filtered out prior to processing of natural language data. Khoja list stop words which consists of 165 word [12, 39].

4.2.3 Stemming Algorithms

Stemming is a root-finding process that consists in removing the affixes from the word [23]. Stemming is a method of word standardization used to match some morphologically related words [39]. It is essential to improve performance in information retrieval tasks especially with highly inflected language like Arabic language. The stemming process reduces the size of the documents representations by 20-50% compared to full words representations [44] which also assist to improve the retrieval for documents.

There are four kinds of affixes: antefixes, prefixes, suffixes and postfixes that can be attached to words. Thus an Arabic word can have a more complicated form if all these affixes are attached to its root. The following Table 4.1 shows an example of word (ليفاوضونهم) with all types of affix [45]:

Table 4.1: An agglutinated form of an Arabic word meaning "to negotiate with them" [45]

Antefix	Prefix	Core	Suffix	Postfix
ل	يـ	فاوض	ون	هم
Preposition meaning "to"	A letter meaning the tense and the person of conjugation	negotiate	Termination of conjugation	A pronoun meaning "them"

Many stemmers have been developed for English and other European languages. These stemmers mostly deal with the removal of suffixes as this is sufficient for most information retrieval purposes. Some of the most widely known stemmers for English are Lovins and Porter stemming algorithms [46].

The cause for needing special stemming algorithms for Arabic language can be described by El-Sadany and Hashish in the following points [44]:

i. Arabic is one of Semitic languages which differs in structure of affixes from Indo-European type of languages such as English and French;

ii. Arabic is mainly roots and templates dependent in the formation of words;

iii. Arabic roots consonants might be changed or deleted during the morphological process.

Arabic language There are three different approaches for stemming: the root-based stemmer; the light stemmer; and the statistical stemmer.

4.2.3.1 Root-based Stemmer

This type of stemmer used morphological analysis to extract the root of a given Arabic word. Several algorithms have been developed for this approach.

- **A Rule and Template Based Stemming Algorithm for Arabic Language**

This algorithm based on morphological rules has been developed, and to enhance its effectiveness, a dictionary of root words is used to determine the right stems. The author's used stemming algorithm which consisted of the following main modules [44]:

- Prefix and suffix removal module.
- Root generator and checking module.
- Pattern generator and checking module.

- Intensification module which handles double letters (تشديد).
- Recoding module.

- **Al-Fedaghi and Al-Anzi Stemming Algorithm**

The author's present an algorithm to generate the root and the pattern of a given Arabic word. The algorithms based on using two files, the first file for trilateral roots, and a second file for all patterns with all possible affixes attached to them. The algorithm does not remove any prefixes or suffixes; it just checks each word against all possible patterns with the same number of letters. If the word and the pattern match, it extracts the root which is comprised of the three letters which are in the positions of the letters (ف ، ع ، ل). If this root is found in the file of roots, then the root and pattern are returned as output. This stemming algorithm takes into consideration that some letters may be deleted or modified during the derivation of words from their roots, and deals with these situations accordingly. The algorithm was tested on various texts, and the percentage of reduced words (extracted roots) ranged from 50 to 80% [47].

- **Khoja Stemming Algorithm**

Shereen Khoja's developed stemmer algorithms [48]. The algorithm, developed by using both Java and C++ languages, removes the longest suffix and the longest prefix. It then matches the remaining word with verbal and noun patterns, to extract the root. The stemmer makes use of several linguistic data files such as a list of all diacritic characters, punctuation characters, definite articles, and 168 stop words. The algorithm achieves accuracy rates of up to 96%. The algorithm correctly stems most Arabic words that are derived from roots.

the Khoja stemmer has several weaknesses. Firstly, the root dictionary requires maintenance to guarantee newly discovered words are correctly stemmed. Secondly, the Khoja stemmer replaces a weak letter with (و) which occasionally produces a root that is not related to the original word. For example, the word (منظمات) (organizations) is stemmed to (ظمأ) (thirsty) instead of (نظم). Here the Khoja stemmer removed part of the root when it removed the prefix and then added (أ) at the end.

4.2.3.2 Light Stemmer

The light stemming approach refers to a process of stripping off a small set of prefixes and/or suffixes, without trying to deal with infixes, or recognize patterns and

find roots [49]. Light stemmer does not need a dictionary like the Root-Based stemmer because it only removes list of predefined affixes from the word without checking if the remained stem is an entry for a dictionary of words. There is no standard algorithm for Arabic light stemming, all trials in this field are a set of rules to strip off a small set of suffixes and prefixes, also there is no definite list of these strippable affixes.

The simple methods used to retrieve stemming for Arabic text word contains two steps to removes affixes: Firstly, remove frequent prefixes from the beginning of words. These prefixes (وبال ،فل ،ولل ،كال ،فال ،ول ،وب ،بال ،للل ،ب ،ال ، ا ،و، فب) are generally prepositions or sometimes several prepositions attached to the words. Secondly, remove suffixes that we judged necessary to truncate from words are those which are the most frequent and represent generally pronouns expressing number or gender of Arabic nouns: (ا ،ه ،ن ،ي ،ت ،ها ،ين ،و ،ان ،ات ،ون ،هم ،نا ،آ ،وا ،هما ،تي). Now we present several proposed algorithms for light stemming:

- Leah Larkey and Margaret Connell [49] developed a light stemming algorithm and combined it with Root-Based stemmer for Khoja. The algorithm was tested for information retrieval tasks of both mono and bilingual. It proved that the combined algorithm surpass the light and the Root-Based stemmers.

- Kareem Darwish [50] presents a rapid method for developing a shallow Arabic morphological analyzer. The analyzer will only be concerned with generating the possible roots of any given Arabic word. The proposed method does not require rules and prefix and suffix lists are constructed manually. Instead, the system replaces the manual processing with automatic processing. From evaluation, the amount of time required to build the rules is reduced to hours rather than days or weeks.

- Hayder Al-Ameed and et. al. [51] proposed an enhanced of a TREC-2002Arabic light stemmer presented by Kareem Darwish. Five proposed stemming algorithms are resulted in significantly better Arabic stemming outcomes in comparison with the TREC-2002 algorithm. The proposed approaches provided better accepted (meaningful) outcomes of Arabic words with up to 30-50% more than TREC stemmer outcomes.

4.2.3.3 Statistical Stemmer

Statistical stemmers attempted to group word variants using clustering techniques. Related words can be grouped based on various string-similarity measures.

One of these algorithms is n-gram which used for many different languages, one would not expect them to perform well on infixing languages like Arabic [46, 52]. The goal of n-gram language model is to determine the probability of a word sequence. In n-gram language models, we condition the probability of a word on the identity of the last (n-1) words [52].

However, Mayfield et al. have developed a system that combines word-based and 6-gram based retrieval, which was performed well for many languages including Arabic [46, 53].

Massimo Melucci and Nicola Orio built HMM algorithm [54] for statistical stemming. Proposed algorithm based on Hidden Markov Models (HMM). The algorithms use a list of words as training set, the method estimates the HMM parameters which are used to calculate the most probable stem for an arbitrary word. Stemming is performed by computing the most probable path, through the HMM states, corresponding to the input word. Linguistic knowledge or a training set of manually stemmed words are not required.

4.3 Term Weighting

Term weighting is one of pre-processing methods used for enhanced text document presentation as feature vector. Term weighting helps us to locate important terms in a document collection for ranking purposes [55]. There are several term weighting schemes the popular term weighting schemes are Boolean model, Term Frequency (TF), Inverse Document Frequency (IDF), and Term Frequency-Inverse Document Frequency (TF-IDF) [9]. Choosing an appropriate term weighting scheme is more important for text categorization [56]. We introduce several details in the next sections for different term weighting schemes.

4.3.1 Boolean model

The Boolean model is the simplest retrieval model based on Boolean algebra and set theory [57]. Boolean model indicates to absence or presence of a word with Booleans 0 or 1 respectively [9].

4.3.2 Term Frequency (TF)

Term frequency *TF(t,d)* is the number that the term *t* occurs in the document d [9]. The TF measures of the importance of term t_i within the particular document d_j can be calculated by equation:

$$Tf_{i,j} = \frac{n_{i,j}}{\sum n_{k,j}} \qquad (1)$$

Where:

$n_{i,j}$: the number of occurrences of the considered term (t_i) in the document d_j.

$\sum n_{k,j}$: Sum of number of occurrences of all terms in document d_j.

4.3.3 Inverse Document Frequency (IDF)

The inverse document frequency (IDF) is one of the most widely used term weighting schemes for estimating the specificity of term in a document collection [58]. It is based on the idea that if a term appears in only a few documents in the collection, then such a term is expected to be a good discriminator of these documents. The IDF weight of a term t can be calculated from document frequency using the formula:

$$IDF_t = log(N/n) \qquad (2)$$

Where:

N: number of documents.

n: number of documents with word i

The IDF of a term is low if it occurs in many documents and high if the term occurs in only a few documents [9].

4.3.4 Term Frequency-Inverse Document Frequency (TF-IDF)

Term Frequency and Inverse Document Frequency (TF-IDF), is a popular method of pre-processing documents in the information retrieval community [56]. TF-IDF works by determining the relative frequency of words in a specific document compared to the inverse proportion of that word over the entire document corpus. Intuitively, this calculation determines how relevant is a given word in a particular document [59]. The TF-IDF calculated by using the formula [19]:

$$w_{ij} = tfidf\ (t_i, d_j) = \frac{f_{ij}}{\sqrt{\sum_{k=1}^{M} f_{kj}^2}} \times \log\left(\frac{N}{n_i}\right) \quad (3)$$

Where:

N: is the number of documents in the data set,

M*:* is the number of terms used in the feature space,

f_{ij}: is the frequency of a term i in document j, and

n_i: denotes the number of documents that term *i* occurs in at least once.

Using this formula, long documents will have a higher length normalization value and thus reducing the *tf* $\times$*idf* value for the terms in that document, this makes them comparable with those in documents of shorter lengths [19]. There are many variations of the *TF-IDF* formula. All are based on the idea that the term-weighting should reflect the relative importance of a term in a document as well as how important the term is in other documents.

4.4 Text Mining Tools

Among well known open source data mining tools offering text mining functionality is the WEKA (Waikato Environment for Knowledge Analysis) suite a collection of machine learning algorithms for data mining and text mining [60]. WEKA includes the following features [61]:

• *Data preprocessing*: Data can be filtered by a large number of methods (over 75), ranging from removing particular attributes to advanced operations such as principal component analysis.

• *Classification*: One of WEKA's drawing cards is more than 100 classification methods it contains. Classifiers are divided into "Bayesian" methods (Naive Bayes, Bayesian nets, etc.), lazy methods (nearest neighbor and variants), rule-based methods (decision tables, OneR, RIPPER), tree learners (C4.5, Naive Bayes trees, M5), function-based learners (linear regression, SVMs, Gaussian processes), and miscellaneous methods. Furthermore, WEKA includes meta-classifiers like bagging, boosting, stacking; multiple instance classifiers; and interfaces for classifiers implemented in Groovy and Jython.

• *Clustering*: Unsupervised learning is supported by several clustering schemes, including EM-based mixture models, *k*-means, and various hierarchical clustering algorithms. Though not as many methods are available as for classification, most of the classic algorithms are included.

• *Attribute selection*: The set of attributes used is essential for classification performance. Various selection criteria and search methods are available.

• *Data visualization*: Data can be inspected visually by plotting attribute values against the class, or against other attribute values.

WEKA also includes support for association rule mining, comparing classifiers, data set generation, facilities for annotated documentation generation for source code, distribution estimation, and data conversion.

In this thesis WEKA as text mining tools used in preprocessing and classification. Therefore, next section discuss preprocessing techniques used for Arabic text and used result for classification.

4.5 Text Preprocessing Tools

The WEKA provides *StringToWordVector* tool. This tool converts String attributes into a set of attributes representing word occurrence (depending on the tokenizer) information from the text contained in the strings. The set of words (attributes) is determined by the first batch filtered (typically training data). The Figure 4.3 and Figure 4.4 present *StringToWordVector* tools and options.

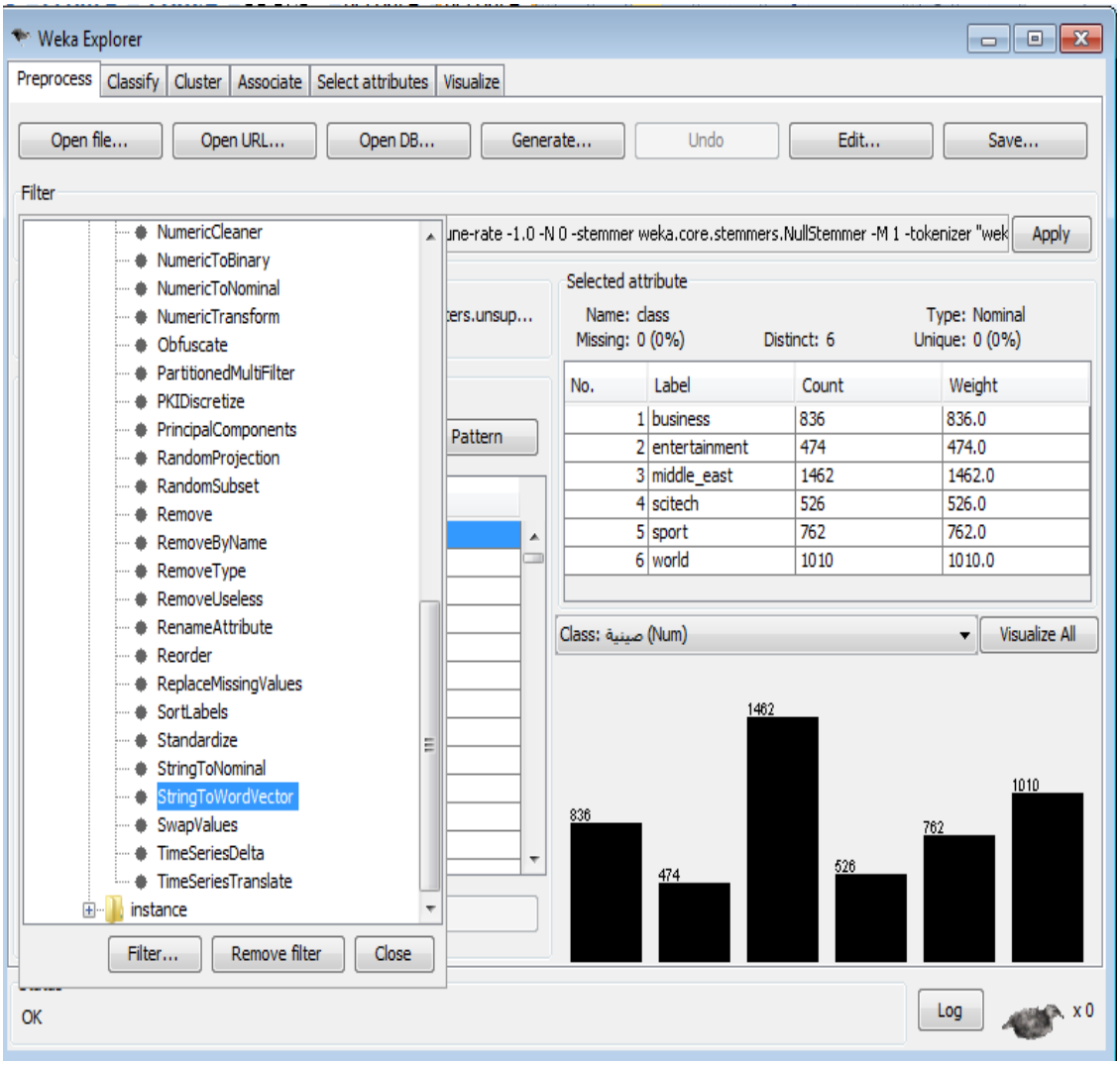

Figure 4.3: String To Word Vector tools

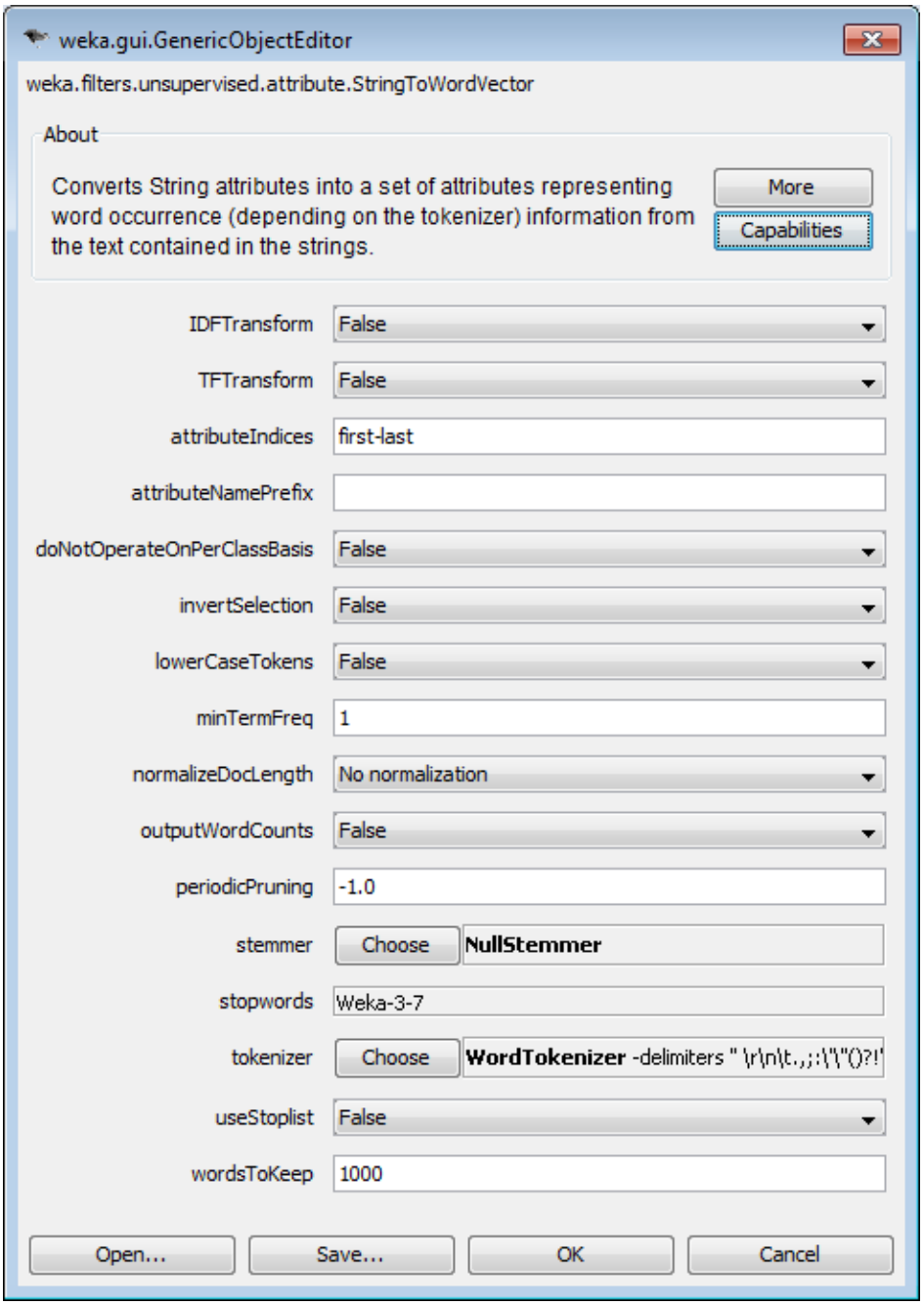

Figure 4.4: String To Word Vector options

We use term weighting in different combinations to structure text data. The Table 4.2 below presents different term weighting combinations. Major combination includes *Boolean, tf, idf, tf-idf, term pruning* and stemmer methods. In my research different combination applied to conclude which preprocessing achieved best results. We applied the following combinations: Boolean, TF, IDF, TF-IDF, TF-IDF-minFreq3, TF-IDF-minFreq5, TF-IDF-minFreq7 and TF-IDF-minFreq9 in all proposed classification experiments.

Table 4.2: Weka String to Word Vector options [9]

Boolean	*Indicating presence (1) or absence (0) of a word*
TF Transform	*Sets whether if the word frequencies in a document should be transformed into log(1+fij), where fij is the frequency of word i in document dj*
IDF Transform	*Sets whether if the word frequencies in a document should be transformed into fij*log(num of Docs/num of Docs with word i), where fij is the frequency of word i in document dj*
TFIDF Transform	*log(1+fij)* fij*log(num of Docs/num of Docs with word i) , where fij is the frequency of word i in document dj.*
minTermFreq	*Sets the minimum term frequency*

Also, other options like stemming used to select stemmer algorithm on the words. WEKA tools provided different algorithms that supported Arabic language show Figure 4.3. In my research, we firstly applied classification without stemmer, after that applied *Light* stemmer and finally, *Khoja* stemmer to compare which stemmer is appropriate for achieving high accuracy.

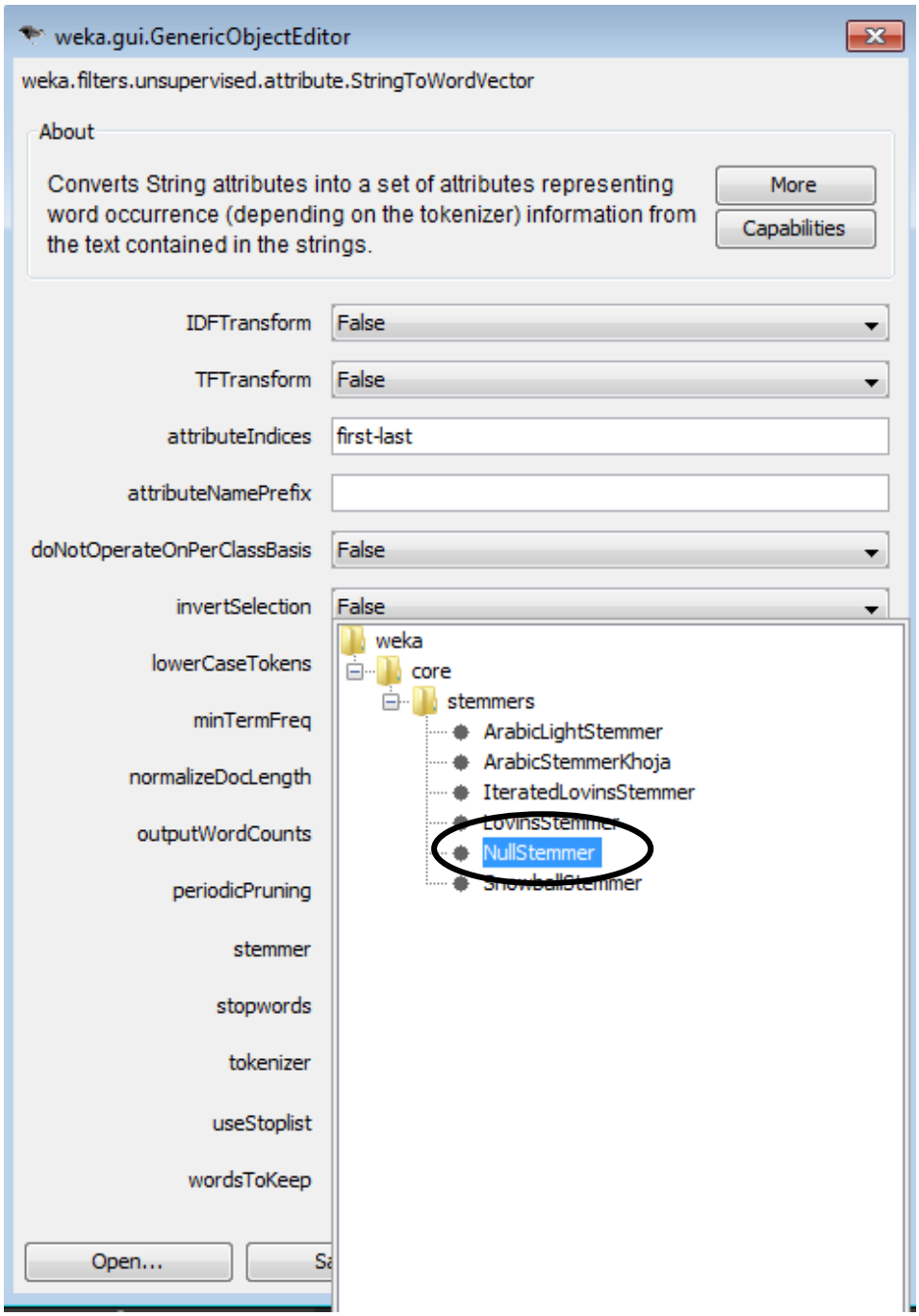

Figure 4.5: Weka Stemmers Algorithms

4.6 Corpora

One of difficulties for Arabic language is the lack of publicly available Arabic corpus for evaluating text categorization algorithms [19]. In other side, English language has different public data set for English text classification. Reuter's collection of news stories are popular and typical example.

Researchers in Arabic text classification used own data set collected from several Arabic website like Al-Jazeera, Al-Nahar, Al-hayat, Al-Ahram, and Al-Dostor. The collected data has different size and different categories used for training and testing. In other hand, the Linguistic Data Consortium (LDC) provides two non-free Arabic corpora, the Arabic NEWSWIRE and Arabic Gigaword corpus.

From previous, Arabic language needs a freely corpus and used as standard or benchmark corpora. Therefore, there are a lot of researches in Arabic text classification field, and some published experimental results, but these results need to be compared by using same data set, because the comparison is hard when using different data set and doesn't have the same standards and conditions to analysis results.

My research used for experimentation freely public data set published by *Saad* in [62] . The dataset collected from *CNN Arabic* website for several reasons, because its free and public, contains suitable number of documents for the classification process and also suitable to the hardware used in experiments to not cause an out of memory problem. CNN Arabic dataset used in experimentation has different domains. Table 4.3 presents domains of CNN-arabic corpus which includes 5070 documents. Each document belongs to 1 of the 6 domains or categories.

Table 4.3: Number of document per category

number	Category (Domain)	# of text document	% from corpus
1	Business	836	16.49%
2	Entertainments	474	9.35%
3	Middle East News	1462	28.84%
4	Science & Technology	526	10.37%
5	Sports	762	15.03%
6	World News	1010	19.92%
Total		**5070**	**100%**

The other dataset used called OSCA. OSCA dataset collected from multiple websites, the corpus includes 22,429 text documents. Each text document belongs 1 of to 10 categories show Table 4.4 .

Table 4.4: OSCA corpus

number	Category (Domain)	# of text document	% from corpus
1	Economic	3102	13.83%
2	History	3233	14.41%
3	Education and family	3608	16.09%
4	Religious and Fatwas	3171	14.14%
5	Sport	2419	10.79%
6	Health	2296	10.24%
7	Astronomy	557	2.48%
8	Low	944	4.21%
9	Stories	726	3.24%
10	Cooking Recipes	2373	10.58%
Total		**22,429**	**100%**

4.7 Evaluation Methods

In this section we will describe the use evaluation criteria for text classification, to give a possibility for comparing the methods and tasks with other works. There are different measures used to measure classification success. Therefore, different basic a measures are used: accuracy, precision, recall, F-measure and time . Accuracy as measure is the number of samples correctly classified. We used accuracy to compute the correction of classification produced from testing data by using LVQ neural network algorithms and time elapsed when applied LVQ algorithms.

The steps used to compute precision and recall [34] by computing confusion matrix show Figure 4.4 for two–class classification problem. A confusion matrix is computed by creating two categories, it is a matrix where test cases are distributed as follows:

1- **True positive (TP):** refers to positive instances that correctly labeled the classifier.
2- **False Negative (FN):** are the positive instances that were incorrectly labeled.
3- **False Positive (FP):** are the negative instances that were incorrectly labeled
4- **True negative (TN)**: refers to negative instances that correctly labeled the classifier

True Positive (TP)	False Negative (FN)
False Positive (FP)	True Negative (TN)

Figure 4.6: Confusion matrix, precision and recall

When classification problem is not binary , confusion matrix gets more complicated. We can compute classifier accuracy as:

$$Accuacy = \frac{\sum_{i=1}^{n} (True\ clasifcation)_i}{\text{Total number of cases}}$$

Where i is the class number and n is total number of the classes.

The common evaluation measures Precision and recall. They are used as generally accepted ways of measuring systems performance. The precision is the percentage of predicted document for the given topic that are correctly classified [10] as following equations:

$$Precision\ P = \frac{TP}{TP+FP} \qquad (5)$$

Also, we computed recall which is the percentage of the total documents for the given topic that are correctly classified as following equation:

$$Recall\ \mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \qquad (6)$$

The F1 measure combines precision and recall. We used the F1 measure to evaluate the performance of text classifiers. The following equations presents F-measure:

$$F1 - \text{measure} = \frac{2.\mathrm{R}.\mathrm{P}}{\mathrm{R}+\mathrm{P}} \qquad (7)$$

Time measurement means the time needed to build a model by using training split dataset this operation called training time breakdown. The unit for time measure calculated as second.

Summary

This chapter describes the methodology used in my research. It describes the preprocessing step, the used corpora and the evaluation criteria. The data mining step, which is LVQ method in my case, was described in chapter 3. It also, describes the text mining tools used in my experiments. The next chapter will be about the results of my experiments.

CHAPTER 5:
Experimental Results and Analysis

This chapter, describes the results and analysis of applying neural network LVQ's algorithms on the selected datasets. Also, we compare our results with other well known classification algorithms. For evaluation purpose, we split each dataset into two parts (70% of the corpus for training and remaining 30% for test). All classification experiment run on machine environment has 64-bit with 4GB RAM.

In the first section we selected which LVQ's algorithm is the best. In the second section we used best LVQ's in different preprocessing presentations. Section three examined the effectiveness of increasing frequency of the token. Section four, applied the algorithm in different domains. Section five compared the results of the LVQ algorithm with others neural classification algorithms. The last section compared LVQ with other well known classification algorithms.

5.1 Comparing Learning Vector Quantization algorithms (LVQ's)

In this section, different LVQ's algorithms have been experimented on CNN-dataset from [62] to determine which one of these algorithms: LVQ1, LVQ2.1, LVQ3, OLVQ1 and OLVQ3 can achieve preferred result. The parameter for five algorithms have been selected empirically by slightly increasing and decreasing their value and analysis the output. From the experiments, it's clear that LVQ2.1 version achieved the highest accuracy (93.03%) and high F-measure (93.00%) followed by OLVQ3 (85.86%), LVQ1 (85.27%), LVQ3 (84.09%). Finally, OLVQ1 has the worst results (76.53%) and lowest F-measure (68.80%). Figure 5.1. shows that the differences between highest accuracy about (7%).

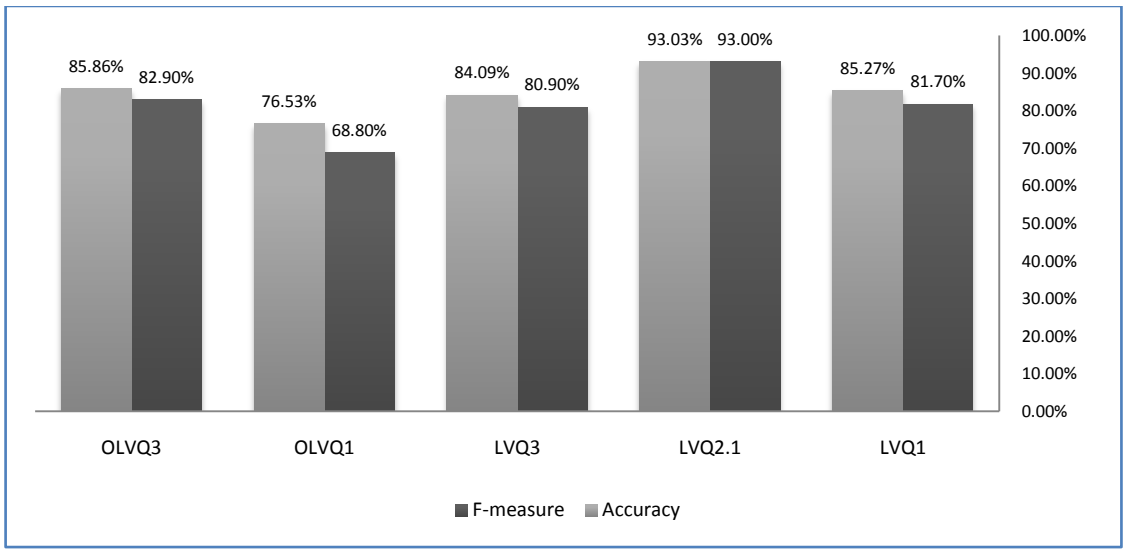

Figure 5.1: Accuracy and F-measure for LVQ's algorithms: CNN dataset

To Confirm our results we used another dataset which is OSCA dataset from [62]. As shown in Figure 5.2, the experiments applied to confirm that LVQ2.1 achieved high accuracy also when dataset size increases. The results present that the LVQ2.1 achieved highest accuracy (95.62%) and highest F-measure (93.80%) followed by OLVQ3 (93.68%), LVQ3 (92.73%), LVQ1 (91.63%). Finally, OLVQ1 has the results (90.07%).

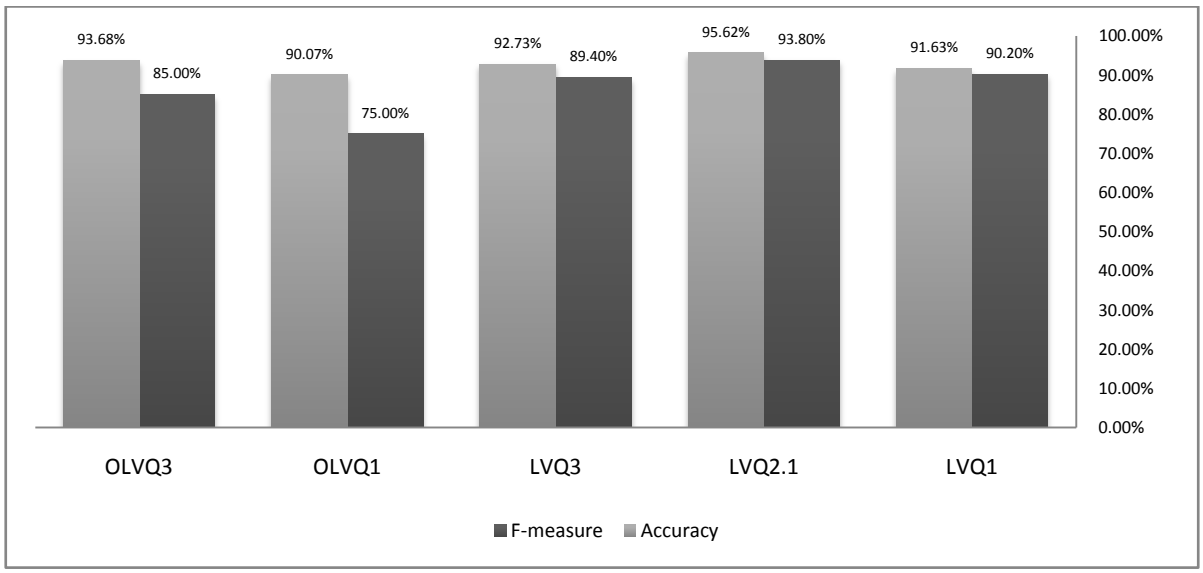

Figure 5.2: Accuracy and F-measure for LVQ's algorithms: OSCA dataset

5.2 Comparing stemming techniques for LVQ's algorithms

This section uses the best LVQ's algorithm (LVQ2.1) from previous section with different stemming to present which stemmer is best. Also, the data set used for experimentation is CNN dataset. Three stemming techniques have been used: Root stemming, light stemming and raw text (without stemming).

Stemming is the process of reducing inflected derived words to their stem or root form. The target is reducing dimension reduction of words to their stem. The stemming technique used khoja stemmer algorithm which reduce words to their stem regardless of different meaning of the same word. Light stemming, in contrast, removes common affixes from words without reducing them to their stems. Figure 5.3 shows the accuracy for different stemming techniques. The results present that accuracy for LVQ2.1 algorithm when using stemming method get highest accuracy (93.03%) because stemming removes all variants of words which have or have not similar

meaning. In other hand, using light stemming method gets accuracy (92.97%), but raw text gets close accuracy (92.04%) this closer because natural of document doesn't contains a lot of variants between words.

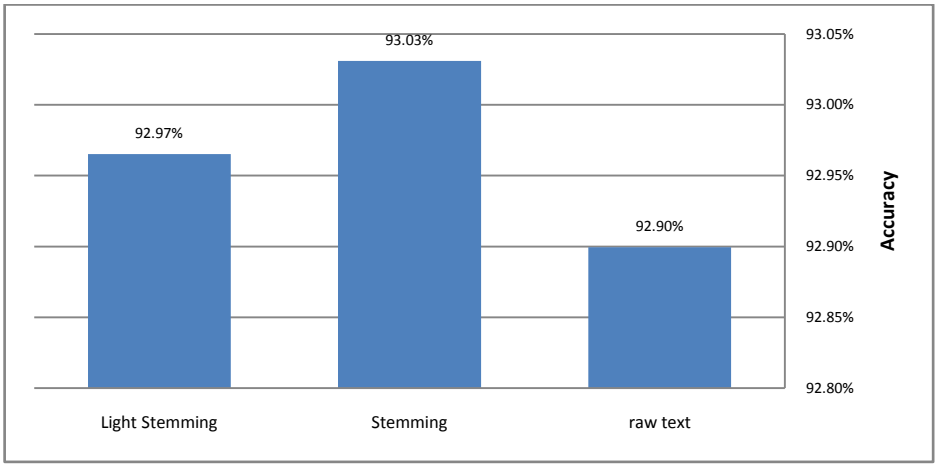

Figure 5.3: LVQ2.1 accuracy for different stemming techniques

Time is critical in neural network algorithms. Therefore, Figure 5.4 shows the time taken to build LVQ2.1 model when used different stemming techniques. Stemming needs less time to build model, but raw text needs four times the time needed to build stemming method.

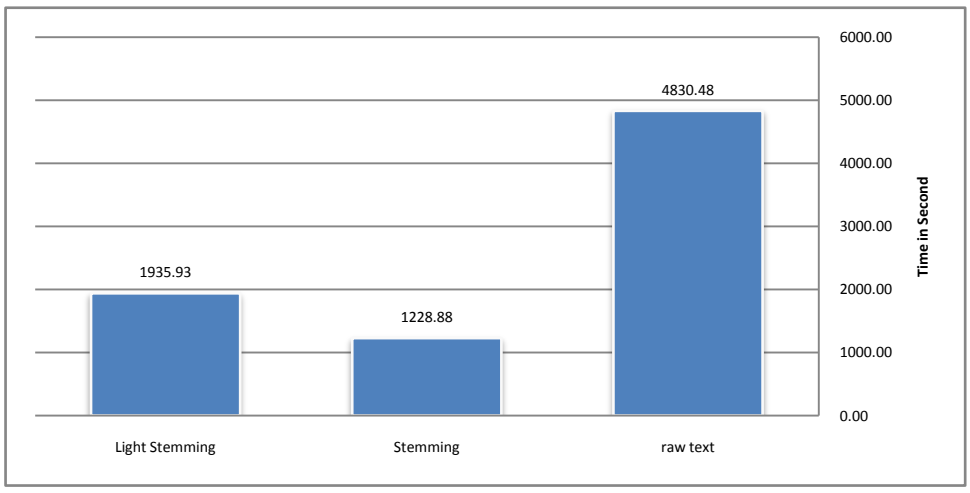

Figure 5.4: LVQ2.1 time for building model when using different stemming techniques

From previous results, high accuracy for neural network classification algorithm and reducing time to build model are important and desired. Therefore, stemming get high accuracy and takes less time to build model.

5.3 Comparing accuracy when change learning rate value

Learning rate value reduced in each iteration cycle. In some cases the test stopping condition when learning rate reaching a sufficiently small value. In this section we change value of learning rate in descending order for LVQ2.1 algorithm. Figure 5.9 present accuracy when using learning rate value (0.3, 0.2, 0.1, 0.05, 0.03 and 0.01). The experimentation present that the pest learning rate is (0.03) and achieved high accuracy (93.82%) and using (0.2) achieved less accuracy (90.72%). For that, changing value of learning rate impact accuracy.

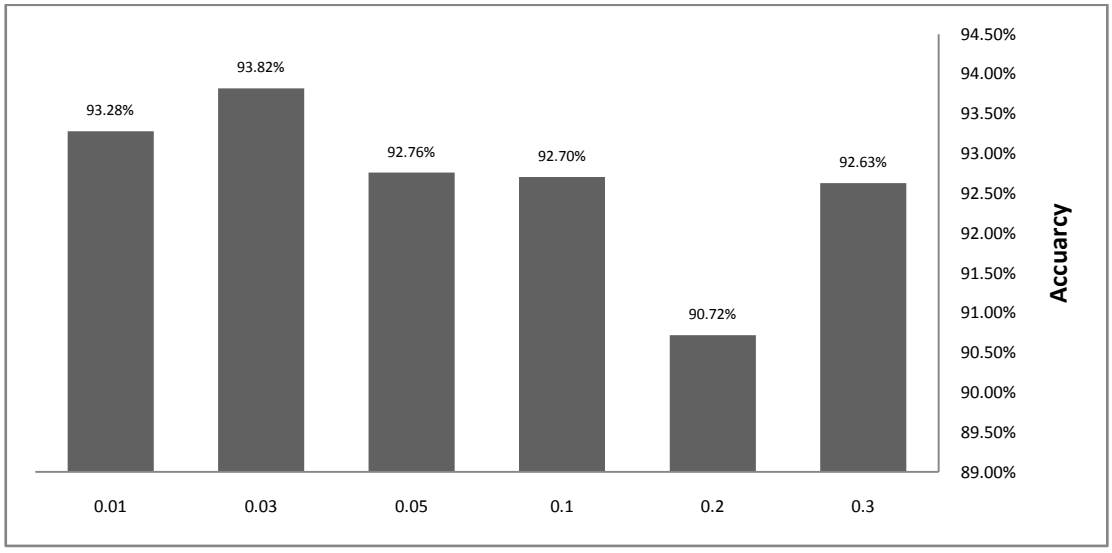

Figure: 5.5 Accuracy when change learning value for LVQ2.1 algorithm

From this section, we can conclude that the best setting is using light stemming and term frequency Freq.=5 and learning rate value (0.03) which achieved high accuracy (93.82%) and less training time.

5.4 Comparing the accuracy for different term weighting schemes

Weighting schemes aim to give higher weight to most discriminative terms. In this section different term count is given to document to measure the importance of term in the documents. There are different weighting schemes applied Boolean (bool), term frequency(tf), inverse document frequency (idf) and (tf-idf). Accuracy for LVQ2.1 when using different presentation schemes present that the term tf-idf has the highest accuracy rate (93.03%). Also, bool, tf and idf model achieved high accuracy rate show

Figure 5.5. Repetition experimental on different stemming techniques (stemmer, light stemmer and raw text) also the term tf-idf achieved high accuracy rate even if the stemming techniques are changed.

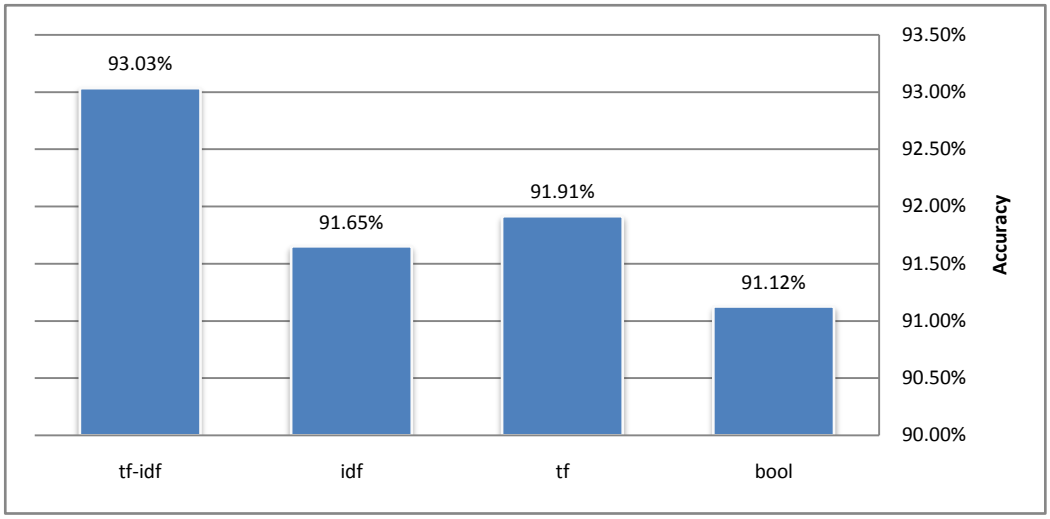

Figure 5.6: Accuracy for term weighting schemes

The required time to build a model for the highest term accuracy need longest time compared with others term weighting schemes. This is because tf-idf needs more computation process because the numbers of operator more than others terms frequencies . In other hand, term frequency (tf) term presentation achieved less time to build LVQ model show Figure 5.6. The tf term used simple equation to weighting term by dividing occurrence of term in document, and the sum of numbers of occurrence of all term's document.

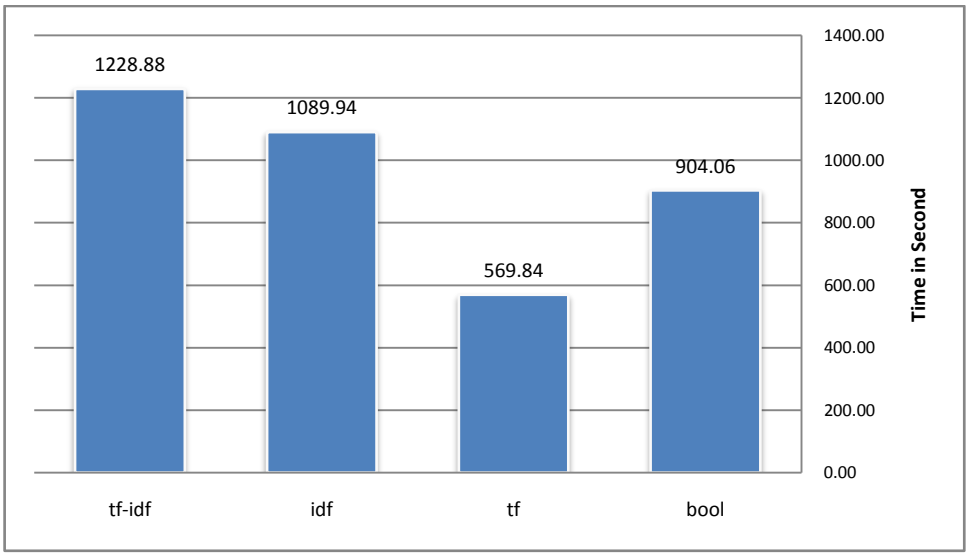

Figure 5.7: Time elapsed to build LVQ model for different term weighting schemes

5.5 Comparing accuracy when increasing term frequency

The minimum term frequency parameter used for preprocessing in String to Word Vector options to set the minimum term frequency (apply term pruning) in the document. The default value for minTermFreq=1 which means applying term pruning on word count that less than 1. In this section we increase term frequency and applied experiments when term frequency equal (3,5,7 and 9) to compare accuracy and time when changing values. The results applied in different stemming techniques (stemming, light stemming and raw text). Figure 5.7 presents stemming techniques in different term frequency values. The observation from results when Freq.=5 the accuracy for stemming and light stemming increased but in raw text decreasing. Also, accuracy decreases when applying TermFreq.=7 or TermFreq.=9 in all stemmer techniques.

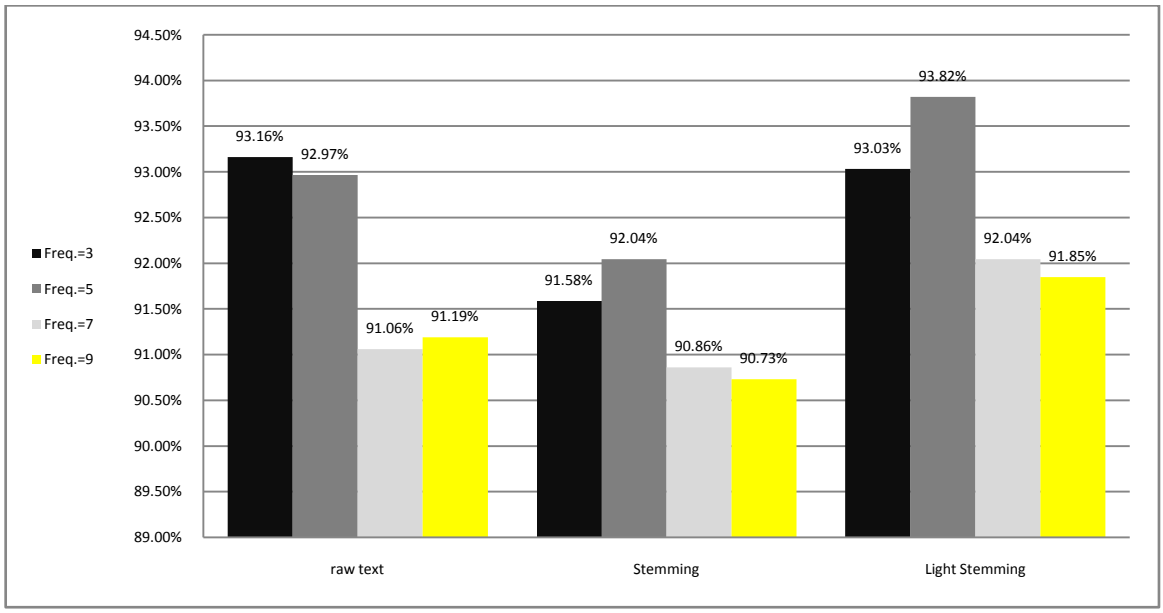

Figure: 5.8 Accuracy for different stemming techniques when increasing frequency term

We can conclude that the best term frequency TermFreq.=5 which achieved high accuracy (93.82%) at light stemming and the time decreasing when is increased TermFreq because the size of data set will be decreased. In all stemmer techniques time decrease but the most is affected by raw text and light stemming. In the frequency=9 all stemmer techniques approximately accumulate in same point which is less than minute for building a LVQ2.1 model as show in Table 5.1.

Table 5.1: Time for building LVQ2.1 algorithms when increase term frequency and using different stemming type

Stemming Type	Time for building LVQ2.1			
	Freq.=3	Freq.=5	Freq.=7	Freq.=9
raw text	298.06	99.29	47.56	26.86
Stemming	62.21	45.71	54.36	37.64
Light Stemming	207.62	75.88	59.78	31.29

5.6 Comparing accuracy and training time with different ANN classification Algorithms

In this section comparison of best LVQ's algorithms (LVQ2.1) with others classification algorithms. This part is comparing accuracy with other neural classification algorithm, Back Propagation neural (BP) network algorithm, Radial Basis Function (RBF) neural network, Kohenon neural algorithm.

Using the same training and testing datasets, Figure 5.10 presents accuracy for neural network classification algorithms, from figure we can conclude that LVQ2.1 as best LVQ's algorithms achieved high accuracy (93.82%) after LVQ2.1 come BP neural network algorithm (91.65%) accuracy. The accuracy decrease to (84.94%) for kohenon and the worst neural is RBF which has accuracy (35.43%).

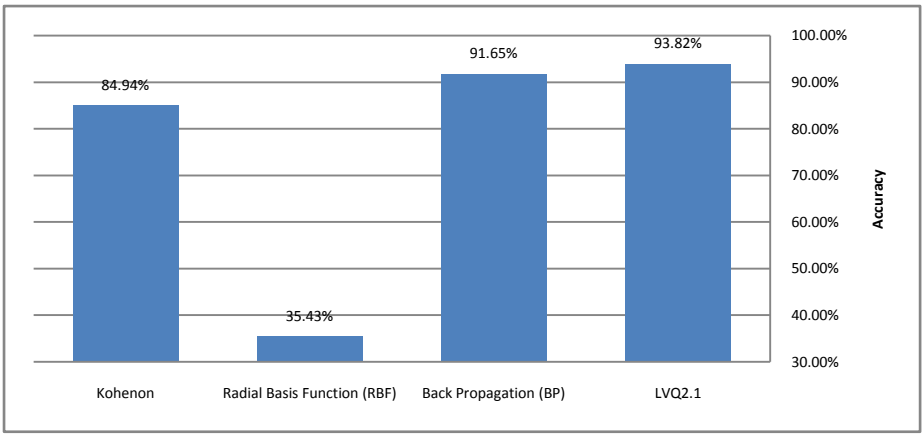

Figure 5.9: Accuracy for neural network classification algorithms

Moreover, Figure 5.11shows the timing required for building neural network algorithms. The results present that LVQ2.1 achieved less timing for training network and building a model. In other hand BP neural algorithm needs a lot of time.

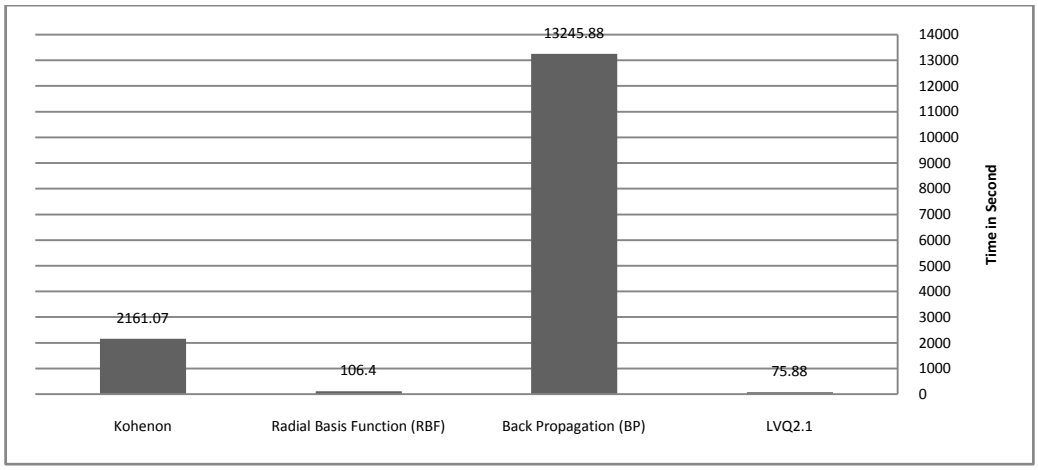

Figure 5.10: Comparing time between different neural network algorithms

From previous results we can conclude that LVQ2.1 as neural network algorithm is able to obtain a high accuracy and a little time when we compare it to other neural network algorithms. For example, Back Propagation and LVQ have a comparative accuracy, but LVQ need less time to build the model. In the other hand, LVQ and Radial Basis Function have comparative time but LVQ has higher accuracy.

5.7 Comparing accuracy with other different classifications algorithms

In this section the comparison of best LVQ's algorithms (LVQ2.1) with others well known classification algorithms. This part comparing accuracy with other traditional classification algorithms Decision Tree (DT), K Nearest Neighbors (KNN) and Naïve Bayes.

Using the same training and testing datasets, Figure 5.12presents accuracy for several classification algorithm, from this figure we can show that LVQ2.1 as best LVQ's algorithms achieved high accuracy (93.82%), after that Naïve Bayes algorithm achieved accuracy (89.08%). The KNN algorithm obtained (87.63%) and value of K=6 determined after series of experiments. The least accurate was for DT (77.71%).

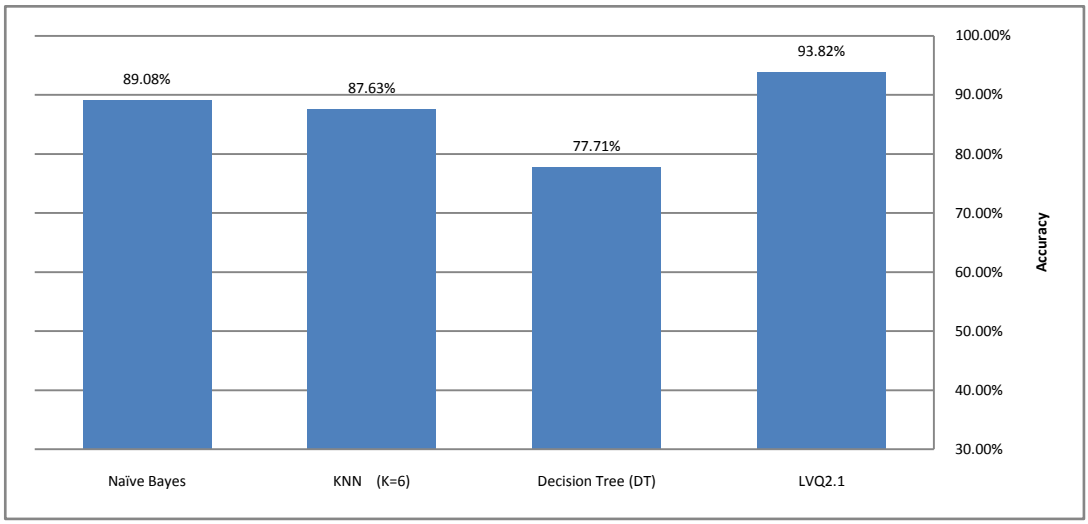

Figure 5.11: Accuracy for several classification algorithms

Moreover, Figure 5.13shows the timing required for building others classification algorithms. The results presents that KNN achieved less timing for building a model. This difference in time resulted because KNN is a lazy learner where all computation happen in testing and at the expense of accuracy. Also neural networks require more time, but it shows that this time much less than some of the classification algorithms like DT .

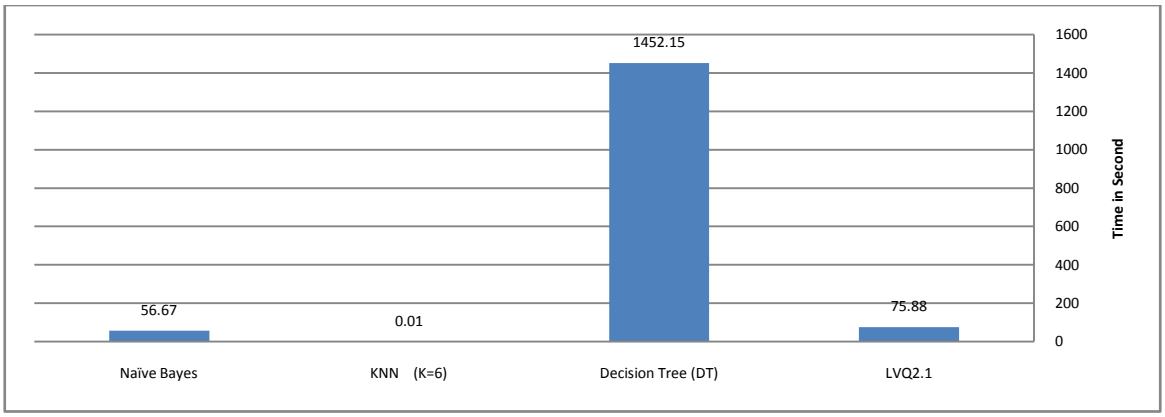

Figure 5.12: Comparing time between different classification algorithms

From previous results we can conclude that LVQ2.1 as neural network algorithm is able to obtain a high accuracy and a little time for some classification algorithms.

From this section we can conclude that the LVQ2.1 algorithms is the best because it achieved high accuracy when we compared it with other neural networks and

traditional classification algorithms. Also, It is true that there are some favor in time for several non neural network algorithms, but this results came at the expense of accuracy.

5.8 Comparing Domains

Table 5.6 shows the precision, recall and F-measure values for classification of documents in separate category using LVQ2.1. The result present that the sport category achieved best precision and recall also achieved f-measure exceed on others category because sports words are limited and clear. It showed that entertainment category had the least f-measure.

Table 5.6: Precision and Recall for LVQ2.1

Category (Domain)	Precision	Recall	F-measure
business	96.9	96.1	96.5
entertainment	94.4	79.1	86.1
middle_east	93.1	94.1	93.6
scitech	85.4	95.0	89.9
sport	98.7	99.6	99.1
world	93.4	92.8	93.1
Average	93.9	93.8	93.8

Summary

From previous sections we can say that LVQ's algorithms achieved the best accuracy and less training time specially when used LVQ2.1 with learning rate value (0.03). The best settings are for preprocessing is using light stemming, tf-idf term weighting and Term frequency = 5. The results present that LVQ2.1 is the better for LVQ's algorithms versions. Also, is the better among neural networks algorithms and among several classification algorithms.

CHAPTER 6:
Conclusion and Future Works

6.1 Conclusion

This thesis improved Arabic text categorization by applying LVQ neural network algorithms. The research presented in this thesis focused on the different versions of LVQ algorithms, to determine the highest accuracy and less training time.

The first experiments applied into different LVQ improvement version (LVQ1, LVQ2.1, LVQ3, OLVQ1 and OLVQ3). The experimental results showed that the performance of the LVQ 2.1 achieved high accuracy (93.03%) and less time rather than other classification algorithms with learning rate value (0.03). To confirm our results we used another dataset to confirm that LVQ2.1 achieved high accuracy also when increase dataset size. The results presented that the LVQ2.1 achieved highest accuracy (95.62%).

The second experiments was done on Arabic text collected from different domains. Due to the classification we selected different preprocessing methods such as term weighting scheme, Arabic morphological analysis (stemming and light stemming) and Term frequency. We found that the best settings are for preprocessing is using light stemming, tf-idf term weighting and Term frequency = 5.

The third set of experiments applied to compare LVQ2.1 with others classification algorithms, (traditional and neural network classification algorithms). The neural network algorithms as Back Propagation neural (BP) network algorithm, Radial Basis Function (RBF) neural network, Kohenon neural algorithm. Also, traditional classification algorithms as Decision Tree (DT), K Nearest Neighbors (KNN) and Naïve Bayes. From experiments, we concluded that LVQ2.1 as neural network algorithm was able to obtain a high accuracy and a little time when we compared it to others neural network algorithms and others traditional classification algorithms.

To confirm our conclusion, Table 6.1, compares our work with other published work in the filed of Arabic text classification.

Table 6.1: Summary table for compare between famous works

System	Algorithms	Accuracy
My research	(LVQ's) LVQ2.1	93.82%
Al-Harbi et. al. [3]	SVM and C5.0	68.65% and 78.42%
El-Kourdi et. al. [20]	Naïve Bayes	68.78%
El-Halees [10]	Maximum Entropy	80.40%
Sawaf et al. [34]	Statistical Methods	62.7%
El-Halees [35] Al-Zoghby [36]	Association Rule	80.4%
Meslh [37]	SVM	88.11%
Harrag and El-Qawasmah [26]	BP Neural Network	88.33%

6.2 Future Work

In the future works, we could extended our work to include the following:

1) Use huge size corpora to make sure that our method can be corporate with such sizes.
2) To do the previous point, adjust our method to work in parallel environment.
3) We also extend experiments to includes others ANN classification algorithms.
4) Use our method with data which has noise such as misspelled words.
5) Work with distributed data, where data can be in more than one place.
6) Ability of our method to work with different kinds of data for example voice data.

References

[1] L. X. Yang Y., "A re-examination of text categorization methods," presented at the In proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development (SIGIR'99), 1999.

[2] Sebastiani F., "Machine Learning in Automated Text Categorization," presented at the ACM Computing Surveys, vol. 34(1), 2002.

[3] A. A. Al-Harbi S., Al-Thubaity A., Khorsheed M., Al-Rajeh A.,, "Automatic Arabic Text Classification," presented at the 9es Journées internationals ,France, 2008.

[4] U. n.-L. o. L. Mart´ın-Valdivia M., Garc´ıa-Vega M.,, "The Learning Vector Quantization Algorithm Applied to Automatic Text Classification Tasks," presented at the Department of Computing, University of Ja´en, Campus Las Lagunillas s/n, Edificio A3, Ja´en, E-23071 SPAIN, 2007.

[5] Kohonen T., "Self-organization and associative memory," presented at the Berlin: Springer-Verlag, 1995.

[6] K. M. Umer M., "Classification of Textual Document using Learning Vector Quantization," *Information Technology Journal, vol. 6(1),* pp. 154-159, 2007.

[7] F. H. Pilevar M.T., Soltani M.,, "Classification of Persian textual documents using learning vector quantization," presented at the appears in: Natural Language Processing and Knowledge Engineering, 2009.

[8] Kohonen T., "Self-Organization Maps," presented at the Spring Third edition, 1995.

[9] A. W. Saad M., "Arabic Text Classification Using Decision Trees," presented at the Workshop on computer science and information technologies CSIT'2010, Moscow - Saint-Petersburg, Russia, 2010.

[10] El-Halees A., "Arabic Text Classification Using Maximum Entropy," *The Islamic University Journal (Series of Natural Studies and Engineering), 15(1),* pp. 157-167, 2007.

[11] Khreisat L., "Arabic Text Classification Using N-Gram Frequency Statistics A Comparative Study," presented at the Proceedings of the 2006 International Conference on Data Mining, Las Vegas, USA, 2006.

[12] Z. T. Syiam M., Habib M., , "An Intelligent System for Arabic Text Categorization," presented at the International Journal of Intelligent Computing and Information Systems IJICIS,vol. 6, no. 1, 2006.

[13] Duwairi R., "Arabic Text Categorization," *Proceedings of the International Arab Journal of Information Technology,* vol. Vol. 4, No. 2, pp. 125-131, April 2007.

[14] NuPhyu T., "Survey of Classification Techniques in DataMining," presented at the Proceedings of the International MultiConference of Engineers and Computer Scientists Vol IIMECS 2009, Hong Kong, 2009.

[15] P.-S. G. Fayyad U., Smyth P. (1996) From Data Mining to Knowledge Discovery in Databases. *AI Magazine* 17-54.

[16] R. M. Mikut R., "Data mining tools," presented at the in Wiley Interdisciplinary Reviews Data Mining and Knowledge Discovery, 2011.

[17] Tan A., "Text Mining: The state of the art and the challenges," presented at the PAKDD'99 Workshop on Knowledge discovery from Advanced Databases (KDAD'99), Beijing, 1999.

[18] H. W. Thabtah F., Al-shammare G., "VSMs with K-Nearest Neighbour to Categorise Arabic Text Data," presented at the Proceedings of the World Congress on Engineering and Computer Science, USA, 2008.

[19] K. G. Al-Shalabi R., Gharaibeh M., " Arabic Text Categorization Using kNN Algorithm," presented at the Proceedings of the Int. multi conf. on computer science and information technology, 2006.

[20] B. A. EL KOURDI M., RACHIDI T., "Automatic Arabic Document Categorization Based on the Naïve Bayes Algorithm," presented at the In the 20th Int. Conf. on Computational Linguistics, Geneva, 2004.

[21] Gershenson C., "Artificial Neural Networks for Beginners," presented at the Formal Computational Skills Teaching Package, COGS, University of Sussex, UK, 2001.

[22] H.-C. A. Harrag F., El-Qawasmeh E., "Performance of MLP and RBF Neural Networks on Arabic Text Categorization Using SVD," presented at the Neural Network World, Czech Republic, 2010.

[23] A.-S. A. M. S. Harrag F., BenMohammed M.,, " A comparative study of Neural networks architectures on Arabic text categorization using feature extraction," presented at the appears in: Machine and Web Intelligence (ICMWI), 2010 International Conference on Issue, 2010.

[24] F. N. Mahmoudi M. , Lucas C., Taghiyareh F., "Artificial Neural Network Weights Optimization Based on Imperialist Competitive Algorithm," presented at the Seventh International Conference on Computer Science and Information Technologies CSIT 2000, Yerevan, Armenia, 2009.

[25] E.-Q. E. Harrag F., "Neural Network for Arabic Text Classification," presented at the 2nd International Conference on the Applications of Digital Information and Web Technologies, ICADIWT 2009, 2009.

[26] E.-Q. E. Harrag F. , Al-Salman A., "Comparing Dimension Reduction Techniques for Arabic Text Classification using BPNN algorithm," presented at the Proceedings of the first International integrated intelligent Computing (ICCIA) Technical co-sponsor by IEEE Computational Intelligence Society, 2010.

[27] H. S. Oyang Y., Ou Y., Chen C., Chen Z., "Data Classification with Radial Basis Function Networks Based on a Novel Kernel Density Estimation Algorithm," presented at the To appear in the January issue of IEEE Transactions on Neural Networks, 2005

[28] Kohonen T., "Self Organizing Maps," presented at the Springer Series in Information Sciences Springer: Espoo, Finland, 1994.

[29] Heaton J., *Introduction to Neural Network with Java*: Kusumadewi, Sri, " Artificial Intelligence (Teknik dan Aplikasinya)", Graha Ilmu,Yogyakarta, 2003.

[30] B. M. Witoelar A., Hammer B., , "Learning Vector Quantization: generalization ability and dynamics of competing prototypes," presented at the Proc. of 6th Int. Workshop on Self-Organizing Maps (WSOM), Univ. Bielefeld, 2007.

[31] G. M. Sommer D., "A Comparison of Validation Methods for Learning Vector Quantization and for Support Vector Machines on Two Biomedical Data Sets," presented at the In M. Spiliopoulou et al. (eds.): From Data and Information Analysis to Knowledge Engineering, 2006.

[32] B. M. H. Ben Khalifa K., Dogui M., Alexandre F., "Alertness states classification with SOMand LVQ Neural Networks," presented at the International Journal onInformation Technology, 2005.

[33] T. Kohonen, "Improved version of learning vector quantization," presented at the IJCNN, San Diego, 1990.

[34] Z. J. Sawaf H., Ney H., "Statistical Classification Methods for Arabic News Articles," presented at the Paper presented at the Arabic Natural Language Processing Workshop (ACL2001), Toulouse, France, 2001.

[35] El-Halees A., "Mining Arabic Association Rules for Text Classification," presented at the In the 1st Int. Conf. on Mathematical Sciences, Al-Azhar University-Gaza, Palestine, 2006.

[36] E. A. Al-Zoghby A., Ismail NA. Hamza T., " Mining Arabic Text Using Soft Matching association rules," presented at the In the Int. Conf. on computer Engineering & System, ICCES'07, 2007.

[37] Mesleh A., "Chi Square Feature Extraction Based Svms Arabic Language Text Categorization System," *Journal of Computer Science 3(6)*, pp. 430-435, 2007.

[38] E.-Q. E. Harrag F., Pichappan P.,, "Improving Arabic text categorization using decision trees," presented at the In the 1st Int. Conf. of NDT'09, 2009.

[39] F. Z. Habib M., Gharib T.,, "A Hybrid Feature Selection Approach for Arabic Documents Classification," *Egyptian Computer Science Journal ECS,vol. 28, no. 3*, pp. 1-7, 2006.

[40] M. F. Umer and M. S. H. Khiyal, " Classification of Textual Documents Using Learning Vector Quantization," *Information Technology Journal*, vol. 6, pp. 154-159, 2007.

[41] Attia M., "Arabic tokenization system," presented at the Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources, 2007.

[42] A. I. AlotaibyF., Foda S.,, "Processing large Arabic text corpora: Preliminary analysis and results," presented at the In Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt, 2009.

[43] H. M. Gharib T., Fayed Z.,, "Arabic Text Classification Using Support Vector Machines," *The International Journal of Computers and Their Applications ISCA, vol.16, no.4*, pp. 192-199, 2009.

[44] A. A. B. Sembok T., Abu Bakar Z.,, "A Rule and Template Based Stemming Algorithm for Arabic Language," *INTERNATIONAL JOURNAL OF MATHEMATICAL MODELS AND METHODS IN APPLIED SCIENCES,Issue 5, Volume 5*, pp. 974-981, 2011.

[45] N. J. Kadri Y., "Effective Stemming for Arabic Information Retrieval," presented at the The Challenge of Arabic for NLP/MT, International Conf. at the British Computer Society (BCS), London, 2006.

[46] Smirnov I., DePaul University, . "Overview of Stemming Algorithms," presented at the Mechanical Translation. Available at: http://the-smirnovs.org/info/stemming.pdf., 2008.

[47] A.-A. F. Al-Fedaghi S., " A new algorithm to generate Arabic root-pattern forms," presented at the In proceedings of the 11th national Computer Conference and Exhibition, King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia, 1989.

[48] A. E. S. Sawalha M., "Comparative evaluation of Arabic language morphological analysers and stemmers," presented at the Proceedings of COLING 2008 22nd International Conference on Comptational Linguistics, COLING 2008, 2008.

[49] B. L. Larkey L., Connell M.,, "Light stemming for Arabic information retrieval," presented at the In Abdelhadi Soudi, Antal van den Bosch, and Günter Neumann, editors Arabic Computational Morphology: Knowledge-based and empirical method , volume 38 of Text, Speech and Language Tech-nology, Springer Verlag, 2007.

[50] Darwish K., "Building a Shallow Arabic Morphological Analyzer in One Day," presented at the Proceedings of the ACL-02 workshop on Computational approaches to semitic languages, 2002.

[51] e. a. Al-Ameed H., "ARABIC LIGHT STEMMER: ANEW ENHANCED APPROACH," presented at the Proceedings of the Second International Conference on Innovations in Information Technology (IIT'05), 2005.

[52] L. M. Meftouh K., Smaili K.,, "MODELING ARABIC LANGUAGE USING STATISTICAL METHODS," *Proceedings of the Arabian Journal for Science and Engineering, Volume 35, Number 2C,* pp. 69-82, 2010.

[53] e. a. Mayfield J., "JHU/APL at TREC 2001: Experiments in filtering and in Arabic, video, and webretrieval," presented at the In TREC 2001. Gaithersburg: NIST, 2001.

[54] N. O. Massimo M., "A novel method for stemmer generation based on hidden Markov models," presented at the Proceedings of the twelfth international conference on Information and knowledge management, 2003.

[55] G. C. Qiu Z., Doherty A.R., Smeaton, A.F.,, "Term Weighting Approaches for Mining Significant Locations from Personal Location Logs," presented at the Proceedingsin CIT(2010), 2010.

[56] T. C. Lan M., Low H., Sung S.,, "A comprehensive comparative study on term weighting schemes for text categorization with support vector machines," presented at the Special interest tracks and posters of the 14thinternational conference on World Wide Web, Chiba, Japan, 2005.

[57] S. M. Srivastava A., "Text Mining: Classification, Clustering, and Applications," presented at the Chapman & Hall/CRC, ISBN: 1420059408, 2009.

[58] U. V. Nanas N., De Roeck A., Domingue J.,, "A comparative study of term weighting methods for information filtering," presented at the Technical Report KMi-TR-128, Knowledge Media Institue, The Open University, 2003.

[59] Ramos J., "Using tf-idf to determine word relevance in document queries," presented at the In First InternationalConference on Machine Learning, New Brunswick:NJ, USA, Rutgers University, 2003.

[60] H. K. Feinerer I., Meyer D.,, "Text mining infrastructure in R," *Journal of Statistical Software, Volume 25, Issue 5,* 2008.

[61] F. E. Bouckaert R. , Hall M., Holmes G., Pfahringer B., Reutemann P. ,Witten I.,, "WEKA-experiences with a java open-source project," *Journal of Machine Learning Research, JMLR,* pp. 2533-2541, 2010.

[62] http://sourceforge.net/projects/ar-text-mining.