International Journal of Computer Science
and Engineering (IJCSE)
ISSN 2278-9960
Vol. 2, Issue 3, July 2013, 137-152
© IASET

International Academy of Science,
Engineering and Technology
Connecting Researchers; Nurturing Innovations

MODERN HIGH-EFFICIENCY AUTOMATIC CATEGORIZATION ALGORITHM FOR ARABIC CORPUS

ABD ELZIZ HASSAN KHARSANI[1], SAMANI A. TALAB[2] & AWAD H. ALI[3]

[1]Department of Computer Science, Salman Bin Abdulaziz University, Kharj, Saudi Arabia

[2,3]Department of Computer Science, University of Neelain, Khartoum, Sudan

ABSTRACT

Text categorization is the process of classifying documents into a predefined set of categories based on the content of the documents. Many studies have discussed the topic, but in principle there are many obstacles to automatizing the categorization process. This paper describes a hybrid commercial preprocessing stemming algorithm to improve the accuracy of the stemming method. The effectiveness of different methods in the Arabic text categorization process is evaluated, and the most suitable methods for the Arabic language are chosen. We achieve improvements of about 96% in the classification process by human expert evaluation. This tool is found to be essential in automatizing the categorization process, because Arabic versions are needed for development and usage purposes.

KEYWORDS: Machine Learning, Stemming Approaches, Bilateral Names, Stop Words, The Root, Document Indexing, The Classifier, Text Categorization

INTRODUCTION

Machine learning approaches are applied to build an automatic text classifier by learning from a set of previously classified documents [29]. In a text categorization system the document must pass through a set of steps: document conversion that converts different types of documents into plain text, stop-word removal to remove insignificant words, stemming to group words that share the same root, feature selection/extraction, super vector construction, feature weighting, classifier construction, and classification.

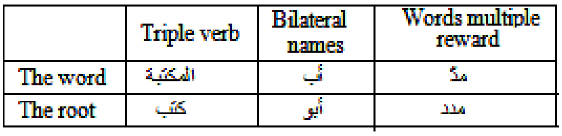
The Arabic language is a Semitic language that is more complex and has a greater morphology than English. It is a highly inflected language, and because of this complex morphology it needs a set of preprocessing routines to be suitable for manipulation. Stop words like prepositions and articles are considered insignificant words and must be removed, and words must be stemmed after stop-word removal. Stemming is the process of removing the affixes from the word and extracting the word root [2,3,4,5,6,7,8]. After the preprocessing routines are applied, the document passes through a document indexing process that involves the creation of an internal representation of the document. The indexing process consists of three phases [9,10]. Finally, the classifier is constructed by learning the characteristics of every category from a training set of documents. Once the classifier is built, its effectiveness (i.e., its capability to take the right categorization decisions) may be tested by applying it to the test set and checking the degree of correspondence between the decisions of the classifier and those encoded in the corpus.

RELATED WORK IN ARABIC TEXT PREPROCESSING

When text documents are categorized, not all features equally represent the semantics of the document. In fact, some of these features may be redundant, adding nothing to the meaning of the document. Others might be synonymous, and therefore capturing one of them would be enough to enhance the semantic for categorization purposes. Consequently, the effective selection of feature words, which reflect the main topics of the text, is an important factor in text categorization. Stemming techniques can be used in Arabic text preprocessing to reduce multiple forms of the word to one form (root or stem). Very little research has been carried out on Arabic text.

The nature of Arabic text is different from that of English text, and the preprocessing of Arabic text is the more challenging stage in text categorization in particular and text mining in general. Table 1 shows the effect of preprocessing.

Table 1: An Example of Root/Stem Preprocessing with Stemming



	Triple verb	Bilateral names	Words multiple reward
The word	المكتبة	أب	مدّ
The root	كتب	أبو	مدد

Stemming can be defined as the process of removing affixes (prefixes, infixes, or/and suffixes) from words to reduce these words to their stems or roots. A root can be defined as a word that cannot be created from another word—in other words, a word without prefixes, infixes, or suffixes. For example, as in Table 1, the root of the Arabic word (المكتبة, the Library) is (كتب, Library). In some anomalous cases the stemmer should reword the bilateral word to triple origin, and this was not taken into account before (مدّ) becomes (مدد).

An Arabic stemming algorithm can be classified, according to the desired level of analysis, as a root-based approach (exp Khoja [11,12]) or a statistical approach (n-gram [13,14]). This section provides a brief review on the three stemming approaches for stemming Arabic text.

A root-based stemmer uses morphological analysis to extract the root of a given Arabic word. Many algorithms have been developed for this approach. The AI-Fedaghi and AI-Anzi algorithm tries to find the root of the word by matching the word with all possible patterns, with all possible affixes attached to the word [15].

The algorithm does not remove any prefixes or suffixes. The AI-Shalabi morphology system uses different algorithms to find the roots and pattern [16]. This algorithm removes the longest possible prefix, and then extracts the root by checking the first five letters of the word.

This algorithm is based on the assumption that the root must appear in the first five letters of the word. Khoja has developed an algorithm that removes prefixes and suffixes, all the time checking that the algorithm is not removing part of the root, and then matches the remaining word against the patterns of the same length to extract the root [11].

The aim of the stem-based or light stemming approach is not to produce the root of a given Arabic word, but to remove the most frequent suffixes and prefixes. Some authors have mentioned light stemming [17,18,12,19], but until now no standard algorithm for Arabic light stemming has been produced.

All trials in this field have consisted of a set of rules to strip off a small set of suffixes and prefixes. There is also no definite list of these strippable affixes. In statistical stemming, related words are grouped based on various string similarity measures. Such approaches often involve n-gram [13,14].

Note that we can select, from among the best known Arabic stemming algorithms for each approach, the Khoja stemmer as root-based [11,12] and the N-Gram as statistics-based [13]. These stemming algorithms present some weakness when they are used alone. Table 2 summarizes some of these weaknesses.

Table 2: Some Weaknesses of Khoja and N-Gram Stemmers

Algorithm	Approach	Weakness
Khoja	Root-based	The Khoja stemmer removes the bilateral and multiple words because they are less than two characters.
N-Gram	Statistical approach	The N-Gram algorithm also removes bilateral and multiple words because they are less than two characters.

To overcome this problem and therefore enhance the performance of the stemming algorithm, we propose in section 4 a new efficient hybrid light+trigram stemmer approach.

ARABIC TEXT CATEGORIZATION

This paper presents the application of machine learning strategies in the field of Arabic text categorization. Many studies discuss text categorization systems for other languages, but few researches involve the Arabic language according to a performed survey. Thus, text collection was performed using local Sudanese newspapers, then the preprocessing routines on the documents were undertaken. This preprocessing included the removal of stop words and the stemming of the documents to cluster the terms according to their similarity. Three stemming approaches were tested, and the results show the hybrid approach of light and statistical stemming to be the most suitable for text categorization tasks in the Arabic language [28]. After stemming, a dictionary was constructed from terms that appear in all the documents at least once. Due to the very high dimensionality of this dictionary (trainer program), several methods for selecting highly informative terms were used. A hybrid method for term selection that combines document frequency thresholding and information gain is proposed.

This proposed method gives excellent results. After term selection, every document is represented as a vector of the weights of the terms. Four term weighing criteria were used; normalized-tfidf is the suggested weighting method. Finally, two non-parametric classifiers were used: the k-NN classifier and Rocchio classifier. The Rocchio classifier shows superiority over k-NN in both efficiency and generalization. In general, most Arabic text categorization tasks involve several steps. The proposed model contains a set of phases that describe the documents: preprocessing routines, document representation techniques, and classification process. Document preprocessing routines include stop-word removal to remove insignificant words and stemming to group words that share the same root.

After that, the super vector is constructed. Feature selection techniques are applied to reduce the dimensionality of the super vector. The document is represented as a vector of weighted terms. Finally, the classifier is constructed and evaluated. Every phase will be described in detail. This section describes the implementation of the workflow steps of two programs, the Classifier Trainer and the Arabic Text categorization (ATC). Basically, the steps in the two programs are similar. Therefore, the "Rocchio" section is divided into two sub-sections to define the implementation details for both programs.

First, stemming is the process of removing all affixes from a word to extract its root. It has been shown to improve performance in information retrieval tasks, especially with highly inflected languages such as Arabic. For the Arabic language there are three different approaches to stemming: the root-based stemmer, the light stemmer, and the statistical stemmer. The root-based stemmer uses morphological analysis to extract the root of a given Arabic word [3,4,5,1]. The aim of the light stemming approach is not to produce the root of a given Arabic word, but to remove the most frequent suffixes and prefixes. The light stemmer is mentioned by some authors [2,6,7,8,27]. In the statistical stemmer, related words are grouped based on various string similarity measures. Such approaches often involve n-gram [7,20]. In our approach the three methods for stemming are tested, with the model providing a comparative study for the root-based stemmer, the light stemmer, and the statistical stemmer to decide which approach is suitable for Arabic text categorization tasks. The hybrid stemmer was found to surpass the other approaches, as discussed in detail in the experimental results section.

Second, to consume an Arabic word from a document, the program processes each character one by one and checks if it is one of the following “legal” characters: Arabic letter, diacritic, and Tatweel-stretching character. Once the program encounters an “illegal” character that does not meet any of the criteria above (like a number or English letter), it finalizes the word. This word is then passed to the next stage, the “stop-words filter”. When program returns to this stage to consume the new word, it will continue processing the characters. It ignores all illegal characters. Once the program encounters a “legal” character, it will start constructing the new word again.

Third, after stop-word removal and stemming, documents are indexed. In true information retrieval style, each document is usually represented by a vector of n weighted terms. This is often referred to as the bag of words approach to document representation [21]. This approach ignores the structure of a document and the order of words in the document. The feature vectors represent the words observed in the documents.

The super vector W (w1,…, wd) in the training set consists of all the distinct words (also called terms) that appear in the training samples after removing the stop words and word stemming. Typically, there can be thousands of features in document classification. Hence, a major characteristic, or difficulty, of text categorization problems is the high dimensionality of the feature space.

Many term evaluation functions have been introduced for term selection for English text categorization [22,23,10]. These functions are document frequency thresholding, information gain, CHI square, odds ratio, NGL coefficient and GSS score. The resultant word is then stored in the database table (Word). Each unique word contains a unique ID. These unique words are considered features. Use of the database makes the retrieval of information flexible and faster.

Fourth, after the significant terms are selected, each term is weighted for every document [10]. Term weighting refers to the different ways of computing term weights. Many weighting schemes are evaluated for English and other languages [24].

Fifth, for classification, there are two main approaches to the construction of text categorization systems. A number of systems embody approaches similar to those used in expert systems for classification or diagnosis. Knowledge engineers define one or more layers of intermediate conclusions between the input evidence (words and other textual features) and the output categories, and write rules for mapping from one layer to another, and for confirming or removing conclusions.

The second strategy is to use existing bodies of manually categorized text in constructing categorizers by inductive learning. After a classifier is constructed, it must be evaluated for the text categorization task. Many different evaluation criteria have been used for evaluating the performance of categorization systems [9,29]. The experimental evaluation of a classifier usually measures the generalization of the classifier, rather than its efficiency, that is, its ability to make the right classification decisions, as shown in Figure 1.

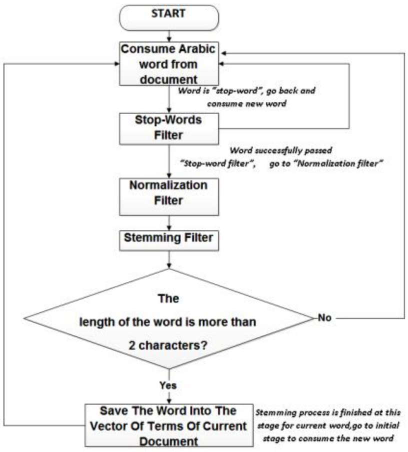

START
Consume Arabic word from document
Word is "stop-word", go back and consume new word
Stop-Words Filter
Word successfully passed "Stop-word filter", go to "Normalization filter"
Normalization Filter
Stemming Filter
The length of the word is more than 2 characters?
No
Yes
Save The Word Into The Vector Of Terms Of Current Document
Stemming process is finished at this stage for current word,go to initial stage to consume the new word

Figure 1: Flowchart of Stemming Algorithm during Processing of the Document

STATEMENT OF THE PROBLEM

This paper uses a hybrid approach of the light and the statistical stemmers. This technique is good with a few categories, but it has a big problem removing bilateral words. Thus, the researcher found that by generating new categories some errors appear, causing a reduction in categorization accuracy, such as a debilitating word and/or terms consisting of two characters (مدّ،عدّ،أب،دم،أم،رمة،أفّ ,بخ ,غس ,بَسّ،ضغ). Some of these words have a very important impact in determining the new categories, such as the word (blood=دم) which has a strong relationship to medicine categories, and the word (بَسّ، غس، ضغ), which has a strong relationship to animals categories. Thus, it was necessary to make improvements in the stemming algorithm to increase the accuracy of classification and to expand the possibility of generating new categories (see Table 3).

Table 3: Bilateral Words

Bilateral words
مرّ، شقّ، هد، عفّ، رقّ ، خفّ، حكّ، هزّ، ظنّ ، سرّ، شكّ، رك، قرّ، شحّ، قلّ، بتّ، صرّ، أفّ، أفّ، بخّ، غس، بسّ، ضغّ، صكّ، مكّ، طبّ، دبّ، دفّ، قرّ، دقّ، هزّ، سفّ، دنّ، حنّ.............

SUMMARY OF PROPOSED CATEGORIZATION PROCESS

Stemming Algorithm

A light stemming algorithm is developed. It removes the most common suffixes and prefixes and keeps the form of the word without changing it. The following algorithm and Figure 2 show the stemming approach:

For Every Word in the Text

- IF the word is not an Arabic word, THEN consider this word as a useless word.
- IF the word contains digits, THEN consider this word as a useless word.
- IF the word length < 3 characters, THEN consider this word as a useless word, except for usefulness word:
  - Bilateral names (أب،دم)
  - Words multiple reward (مدّ)
- Remove diacritics.
- Normalize the word.

- IF the word is a stop word, THEN consider this word as a useless word.
- Remove prefixes.
- Recursively remove suffixes.
- IF the word is a stop word, THEN consider this word as a useless word.

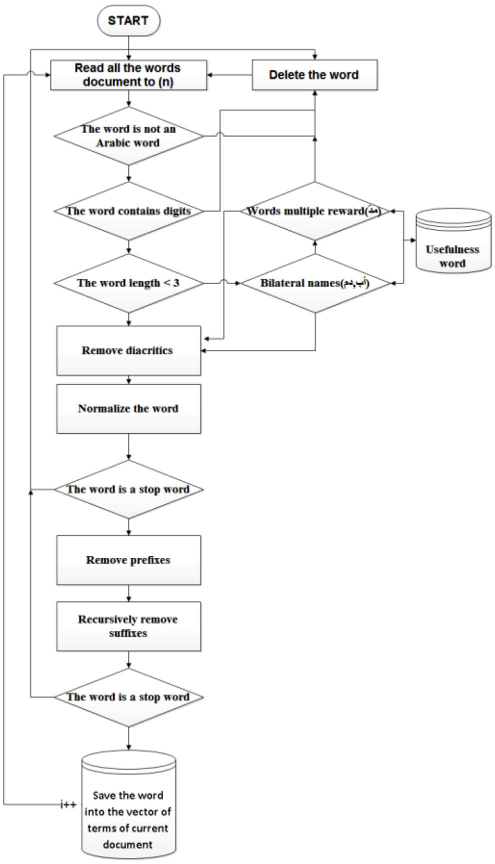


Figure 2: Flowchart of the Stemming Process

Stop-Words Filter

There is a predefined “stop-word” list in the program. If the new word matches any word in “stop-word” list, this word is ignored and the flow is returned to the previous stage to consume the new word. In case of success, the flow will pass through to the next stage, the “normalization filter” [25].

Normalization Filter

Normalization is a pre-stage of actual stemming, which is performed to unify the word.

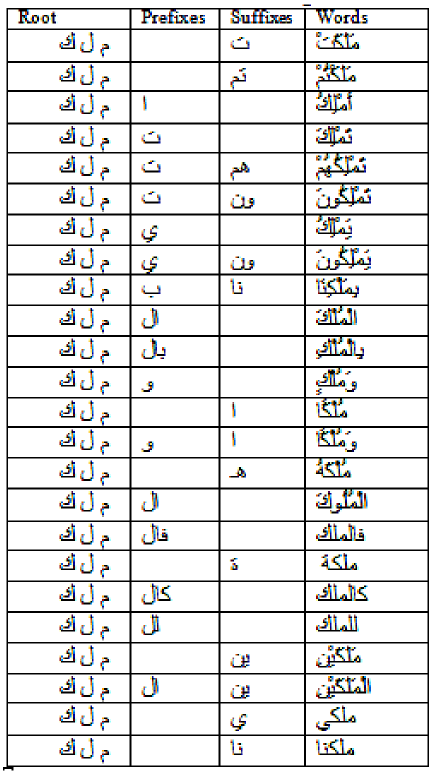
The actions are applied to the word:

- Alef with madda above (آ), Alef with hamza above (أ), and Alef with hamza below (إ) are replaced with Alef (ا).
- Alef Maksura (ى) is replaced with Yeh (ي).
- Teh Marbuta (ة) is replaced with Heh (ه).
- Remove all Arabic diacritics.
- Remove all Tatweel characters (-).

Stemming Filter

The program removes all Arabic prefixes and suffixes from the word. Table 4 shows the prefixes and suffixes to be removed.

Table 4: Results of Stemming Filter



Root	Prefixes	Suffixes	Words
م ل ك		ت	مَلَكْتَ
م ل ك		تم	مَلَكْتُم
م ل ك	ا		أَمْلِكُ
م ل ك	ت		تَمْلِكُ
م ل ك	ت	هم	تَمْلِكُهُمْ
م ل ك	ت	ون	تَمْلِكُونَ
م ل ك	ي		يَمْلِكُ
م ل ك	ي	ون	يَمْلِكُونَ
م ل ك	ب	نا	بِمُلْكِنَا
م ل ك	ال		المُلْكُ
م ل ك	بال		بِالمُلْكِ
م ل ك	و		وَمُلْكِ
م ل ك		ا	مَلَكَا
م ل ك	و	ا	وَمَلَكَا
م ل ك		هـ	مَلَكَهُ
م ل ك	ال		المُلُوكُ
م ل ك	فال		فالملك
م ل ك		ة	ملكة
م ل ك	كال		كالملك
م ل ك	لل		للملك
م ل ك		ين	مَلَكَيْنِ
م ل ك	ال	ين	المَلَكَيْنِ
م ل ك		ي	ملكي
م ل ك		نا	ملكنا

Term Selection

The Document Frequency Term Selection method was implemented. The program involves two types of “word frequencies”:

- Word frequency (in scope of the document) – the number of times the word occurs within the document
- Document frequency for a word (in scope of the corpus) – the number of documents in which the word occurs at least once, as show in the diagram in Figure 3, the algorithm of calculating “word frequencies”. The program processes each document one by one in the corpus

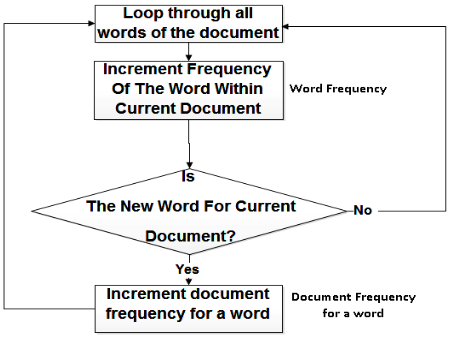

Loop through all words of the document
Increment Frequency Of The Word Within Current Document
Word Frequency
Is The New Word For Current Document?
No
Yes
Increment document frequency for a word
Document Frequency for a word

Figure 3: Flowchart of Term Selection

In the term selection step, the program checks document frequency for collected words in the corpus. If document frequency for the word is less than 2, the word is removed from the document where it occurs. Document frequency is not applied for a small number of documents in the corpus (less than 100 documents). This is done on purpose in order to preserve some essential words in the documents, so categorization results will be more correct.

Weighting

Weighting is the process of assigning the value for each word in the documents. The weight of the word indicates how significant or useful this word is within the document.

The Classifier Trainer program implements “normalized-tfidf weighting” using the following formula:

$$\mathrm{a}_{\mathrm{iK}} = \frac{f_{\mathrm{iK}} * \log\left(\frac{N}{n_i}\right)}{\sqrt{\sum_{j=1}^{M}\left[f_{\mathrm{iK}} * \log\left(\frac{N}{n_i}\right)\right]^2}}, \quad (1)$$

where

- a_{iK} – the weight of word i in the document k;
- f_{iK} – the frequency of word i in document k;
- N – the number of documents in the corpus;
- n_i – the document frequency for a word;
- M – the number of rows in the document.

The ATC program implements “term frequency weighting” with the following formula:

- $\mathrm{a}_{\mathrm{iK}} = f_{\mathrm{iK}}$,

where:

- a_{iK} – the weight of word i in the document k;
- f_{iK} – frequency of word i in document k.

In the experimental section our internal experiments show that categorization results are better when the ATC implements “term frequency weighting” instead of “normalized-tfidf weighting”

Rocchio Classification

To train the classifiers, the extracted feature vectors as well as the correct labels are needed. The data used for training consist of a set of 60,000 texts, chosen very carefully from six categories, with a mean of 10000 for each category, which are important to make the categorization process. The categories are Culture, Economy, International news, Local news, Religion, and Sports. This program creates centroids per-category to be used in the ATC program. Centroids per-category are presented as the vector of accumulated weights of unique words collected from all the documents belonging to a certain category. Figure 4 shows the scheme to create the vector of centroids per-category.

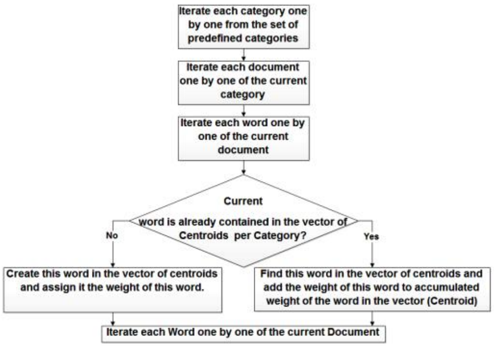
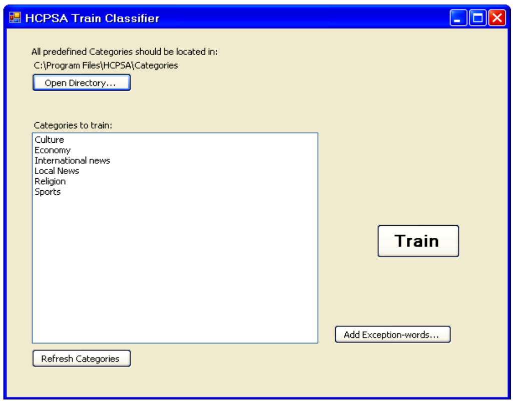

Iterate each category one by one from the set of predefined categories
Iterate each document one by one of the current category
Iterate each word one by one of the current document
Current word is already contained in the vector of Centroids per Category?
No
Yes
Create this word in the vector of centroids and assign it the weight of this word.
Find this word in the vector of centroids and add the weight of this word to accumulated weight of the word in the vector (Centroid)
Iterate each Word one by one of the current Document

Figure 4: Flowchart of the Classifier Trainer



All predefined Categories should be located in:
C:\Program Files\HCPSA\Categories

Categories to train:
Culture
Economy
International news
Local News
Religion
Sports

Figure 5: Screenshot of our Proposed Classifier Trainer

Document to Certain Category

This is the final step, in which the actual assignment of the document to a certain category is performed.

The theory behind the Rocchio classifier is that if we average the tf-idf vectors of all the documents in each group, we will end up with one centroid per group (created by the Classifier Trainer program). We then compare the document in question to our averaged group centroids, and assign the document to the group to which it is most similar. For a better understanding of the implementation details, the following terms are defined:

- Score table – the table created for each document to be assigned to a certain category;
- Points per category – calculated points, which indicate the relevance of the document to a certain category. Table 5 shows the schema of scores.

Table 5: Score of Document to Certain Category

Predefined Category	Points per Category
Category name 1	<decimal number>
Category name 2	<decimal number>
etc…	<decimal number>

Points per category" is calculated by the following formula:

$$\text{Points per cattegory} = \sum_{n=0}^{z} \left(w_{ic} * a_{iK} \right), \tag{2}$$

where

- w_{ic} – the weight (centroid) of word (i) in the category of centroids (c) (calculated in the Classifier Trainer program and loaded from centroids XML-files);

- W_{ic} – the weight of word (i) in the document (k) (calculated in the step of Weighting in Automatic Text Categorization, ATC program);
- z – the number of times word (i) is found in the vector in category of centroids (c).

The algorithm below illustrates the way the score table is calculated for each document in the corpus:

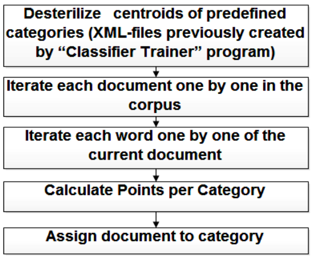

Desterilize centroids of predefined categories (XML-files previously created by "Classifier Trainer" program)
Iterate each document one by one in the corpus
Iterate each word one by one of the current document
Calculate Points per Category
Assign document to category

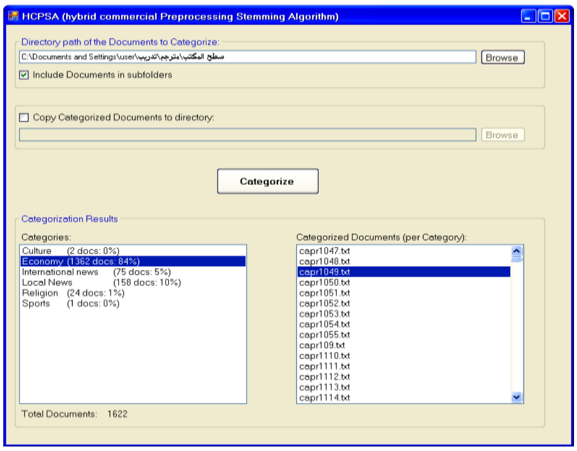
Figure 6: Flowchart of Document to Certain Category Algorithm

Once the calculation of the score table is completed for the document, the program calculates the max value in the "Points per-category" column. The current document will be assigned to the category name corresponding to the max value to be found. The methodology of our proposed classifier is to get the documents that already belong to some category (in our case, "Economy"), then categorize "Economy" documents. In the end of the categorization process we look at the results. The number of documents assigned, for example to the Economy category, is the accuracy of the ATC classifier. The following example illustrates the general steps of our classifier for finding documents that belong to 'Economy," as shown in Figure 7.

A total of 1622 documents were selected in advance from Sudanese economic newspapers for the period 2011-2012. These documents had not been used in training the classifier program. The steps are as follows:

- As the mandatory pre-step, centroids are created.
- After that, the ATC program is launched.
- The path to the documents is specified to categorize in "directory path of the documents to Economy categorize".
- The process of "Categorize" begins.

Table 6 shows the results of the categorization.



Directory path of the Documents to Categorize:
C:\Documents and Settings\user\تدريب\مترجم\سطح المكتب

☑ Include Documents in subfolders

☐ Copy Categorized Documents to directory:

Categorization Results

Categories:

Culture (2 docs: 0%)
Economy (1362 docs: 84%)
International news (75 docs: 5%)
Local News (158 docs: 10%)
Religion (24 docs: 1%)
Sports (1 docs: 0%)

Categorized Documents (per Category):

capr1047.txt
capr1048.txt
capr1049.txt
capr1050.txt
capr1051.txt
capr1052.txt
capr1053.txt
capr1054.txt
capr1055.txt
capr109.txt
capr1110.txt
capr1111.txt
capr1112.txt
capr1113.txt
capr1114.txt

Total Documents: 1622

Figure 7: Screenshot of our Proposed Text Categorization

Table 6: Categorization Results

Category	%	Count
Culture	0	2
Economy	84	1362
International news	5	75
Local news	10	158
Religion	1	24
Sports	0	1

The results show that the ATC program correctly assigned 1362 of 1622 documents to the "Economy" category. To calculate the percentage of documents belonging to this category, the program used Formula 3:

$$\frac{1362}{1622}\times 100 = 83.9\% \approx 84\% \tag{3}$$

EVALUATION

The Classifier Evaluation

Classification generalization is usually measured in terms of the classic information retrieval notions of precision (π) and recall (ρ) [26], adapted to the case of text categorization. Precision (πi) with respect to ci is the probability that if a random document dx is classified under ci, this decision is correct. Analogously, recall (ρi) with respect to ci is defined as the probability that, if a random document dx ought to be classified under ci [27], this decision is taken. Precision and recall are calculated as follows:

$$\pi_i = \frac{a_i}{a_i + b_i}, p_i = \frac{a_i}{a_i + c_i}, \tag{4}$$

Where

- "a" is the number of documents correctly assigned to this category;
- "b" is the number of documents incorrectly assigned to this category;
- "c" is the number of documents incorrectly rejected from this category;
- "d" is the number of documents correctly rejected from this category.

To obtain estimates of π and ρ, two different methods may be adopted, as follows:

Micro averaging: π and ρ are obtained by summing over all individual decisions:

$$\pi^{\mu} = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|} a_i + b_i}, p^{\mu} = \frac{\sum_{i=1}^{|C|} a_i}{\sum_{i=1}^{|C|} a_i + c_i}. \tag{5}$$

Macro averaging: π and ρ are first evaluated "locally" for each category, and then "globally" by averaging over the results of the different categories:

$$\pi^{M} = \frac{\sum_{i=1}^{|C|} \pi_i}{|C|}, p^{M} = \frac{\sum_{i=1}^{|C|} p_i}{|C|}. \tag{6}$$

Since most classifiers can be arbitrarily tuned to emphasize recall at the expense of precision (and vice-versa), only combinations of the two are significant. The most popular way to combine the two is the function $F_{\beta i} = \frac{(B^2+1)\pi_i p_i}{\beta^2 \pi_i + p_i}$, for some value $0 \le \beta \le \infty$. Usually, β is taken to be equal to 1, which means that the $F_{\beta i}$ function becomes $F_{1i} = \frac{2\pi_i p_i}{\pi_i + p_i}$, i.e., the harmonic mean of precision and recall. Similar to precision and recall, the F_β function can be estimated using two methods: micro averaging and macro averaging.

Experimental Setting for Classifier Evaluation

In our approach three methods for stemming are tested, with the model providing a comparative study for the root-based stemmer, the light stemmer, and the statistical stemmer to decide which approach is suitable for Arabic text categorization tasks. This automatic categorizer extraction is compared with a manual strategy achieving almost the same improvement. Experimental results shows that the hybrid approach of light stemmer + statistical trigram (0.8) stemmer is the most suitable stemming algorithm for an Arabic text categorization system, as shown in Figure 8.

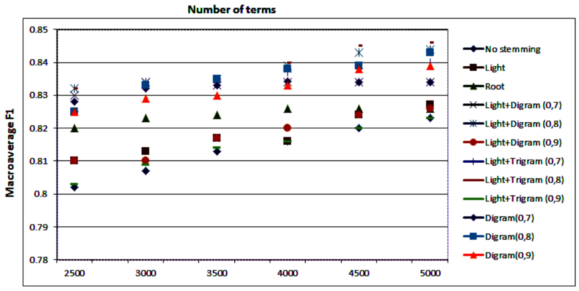


Figure 8: Effect of Stemming in Categorization Accuracy

The experiments shown in Figure 9 indicate that using any of those criteria separately gave results that are close to each other. It may be noticed as well that when information gain and odds ratio are used, most of the documents do not contain any term in the list of the selected terms. In other words, information gain and odds ratio select terms with rare appearance in the data set (i.e., terms with very low document frequency). This problem motivated the use of a hybrid approach combining document frequency thresholding and other criteria. Document frequency is used to remove rare terms and the other criteria to select terms from the remaining list.

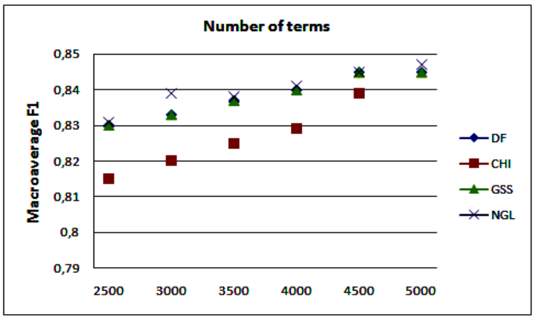


Figure 9: Effect of Term Selection on Categorization Accuracy

When this hybrid approach is used, the results show that using document frequency thresholding to remove terms with a document frequency of less than 2 then selecting terms that have a high score in information gain gives the highest results. When document frequency thresholding is used to remove terms with document frequency less than 3, the number of terms remains. There were about 4100 terms, so this hybrid method was tested to select a number of terms less than

4000 only. One drawback of this method is that for a few documents, when being represented as vectors, the weight of all their terms is zero (i.e., they contain no term from the selected list of terms).

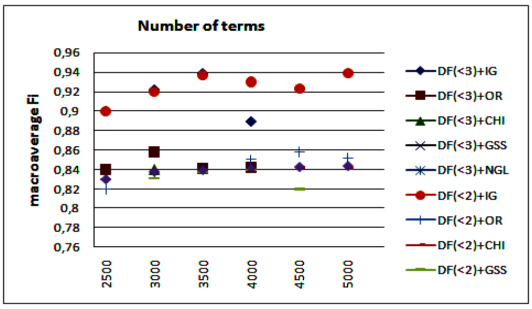


Figure 10: Accuracy of the Classification Process for Different Hybrid Term Selection Approaches

The suitable term weighting method is examined after examination of the best criteria for feature selection. In experiments, the hybrid approach of light and trigram stemming with similarity threshold (0.8) is used for the stemming phase, hybrid feature selection criteria of document frequency thresholding and information gain and Rocchio classifier with β=1.6 and γ=0.4 for classification. The results shown in Figure 11 indicate that normalized-tfidf is the preferable method for term weighting.

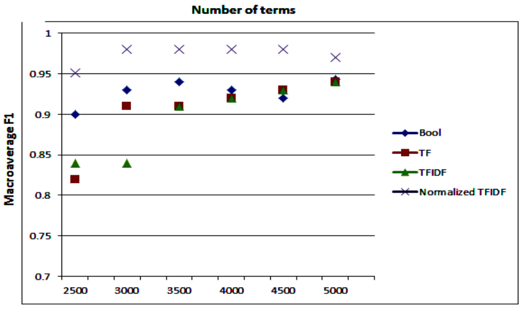


Figure 11: Effect of Term Weighting Method in Categorization Accuracy

Two non-parametric classification methods widely used with text categorization tasks were used. The Rocchio classifier shows the better results for our system. The hybrid approach of light and trigram stemming with similarity threshold (0.8) was used for the stemming phase, and the hybrid feature selection criteria of document frequency thresholding and information gain and normalized-tfidf method for term weighting. For time dependences score we selected 1622 documents, with every document containing 200 words, as shown in Table 7.

Table 7: Time in Hours Used for Classifying the Data Corpus Using K-NN and Rocchio Classifiers

No. of Terms	K-NN	Rocchio
324400	3:18:25	0:00:09

The results in Figure 12 and Table 7 show that the Rocchio classifier is superior to the k-NN classifier in both time and accuracy.

Different values for k (from k=1 to k=19), and for β and γ for the Rocchio classifier are used. The k-NN has many disadvantages in selecting the value for k. Moreover, the k-NN is not efficient. The Rocchio classifier is more efficient, as it classifies documents using centroids of every class instead of using every training document in the data corpus. The best values for β and γ are 1.6 and 0.4, respectively. The error rate is very small, which results from using a small data corpus. If the data corpus is huge enough, this may lead to an increase in the error, as shown in Figure 12.

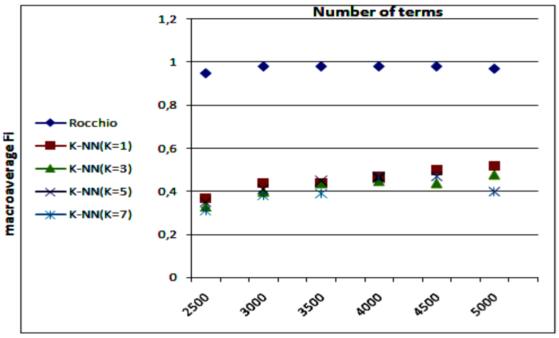


Figure 12: Effect of Classifier in Categorization Accuracy

Finally we made our own text collection. This collection consisted of 1250 documents and contained 140,000 words collected from the three main Sudanese newspapers (AlRayaam, Goon Sport, Alsudani Newspaper) during the period from August 2010 to September 2012. These documents cover six topics. Table 8 shows the number of documents for each topic. The documents had an average size of about 120 words before stemming and stop-word removal. The document represents the first paragraph of an article, which was chosen because it usually contains an abstract to the whole article.

Table 8: Number of Documents for Each Topic in the Text Collection

Topic	No. of Documents
Culture	233
Economy	233
International news	280
Local news	231
Religion	121
Sports	102

The evaluation groups consisted of human judges, six experts for each category. The human judges were professional experts in these fields. The result of the classifier output was checked by a critic for each category. Table 9 shows the results.

Table 9: Result of the Classifier Evaluation

Topic	Total Number of Texts in Category	Number of Texts that Belong to Topic	Proportion of Matching
Culture	233	227	97%
Economy	233	225	97%
International news	280	269	96%
Local news	231	220	95%
Religion	121	115	95%
Sports	102	97	95%
Average			**95,833**

CONCLUSIONS AND FUTURE WORK

Many stemming algorithms can be used in Arabic text preprocessing to reduce multiple forms of the word to one form (root or stem), but no complete stemmer for Arabic language gives high accuracy. In this paper, in order to improve the accuracy of stemming and therefore the accuracy of our TC system, a new and efficient algorithm for Arabic text stemming is proposed. The majority of problems related to Arabic categorization have been resolved, and an accuracy better than 96% has been achieved. Our future work will involve using the ATC to improve the performance of Arabic corpus in English language machine translation in choosing the correct meaning of the target language lexicon.

Also suggested is the possibility of a new user interface to increase manually the weight of some terms that the user thinks is important to determine the new category.

REFERENCES

1. M.Hadni, A.Lachkar, and S.Alaoui Ouatik. “A New and Efficient Stemming Technique for Arabic Text Categorization”. Conference Publications, IEEE, pp. 791-796, 2012.
2. M. Aljlayl and O. Frieder. “On Arabic search: improving the retrieval effectiveness via a light stemming approach”. In: ACM CIKM 2002 International Conference on Information and Knowledge Management, McLean, VA, USA, pp. 340-347. 2002.
3. S. Al-Fedaghi and F. Al-Anzi. “A new algorithm to generate Arabic root-pattern forms”. In: Proceedings of the 11th national Computer Conference and Exhibition, pp. 391-400, March 1989.
4. R. Al-Shalabi and M. Evens. “A computational morphology system for Arabic”. In: Workshop on Computational Approaches to Semitic Languages, COLING-ACL98, August 1998.
5. S. Khoja. “Stemming Arabic Text”. Lancaster, U.K., Computing Department, Lancaster University, 1999.
6. L. Larkey and M. E. Connell. “Arabic information retrieval at UMass in TREC-10”. Proceedings of TREC 2001, Gaithersburg: NIST, 2001.
7. L. Larkey, L. Ballesteros, and M. E. Connell. “Improving stemming for Arabic information retrieval: Light stemming and co-occurrence Analysis”. Proceedings of SIGIR’02, pp. 275–282, 2002.
8. A. Chen and F. Gey. “Building an Arabic stemmer for information retrieval”. In: Proceedings of the 11th Text Retrieval Conference (TREC 2002), National Institute of Standards and Technology, 2002.
9. F. Sebastiani. “A tutorial on automated text categorisation”. Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence, pp. 7-35, 1999.
10. K. Aas and L. Eikvil. “Text categorisation: A survey”, Technical report, Norwegian Computing Center, 1999.
11. S. Khoja. "Stemming Arabic text". Lancaster, U.K., Computing Department, Lancaster University, 1999.
12. Kraaij, W. “Viewing stemming as recall enhancement.” In: Proceedings of ACM SIGIR , pp. 40-48, 1996.
13. Khreisat, L. "Arabic text classification using N-gram frequency statistics a comparative study". Proceedings of the 2006 International Conference on Data Mining (pp. 78-82). Las Vegas, NV: USCCM.
14. S. H. Mustafa and Q. A. AI-Radaideh. "Using N-grams for Arabic text searching". Journal of the American Society for Information Science and Technology Volume 55, Issue 11, pp. 1002-1007, 2004.
15. S. AI-Fedaghi and F. AI-Anzi. "A new algorithm to generate Arabic root-pattern forms". In: Proceedings of the 11th national Computer Conference and Exhibition, pp. 391-400, March 1989.
16. R. A1-Shalabi and M. Evens. "A computational morphology system for Arabic". In: Workshop on Computational Approaches to Semitic Languages, COLING-ACL98, August 1998.

17. M. Aljlayl and O. Frieder. "On Arabic search: improving the retrieval effectiveness via a light stemming approach". In: ACM CIKM 2002 International Conference on Information and Knowledge Management, McLean, V A, USA, pp. 340-347, 2002.

18. L. Larkey, and M. E. Connell. "Arabic information retrieval at UMass in TREC-IO". Proceedings of TREC 2001, Gaithersburg: NIST. 2001.

19. A. Chen and F. Gey. "Building an Arabic stemmer for information retrieval". In: Proceedings of the l1th Text Retrieval Conference (TREC 2002), National Institute of Standards and Technology, 2002.

20. S. H. Mustafa and Q. A. Al-Radaideh. “Using N-grams for Arabic text searching”. Journal of the American Society for Information Science and Technology Volume 55, Issue 11, pp. 1002–1007, 2004.

21. T. Mitchell. “Machine Learning”. McGraw-Hill, New York. 1997.

22. M. Rogati and Y. Yang. “High-Performing Feature Selection for Text classification”. CIKM'02, ACM, 2002.

23. T. Liu, S. Liu, Z. Chen, and Wei-Ying Ma. “An evaluation on feature selection for text clustering”. Proceedings of the 12th International Conference (ICML 2003), Washington, DC, USA, pp. 488-495, 2003.

24. C. Liao, S. Alpha, and P. Dixon, "Feature preparation in text categorization". Australasian Data Mining Workshop in CEC 2003.

25. http://members.unine.ch/jacques.savoy/clef/index.html.

26. "molto-project.eu". molto-project.eu. Retrieved 2012-06-12.

27. M. M. Syiam, Z. T. Fayed, and M. B. Habib. “An Intelligent System For Arabic Text Categorization”. ISSN 0218-2157, Vol.6, pp. 1-15, 2006.

28. Motaz K. Saad and Wesam Ashour, “Arabic morphological tools for text mining”, pp. 112-117, European University of Lefke, Cyprus, 2010.

29. F. Sebastiani. “Machine learning in automated text categorization”. ACM Computing Surveys, volume 34 number 1, pp. 1-47, 2002.