Arabic Text Categorization

Rehab Duwairi
Department of Computer Information Systems, Jordan University of Science and Technology, Jordan

Abstract: *In this paper, we compare the performance of three classifiers for Arabic text categorization. In particular, the naïve Bayes, k-nearest-neighbors (knn), and distance-based classifiers were used. Unclassified documents were preprocessed by removing punctuation marks and stopwords. Each document is then represented as a vector of words (or of words and their frequencies as in the case of the naïve Bayes classifier). Stemming was used to reduce the dimensionality of feature vectors of documents. The accuracy of the classifiers is compared using recall, precision, error rate and fallout. The results of the experimentations that were carried out on an in-house collected Arabic text show that the naïve Bayes classifier outperforms the other two.*

Keywords: *Text categorization, naïve Bayes, knn, distance-based classifier, Arabic language.*

Received October 10, 2005; accepted March 1, 2006

1. Introduction

Text categorization is a process during which a set of documents are assigned labels; the set of labels are known in advance [1, 18, 23]. The advances in network technologies and the presence of the web have encouraged many individuals and organizations to make their data available online. Users nowadays are faced with finding their information needs quickly and efficiently. Therefore, text categorization becomes vital where it can be utilized in many applications such as classification of news stories, email-messages and web pages [1]. A wide range of text categorization algorithms have been developed such as naïve Bayes classifier [10], chain augmented naïve Bayes classifier [19], support vector machines [13], generalized discriminant analysis [17], k-nearest neighbors algorithm [13], optimized k-nearest neighbors using P-trees [20], neural networks [21] and generalized instance sets [15]. Most of these algorithms were tested against English text, however, some were applied to Chinese text such as the work reported in [13, 24].

Text categorization techniques that address Arabic text are rare in the literature. The work of Sawaf, Zaplo, and Ney [22] uses maximum entropy technique for Arabic document clustering. Document clustering, in their approach, started by randomly assigning documents to clusters. In subsequent iterations, documents were shifted from a cluster to another only if an improvement was gained. The algorithm terminated when no further improvements could be achieved. Therefore, their work is considered as unsupervised learning whereas the work reported here uses supervised learning for text classification. The work of El-Kourdi, Bensaid & Rachidi [7], by comparison, utilizes a naïve Bayes classifier to classify in-house collected Arabic documents. Finally, the work of Duwairi [6] describes a distance-based classifier for Arabic text categorization.

This paper compares the accuracy of three classifiers when used for Arabic text categorization. The first classifier is a distance-based one – where every category is represented as a vector of keywords that appear collectively in the training document examples that are known to belong to that category. Afterwards, unclassified documents are classified based on their closeness to category vectors. The category vectors are represented using the bag-of-word model [5], i. e., the model does not take word order into consideration. The second classifier is the classical naïve Bayes classifier; and the third one in the k-nearest-neighbor (knn) classifier.

The accuracy of the classifiers was tested using an in-house collected Arabic text corpus. The collected corpus contains 1000 documents that vary in length and writing styles and fall into 10 categories (100 documents per category). One half of these documents were used for training and the other half was used for testing. The documents were preprocessed by removing punctuation marks and stopwords. Word stemming was used as a filtering mechanism to reduce the number of features extracted from documents and therefore reducing the dimensionality of the feature vectors. The three classifiers performed very well as shown in the experimentation section of this paper, but the naïve Bayes classifier superceded the other two.

The rest of this paper is organized as follows. Section 2 provides an introduction to the Arabic language. Section 3 on the other hand, highlights the major functions that are usually done in preprocessing documents to prepare them for text categorization, with emphasis on the Arabic language. Section 4 describes the proposed classifiers in details. Experimentation and result analysis are presented in section 5. Finally,

section 6 draws the conclusions of this paper and outlines future work.

2. Arabic Language

The Arabic alphabet consists of the following 28 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 Arabic hamza (ء) which is considered as a letter by some Arabic linguistics. The letters (ا و ي) are vowels, the rest are consonants. Arabic is written from right to left. Arabic letters have different styles when appearing in a word depending on the letter position (beginning, middle or end of a word) and on whether the letter can be connected to its neighbor letters or not. For example, the letter (س) has the following styles (سـ) if it appears at the beginning of a word (such as the word ساعة which means clock); (ـسـ) if the letter appears in the middle of a word (such as the word يسجل which means register); (ـس) if the letter appears at the end of a word (such as the word حبس which means imprison). Finally, the letter (س) can appear as (س) if it appears at the end of a word but disconnected from the letter located to its right (such as the word درس which means study). Diacritics are signals placed below or above letters to double the letter in pronunciation or to act as a short vowel. Arabic diacritics include: Shada, dama, fathah, kasra, sukon, double dama, double fathah, double kasra. Different letter styles and diacritics make parsing Arabic text a non-trivial task (for a more thorough introduction to the Arabic language, please refer to Duwairi [6]).

3. Document Preprocessing

Document preprocessing involves the removal of punctuation marks, formatting tags, prepositions, pronouns, conjunctions and auxiliary verbs. The rest of words are retuned and are referred to as keywords or features. The number of these features is usually large for large documents and therefore some filtering can be applied to these features to reduce their number and to remove redundant features. Arabic is a very rich language, often a verb in its root pattern is augmented with prefixes, infixes and suffixes to refer to the time during which the event is occurred, whether the verb is plural or singular, and the sex of the participants in the verb. For example, the word (درس), which corresponds to the English verb *study*, can have several patterns for instance, if the prefix (ي) is added to the verb, it becomes (يدرس) and the time of the verb is present, done by one male. But if, on the other hand, the suffix (ا) is added to the verb, then it becomes (درسا) and means that the time of the event is past, number of participants is two and they are males.

The richness of the Arabic language increases the size of the feature vectors. Fortunately, Arabic has its built-in filtering mechanism – where words can be mapped into their root patterns using stemming. Root patterns in Arabic are divided into three, four, five, or six-letter patterns. Over 80% of Arabic words can be mapped into 3-letter root patterns.

Representing a word to its root pattern considerably reduces the number of words [9]. However, roots are semantically week in the since that several words can be mapped into the same root and thus loosing the sense (past, present, or future), the participants in the verb and so on. For example, the several forms of the verb go (درس) are reduced to the three-letter root pattern (درس). Despite this, mapping a word to its root reduces the dimension of document vectors.

Root extracting or stemming algorithms for Arabic text fall into two groups:

1. Algorithms that remove prefixes, infixes and suffixes from words and then map them into a set of predefined root patterns. In this style, a word may need to be scanned several times before it can be mapped into its root pattern. The works reported in [2, 8, 12, 14] fall into this category.
2. Algorithms that employ a letter weight and order scheme – where letters in a word are given weights and are assigned ranks or orders; and then the root is extracted by processing these assigned weights and ranks. The work reported in [3] is an example of an algorithm that falls in the second category.

This paper uses the work reported in [3] for root extraction. Al-Shalabi *et al.* [3] extracted word roots by assigning weights and ranks to the letters that constitute a word. Weights are real numbers in the range 0 to 5. The assignment of weights to letters was determined by extensive experimentation with Arabic text. The Arabic alphabet, according to Al-Shalabi, Kanaan & Al-Serhan [3], was divided into six groups as shown in Table 1.

Table 1. Assignments of letters to weights.

Arabic Letters	Weight
ا ة	5
ي ى ء	3.5
ت ى و	3
أ إ م ن	2
ل س ه	1
Rest of the Arabic Alphabet	0

The rank or order of letters in a word depends on the length of that word and on whether the word contains odd or even number of letters. Table 2, adapted from [3], shows the assignment of ranks to letters. N is the number of letters in a word.

After determination of the weight and rank of every letter in a word, letter weights are multiplied by the

letter rank. The three letters with the smallest product value constitute the root (read from right to left). Table 3, shows an example of using Al-Shalabi *et al.* [3] root extractor.

Table 2. Order of letters (adapted from [3]).

Letter Position From Right	Rank (If Word Length Is Even)	Rank (If Word Length Is Odd)
1	N	N
2	N - 1	N - 1
3	N - 2	N - 2
.	.	.
$\lceil N/2 \rceil$	N / 2 + 1	$\lceil N/2 \rceil$
$\lceil N/2 \rceil + 1$	N / 2 + 1 - 0.5	$\lceil N/2 \rceil + 1 - 1.5$
$\lceil N/2 \rceil + 2$	N / 2 + 2 - 0.5	$\lceil N/2 \rceil + 2 - 1.5$
$\lceil N/2 \rceil + 3$	N / 2 + 3 - 0.5	$\lceil N/2 \rceil + 3 - 1.5$
.	.	.
N	N - 0.5	N - 1.5

Table 3. An Example of using the root extractor described in [3].

Word	المحافظة							
Letters	ة	ظ	ف	ا	ح	م	ل	ا
Weight	5	0	0	5	0	2	1	5
Rank	7.5	6.5	5.5	4.5	5	6	7	8
Product	37.5	0	0	22.5	0	12	7	40
Root	حفظ							

Al-Shalabi *et al.* [3] work handles only three-letter roots. Fortunately, over 80% of Arabic words have three-letter roots and the words that are commonly used in Arabic writings have three-letter roots. The accuracy of their root extractor was reported to be over 90%.

4. Classifiers

This section explains how the three classifiers were built and used for categorizing Arabic documents.

4.1. Distance-Based Classifier

Assume that $\Delta = \{TD_1, TD_2, \ldots, TD_n\}$ is a set of document examples that belong to category C_i, where n is the number of documents in that category. To build the feature vector of category C_i, every $TD_j \in \Delta$ is processed in the following manner:

1. Punctuation marks and stopwords are removed from the document. Even this standard step in processing documents has its special flavor when dealing with Arabic text. For example, the conjunction letter waw (و) which corresponds to *and* in English can be confused with words that start with the letter (و). Consider the word (وصل) which means *arrived* in English; this word starts with the letter (و) but this letter is an integral part of the word. However, the (و) in the word (وذهب), which corresponds to "*and gone*" in English, is extra and in fact represents a conjunction letter. One way to eliminate this confusion is to map a word to its root pattern where extra letters are removed from words.
2. Roots of the remaining words are extracted to reduce the dimension of category-specific feature vectors.
3. The extracted roots are added to the feature vector of category C_i.

The above process is repeated for every category. Thus, the feature vector of a category consists of keywords, in root format, which are present in the union of the training documents that are known to belong to this category. This means that if a word appears one or more times in a document that is known to belong to C_i, then that word is added to the feature vector of category C_i. At the end of the training phase, the m-dimensional space, against which unclassified documents are compared, is created. It consists of a feature vector per category. m is the number of categories.

To categorize a new document, such as X, it is preprocessed by removing punctuation marks and stopwords, and then by extracting the roots of the remaining keywords. After that, the feature vector of X is compared with the feature vectors of the categories one at a time. The Dice measure (as defined in equation 1 below [11] was used to compute the similarity. Once the feature vector of X has been compared to the feature vectors of all categories, it is assigned to the category with the maximum similarity. i. e., the category of X is the category Y where $sim_{Dice}(F_X, F_Y)$ is maximum.

$$sim_{Dice}(F_X, F_Y) = \frac{2 \times | F_X \cap F_Y |}{| F_X | + | F_Y |} \quad (1)$$

Where F_X and F_Y are feature vectors of Document X and category Y, respectively and | | is the number of keywords in the feature vector.

4.2. Knn Classifier

The knn classifier does not carry out any off-line learning to generate category-specific knowledge during its learning phase; and therefore has a very fast training time; by comparison, it performs on-line scoring to find the k nearest training documents to a test document and makes its prediction based on the statistical presumption that documents in the same category have similar features during its testing phase. The knn classifier scans its document space or database once for every test document and therefore it has a slow and costly testing phase that requires multiple database scans. The distance-based classifier, described in section 4.1 resolves this deficiency in the knn classifier by learning category specific features during the learning phase.

In this work, we have implemented a knn classifier for Arabic text. Training and test documents are

represented as feature vectors that consist of words in stemmed format (i. e., the bag of words model was used). Training documents form the training space and new documents are categorized based on their closeness to documents in this space. The Dice function, as described in formula (1), was used as closeness measure. We experimented with different values of k. In particular, k was 10 in the first experiment, 20 in the second, 50 in the third and 100 in the fourth. Majority voting was used to determine the category of an unclassified document.

4.3. Naïve Bayes Classifier

Bayes classifiers predict the category (c_j) of a document (d_i) by the following Bayes' rule [4]:

$$P(c_j \mid d_i) = \frac{P(c_j) \times P(d_i \mid c_j)}{P(d_i)} \quad (2)$$

Where c_j is a category and d_i is a document. Usually, in text classification, a document (d_i) is represented as a vector of keywords (v_1, v_2, …, v_t). Therefore calculating the probability P ($c_j \mid d_i$) is not trivial. To simplify this calculation, naïve Bayes classifier assumes that the features v_1, v_2, …, v_t are independent given the category label c_j. This assumption may be unrealistic but it greatly simplifies the computation of rule (2) above. Given the independence assumption, rule (2) can be rewritten as:

$$P(c_j \mid d_i) = P(c_j) \times \frac{\Pi_{k=1}^{t} P(v_k \mid c_j)}{P(d_i)} \quad (3)$$

Based on rule (3) a maximum posterior classifier can be constructed by seeking the optimal category that maximizes the posterior P ($c_j \mid d_i$):

$$c_{optimal} = \arg\max_{c_j \in C} \{P(c_j \mid d_i)\} \quad (4)$$

$$= \arg\max_{c_j \in C} \left\{ P(c_j) \times \frac{\Pi_{k=1}^{t} P(v_k \mid c_j)}{P(d_i)} \right\}$$

$$= \arg\max_{c_j \in C} \left\{ P(c_j) \times \Pi_{k=1}^{t} P(v_k \mid c_j) \right\} \quad (5)$$

Note that P (d_i) is constant for all categories.

P ($v_k \mid c_j$) is calculated using:

$$P(v_k \mid c_j) = \frac{f}{F} \quad (6)$$

f is the frequency of a word (v_k) in the test document. F is the number of documents in which word v_k has appeared in. To handle the cases where a word does not appear in a document (i. e., to a void zero probability) *add one Laplace* smoothing is used and therefore rule (6) becomes:

$$P(v_k \mid c_j) = \frac{f + 1}{F + W_j} \quad (7)$$

and W_j equals number of training documents in the category c_j.

5. Experimentation and Result Analysis

To assess the accuracy of the proposed classifiers, Arabic text corpus was collected from online magazines and newspapers. 1000 documents that vary in length and writing styles were collected. These documents fall into 10 pre-defined categories. Every category contains 100 documents. The set of predefined categories include: Sports, economic, Internet, art, animals, technology, plants, religion, politics, medicine, and plants. Two individuals manually categorized the collected documents; and every document was assigned to only one category, whenever a document was found to belong to more than one category, it was assigned to the category with the maximum likelihood according to human categorizer's judgment. For every category, 50 documents were randomly specified and used for training and the remaining 50 were used for testing.

Accuracy of the classifier is expressed in terms of recall, precision, fallout and error rate as described in [16]. Figure 1 shows recall, precision, error rate and fault values for different values.

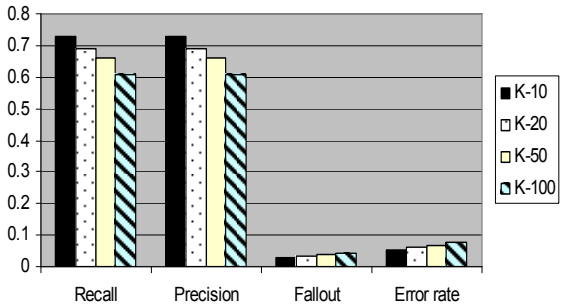

Figure 1. Recall, precision, error rate, and fallout values for k = 10, 50, 30 & 100.

Table 4 shows recall, precision, fallout and error-rate for every category when the distance based classifier was used. It also shows the *micro* average of these values for all categories. As it can be seen from Table 4, recall reaches its heights value (0.98) for the economic category, while the lowest value of recall was (0.22) for the Internet category. The second lowest value for recall was (0.38) for the technology category. When the classifier's output was reexamined, it was noticed that a large percentage of the misclassified documents in the Internet category were categorized under the technology category and vice versa. This is due to the fact that documents that belong to the Internet category and those which belong to the technology category share many common words. We assume that a hierarchical classifier is more appropriate as Internet would be placed as sub-category of technology. Precision reaches its heights value (1.00) for the politics and sport categories; and

its lowest value for the Art category (0.3925). The rest of the table is self explanatory.

Table 4. Recall, precision, error rate and fallout values for categories (the distance-based classifier).

Category	Recall	Precision	Fallout	Error Rate
Animal	0.6600	0.9167	0.0067	0.0400
Art	0.8400	0.3925	0.1444	0.1460
Economic	0.9800	0.3952	0.1667	0.1520
Internet	0.2200	0.6471	0.0133	0.0900
Medicine	0.6800	0.7727	0.0222	0.0520
Plant	0.5800	0.9355	0.0044	0.0460
Politics	0.4400	1.0000	0.0000	0.0560
Religion	0.5800	0.8529	0.0111	0.0520
Sport	0.9200	1.0000	0.0000	0.0080
Technology	0.3800	0.4872	0.0444	0.1020
Micro average	0.6280	0.7400	0.0413	0.0744

Figure 2 shows the values of recall and precision for the categories of technology, religion, politics, medicine and plants when the naïve Bayes classifier was used. The technology category achieves the highest precision (1.0); while the politics category achieves the lowest precision (0.67). Recall, by comparison, reaches it highest value for the religion category (1.0) and its lowest value for the technology category (0.36).

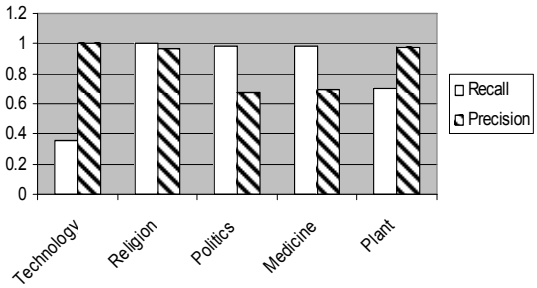

Figure 2. Recall & precision for five categories using the naïve Bayes classifier.

Figure 3, on the other hand depicts the values of recall, precision, error rate and fallout for the three classifiers at hand. It is clear that the naïve Bayes classifier has the best accuracy. Then the knn classifier comes in the second place (when k equals 50) and the worst classifier for this dataset was the distance-based classifier.

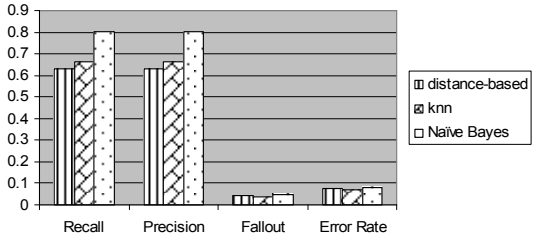

Figure 3. Recall, precision, fallout and error rate for distance-based, knn and naïve Bayes classifiers.

6. Conclusions and Future Work

This paper has investigated text categorization for Arabic language. It uses three classifiers, namely: The knn, naïve Bayes and distance-based classifiers. The knn and naïve Bayes classifiers agree on their definition and usage with what is available in the literature. The distance-based classifier, on the other hand, collects information about the categories under consideration in its learning phase. A category is described by a set of keywords that form its feature vector which is constructed by scanning a set of previously categorized documents (called positive examples).

Unclassified documents were preprocessed by removing stopwords and punctuation marks. The rest of words were stemmed and stored in feature vectors. Every test document has its own feature vector. For the knn and distance-based classifiers, the vectors were bag of words. For the naïve Bayes classifier, the words, their frequency and number of documents in which these words appeared (inverse document frequency) were also stored in the feature vectors.

Processing Arabic text was a nontrivial task due to the richness of the language. The sizes of the feature vectors can grow very large for large documents and therefore stemming was used as a filtering mechanism. The accuracy of the classifiers was measured using recall, precision, fallout and error rate. The three classifiers were tested using in-house collected Arabic text. Unclassified documents were categorized using the three classifiers in turn. The results showed that the performance of the naïve Bayes classifier outperformed the other two classifiers.

We plan to continue working with Arabic text categorization as this area is not widely explored in the literature. We plan to investigate the suitability of filtering mechanisms such as the information gain of words or word clustering for Arabic text categorization.

Acknowledgements

The author would like to thank Mr. Iyad Al-Azzam and Mr. Mohammed Al-Refai for their help. The author also thanks the anonymous reviewers whom their helpful comments helped in shaping this final draft.

References

[1] Al-Mubaid H., "Machine Learning Approach for Automatic Text Categorization," *in Proceedings of the International Arab Conference on Information Technology (ACIT'2003)*, Alexandria, Egypt, pp. 834-840, 2003.

[2] Al-Shalabi R. and Evans M., "A Computational Morphology System for Arabic," *in Proceedings of Computational*

Approaches to Semitic Languages Workshop (COLING'98), Montreal, Canada, pp. 58-65, 1998.

[3] Al-Shalabi R., Kanaan G., and Al-Serhan H., "New Approach for Extracting Arabic Roots," *in Proceedings of the International Arab Conference on Information Technology (ACIT'2003)*, Alexandria, Egypt, pp. 42-59, 2003.

[4] Duda R. and Har P., *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[5] Dumais S., Plat J., Heckerman D., and Sahami M., "Inductive Learning Algorithms and Representations for Text Categorization," *in Proceedings of the 7th ACM International Conference on Information and Knowledge Management*, Bethesda, USA, pp. 148-155, 1998.

[6] Duwairi R., "Machine Learning for Arabic Text Categorization," *Journal of the American Society for Information Science and Technology (JASIST)*, vol. 57, no. 8, pp. 1005-1010, 2005.

[7] El-Kourdi M., Bensaid A., and Rachidi T., "Automatic Arabic Documents Categorization Based on the Naïve Bayes Algorithm," *in Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages (COLING-2004)*, University of Geneva, Geneva, Switzerland, 2004.

[8] El-Sadany T. A. and Hashish M. A., "An Arabic Morphological System," *IBM Systems Journal*, vol. 28, no. 4, pp. 600-612, 1989.

[9] Eldos T., "Arabic Text Data Mining: A Root Based Hierarchical Indexing Model," *International Journal of Modeling and Simulation*, vol. 23, no. 3, pp. 158-166, 2003.

[10] Eyheramendy S., Lewis D., and Madiagn D., "On the Naïve Bayes Model for Text Categorization," *in Proceedings of the Artificial Intelligence and Statistics Conference*, Key West, FL, USA, 2003.

[11] Ganesan P., Garcia-Molina H., and Widom J., "Exploiting Hierarchical Domain Structure to Compute Similarity," *ACM Transactions on Information Systems*, vol. 21, no. 1, pp. 64-93, 2003.

[12] Gheith M. and El-Sadany T., "Arabic Morphological Analyzer on a Personal Computer," *in Proceedings of the Arabic Morphology Workshop*, Stanford University, California, USA, pp. 55-65, 1987.

[13] He J. I., Tan A., and Tan C., "On Machine Learning Methods for Chinese Document Categorization," *Applied Intelligence*, vol. 18, pp. 311-322, 2003.

[14] Hilal Y., "Automatic Processing of Arabic Language and Application [In Arabic]," *in Proceedings of the 1st Kuwaiti Computer Conference*, Kuwait, pp. 145-171, 1989.

[15] Lam W., "Automatic Textual Document Categorization Based on Generalized Instance Sets and Meta-Model," *IEEE Transactions on Pattern Analysis and Machine Learning Intelligence*, vol. 25, no. 5, pp. 628-633, 2003.

[16] Lewis D., "Evaluating and Optimizing Autonomous Text Classification Systems," *in Proceedings of the 18th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, Washington, USA, pp. 246-254, 1995.

[17] Li T., Zhur S., and Ogihara M., "Efficient Multi-Way Text Categorization Via Generalized Discriminant Analysis," *in Proceedings of the CIKM'03 Conference*, New Orleans, Louisiana, USA, pp. 317-324, November 2003.

[18] Meretakis D., Fragoudis D., Lu H., and Likothanassis S., "Scalable Association-Based Text Categorization," *in Proceedings of the 9th ACM International Conference on Information and Knowledge Management*, McLean, VA, USA, pp. 5-11, 2000.

[19] Peng F., Schuurmans D., and Wang S., "Augmenting Naïve Bayes Classifiers With Statistical Language Models," *Information Retrieval*, vol. 7, pp. 317-345, 2004.

[20] Rahal I. and Perrizo W., "An Optimized Approach for KNN Text Categorization Using P-Trees," *in Proceedings of the ACM Symposium on Applied Computing*, Nicosia, Cyprus, pp. 613-617, 2004.

[21] Ruiz M. and Srinivasan P., "Hierarchal Neural Networks for Text Categorization," *in Proceedings of the SIGIR'99 Conference*, Berkeley, California, USA, pp. 281-282, 1999.

[22] Sawaf H., Zaplo J., and Ney H., "Statistical Classification Methods for Arabic News Articles," *in Proceedings of the Arabic Natural Language Processing Workshop (ACL2001)*, Toulouse, France, pp. 1-6, 2001.

[23] Sebastiani F., "Machine Learning in Automated Text Learning," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1-47, 2002.

[24] Tsay J. J. and Wang J. D., "Improving Linear Classifier for Chinese Text Categorization," *Information Processing and Management*, vol. 40, no. 2, pp. 223-237, 2004.

Rehab Duwairi received her BSc degree in computer science from Yarmouk University, Jordan, in 1989, MSc and PhD degrees in computer science from the University of Wales, Cardiff, UK, in 1994 and 1997, respectively. In 1997, she joined Jordan University of Science and Technology, Jordan, where she is currently working as an associate professor of computer science. Her research interests include object oriented databases, data mining, semantic integration of structured and unstructured data, and text categorization.