IJICTR International Journal of Information & Communication Technology Research

Volume 5- Number 3- Summer 2013

Quality Improvement of Information Retrieval System Results by Document Classification

Mojgan Farhoodi
Information Technology Faculty
CyberSpace Research Institute
Tehran, Iran
farhoodi@itrc.ac.ir

Saeed Shiry Ghidary
Computer Engineering & IT Dept.
Amirkabir University of Technology
Tehran, Iran
shiry@ce.aut.ac.ir

Alireza Yari
Information Technology Faculty
CyberSpaceResearch Institute
Tehran, Iran
a_yari@itrc.ac.ir

Received: September 16, 2012- Accepted: June 2, 2013

Abstract—**In traditional search engines, the most common way to show results for a query is to list documents in order of their computed relevance to the query. However, the ranking is independent of the topic of the document;so the results of different topics are not grouped together. In this situation, the user must scroll though many irrelevant results until his desired information need is found. One solution is to organize search results via classification.**

Many researchers have shown that classifying web pages can improve a search engine's ranking of results. Intuitively results should be more relevant when they match the class of a query. In this paper, we present a simple framework for classification-enhanced ranking that uses query class in combination with the classification of web pages to derive a class distribution for the query. In this regard, we propose a hybrid IR search strategy that begins with a 3-gram classification-based strategy and reverts to a ranked-list strategy if the user doesn't find the target document in selected class.The experiment results on Hamshahri corpus show satisfactory results.

Keywords-Information Retrieval; Hamshahri; Ranking; Classification; SVM; KNN; N-gram language modeling; Smoothing methods.

I. INTRODUCTION

Web-based search engines enable fast and easy access to documents that relate to users' information needs. These engines present search results in a ranked ordered list. The ranks of the documents are determined by their relevance to the corresponding query. This relevance measure depends predominantly on the users' ability to suitably describe their information need as a query text. Unfortunately, most queries are short, and unconscious assumptions are made regarding the context of query terms, making the query ambiguous or vague. This leads to a low precision in the retrieved results and users are forced to manually sort through the list to find relevant documents. This would be unproblematic if the users could easily separate irrelevant documents from the relevant ones. However, the current presentation style of ranked lists used by most search engines does not make it easy to do so. Document ranking becomes virtually obsolete when documents lower in the list are more relevant to the user than the ones with higher ranks. Therefore, a major challenge for efficient web search is to make search results helpful to the user

even if the query is poorly formulated. For example, the users issuing a Web query "*apple*" might expect to see web pages related to the fruit apple, or they may prefer to see products or news related to the computer company. While the user is only interested in one topic, it is not possible for the search engine to know which topic is relevant based on the query alone. Moreover, the standard ranking of the documents in the result set is independent of the topic. Thus, the rank-ordered result set has an arbitrary topic ordering. Referring to the "*apple*" example, this means that a user must scroll through a ranked list in which many documents are not relevant.

Automatic classification of web pages into relevant categories is the current research topic which helps the search engine to get relevant results. At query time the user is asked to specify one or more desired categories so that only the results in those categories are returned, or the search engine returns a list of categories under which the pages would fall. This approach works when the user is looking for a known item [1]. In such a case, it is not difficult to specify the preferred categories. However, there are situations in which the user is less certain about what documents will match, for which the above approach does not help much. Search results are usually presented in a ranked list. However, presenting categorized, or clustered, results could be more useful to users.

In this paper we reviewed how grouping strategy can assist the user by organizing the documents in the result set into groups, all documents within a group referring to a common topic. Intuitively, for the case that users may be satisfied with one relevant result, we would expect grouping to substantially reduce search time, where search time is measured by the number of documents a user must examine before finding the desired document [2]. For the query "apple", a user might be shown two distinct groups of documents: one referring to the fruit appleand the other referring to the computer company. A user can immediately ignore the non-relevant topic and focus his attention to only the relevant topic. For this simple example, this grouping may, on average, halve the number of documents the user must examine.

In this paper, we attempt to quantify the benefits of grouping documents based on classification. However, we note that our experimental approach may be applied to any method of grouping documents.

The remainder of the paper is organized as follows. Section 2 describes search result classification. Then Section 3 reviews related work. Section 4 explains the proposed classification-based IR system. Section 5 is an overview of some important classification methods. Section 6 provides experiment and experimental results are discussed in section 7. Finally Section 8 provides the conclusion and outline future research direction.

II. Search Result Classification

Many main commercial search engines present search results in a ranked list, referred to as the result set. The ranks of thedocuments are determined by the relevance to the corresponding query. For many queries, the result set includes documents on a variety of topics, rather than a single topic. This variation is often due to the fact that a typical query contains about only two to three terms, which is insufficient to locate the desired information unambiguously.

These results in a low precision in the retrieved results and forces the user to sift through the list to find the relevant documents, in particular for informational queries. It will be ideal if current search engines can separate the irrelevant documents from the relevant documents. However, the ranked list interface of search engines cannot solve this issue even ranking algorithms become perfect since users could have different information needs when they issue the same query. Moreover, where users may be satisfied with one relevant result, i.e., for navigational queries or queries in a question answering system, result diversification can be adopted, which will benefit the user population overall within the ranked list interface. Search result classification has been proposed as a solution to this problem.

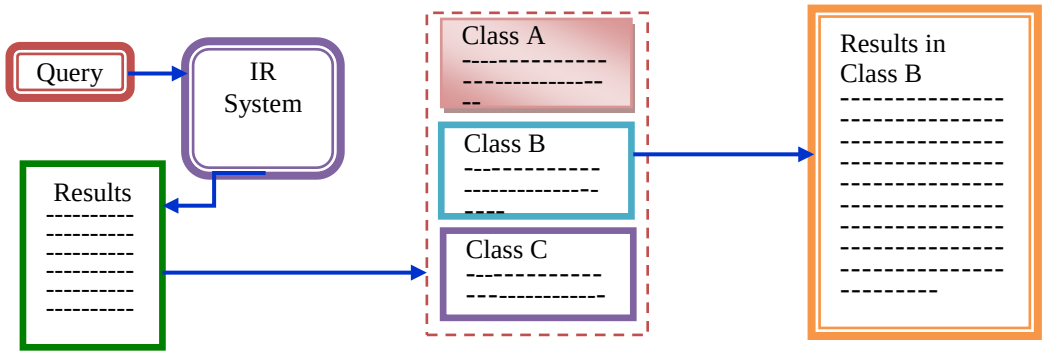
III. Related Work

Improving the quality of search results by information retrieval systems is one of the important issues that many researchers are working in this fields [4]. In recent decades, much activity has been done in this regards that one of them is search results categorization.

This concept of grouping search results has been discussed in [5], where it was shown that relevant documents tend to be more similar to each other rather than to non-relevant documents, indicating that relevant documents can be grouped into one category. The two main methods of grouping results are clustering and classification [6, 7].

Clustering methods typically extracts key phrases from the search results for grouping purposes and attach to each group a candidate cluster label [8, 9]. The search results are treated as a bag of words/phrases, which are ranked according to the statistical features that have been found.

Classification uses predefined category labels that are more meaningful to users than generated labels. [10] developed a user interface that organizes web search results into hierarchical categories. He used text classification algorithms to automatically classify arbitrary search results into an existing category structure on-the-fly. [11] proposed an approach to presenting web search results that supports personalization, taking into consideration users' perspective. They developed a post-retrieval algorithm which uses document classification techniques to organize search results into a meaningful hierarchy of topics. In [12], a novel algorithm known as deep classifier was proposed to classify the search results into detailed hierarchical categories. Given the search results in response to a query, the algorithm first prunes a wide-ranged hierarchy into a narrow one with the help of some Web directories. [13] suggested a search result classification system based on the degree of suitability for specialists. Their system classifies documents based on whether a document is "for specialists" or "non for specialists". This classification is done by ranking each document using the density of



a) *Standard IR model* b) *Classification-based IR model* c)Results in Class b

Figure 1. Conceptual framework of classification-based IR system

keywords and by a threshold. [14] described a search interface that combines context-free syntactic search with context-sensitive search guided by classification, and that can potentially cover most of the Web. This allows the user to submit focused queries, thus improving on the precision of search engines while improving on the coverage of net directories.

[2] suggested research method based on classification that can improve search time in comparison to the traditional list-based search, where documents within a category are ranked according to their relative ranking in the original search engine results list.

In [15], document classification is done in indexing time but [2] categorized the returned documents in searchtime. Also later method is less efficient because the search and classification of results will be conducted at the same time; but most users are more satisfied with this method.

Most previous research in this area has focused on evaluating the grouping of search results versus the traditional list-based method and has not considered a hybrid model; even when a hybrid model is implemented [2], they don't consider user query classification. In this paper, in addition to considering the document class, we involve the user query class to search results categorization.

IV. Proposed Classification-Based IR System

A. Conceptual View

The architecture and ranking method of a classification-based IR system that we have been developing in this section described in Fig.1. In most standard information retrieval systems such as Google, Bing and Yahoo!, search engines gives a user query and then returns a ranked list of documents as the result set. Web search engines satisfy this assumption.

In our proposed system after giving a ranked set of documents, it is necessary the results classified into their respective classes. Figure 1 provides a conceptual view of the classification-based information retrieval system we have developed. Figure 1.a depicts the ranked set of documents provided by a standard IR system. Our system classifies these documents into a number of classes, ranks the classes and then displays a ranked list of classes to the user, as depicted in Figure 1.b. When a user clicks on a particular class, the ranked set of documents in this class is then displayed to the user, as shown in Figure 1.c.

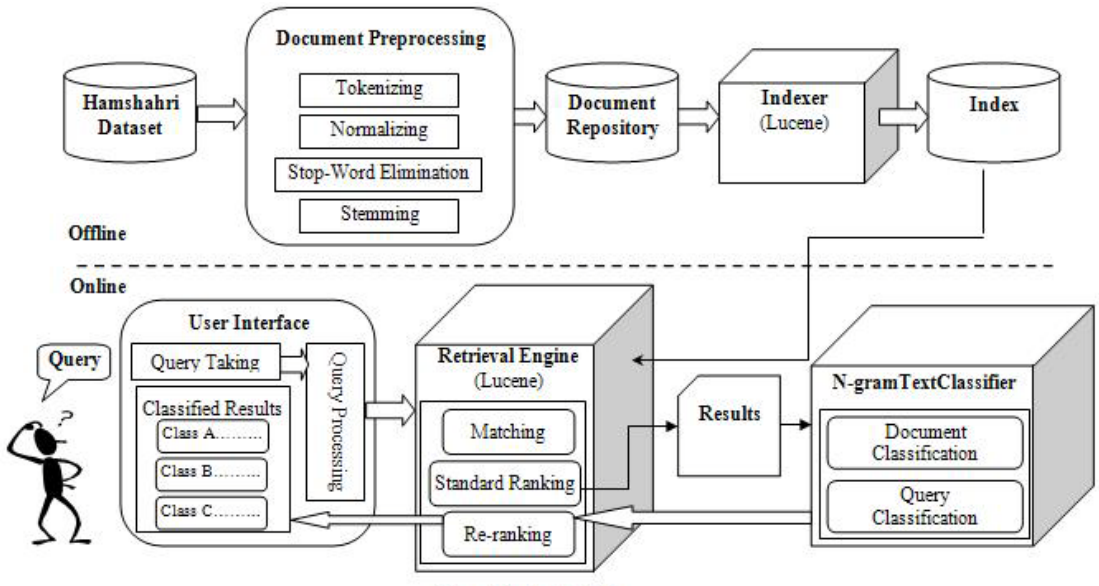


Figure 2. System Architecture

B. Architecture

B.1. Overview

As shown in Figure 2, the proposed system architecture is divided into two parts, offline and online.

- Offline: Hamshahri [16] news have been stored in XML format; so at first, by using a XML parser, useful data are extracted from related tags and each news saved in a separate text file. Then lingual preprocessing is applied to all text files and relevant information such as document number, news text and etc are stored in the index.

- Online: In this part, after taking the user query and applying the lingual preprocessing to it, the processed query is sent to Lucene retrieval engine. Then the retrieved results passed to an *n*-gram classifier. Classifier assigns a predefined class to each document. Retrieval engine grouped the documents according to their assigned class and ranks the classes by equation1.

B.2. Class Rank

After |*D*| documents returned by standard IR system as the result set,classification process is applied on the original result set and the documents are divided into |C| top-level classes. Each class, c_i, consists of a set of documents, $d_{i,j}$, where $1 \leq j \leq |c_i|$, and $|c_i|$ denotes the number of documents in a class, c_i.

Because of the documents grouped into some classes, it is necessary to rank these classes. We proposed thateach class, c_i, is assigned a score according to equation1:

$$Rank(c_i) = \frac{|c_i|}{|D|} * P(q \in c_i) * Avg(\sum_{j=1}^{|c_i|} score(d_j)) \quad (1)$$

In our case, the rank of each class is depends on some parameters as follow:

- The number of returned documents in c_i.
- The probability of user query belongs toc_i.
- The average of scores all documents in c_i. Score(d_j) is the score of document(d_j) that is as output of the standard IR system.

For notational convenience, we assume the classes to be ordered such that class c_i has rank *i*.

The result is illustrated in Figure 3. On the left part of Figure 3 is the retrieved results set, the results are classified into 3 classes as shown in the middle part. The ranking is illustrated in the right panel and it is ordered by the class score, which is determined by the Equation1.

B.3. Document Rank

After classes were ranked, the documents, $d_{i,j}$, within each class, c_i, must be sorted. In the result set that returned by standard IR system, each document has a score. We use these scores for sorting the documents in each class.

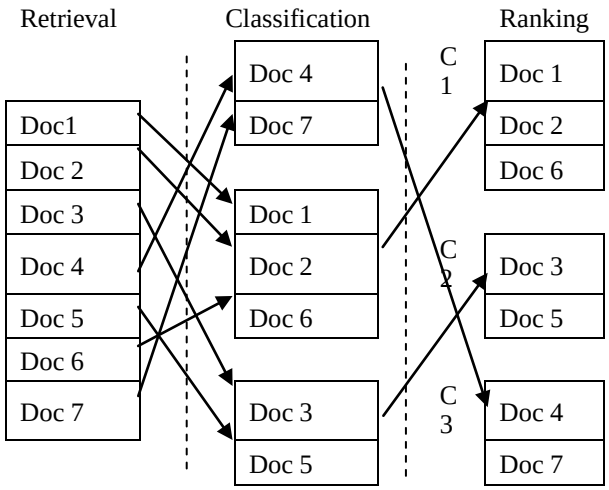


Figure 3. The Illustration of Ranks

So, documents within the class are then ranked according to their scores, the highest score being ranked first, as in traditional search engines. For notational convenience, we assume the documents to be ordered such that document $d_{i,j}$ has rank *j* in class c_i.

B.3. Out-Class Document Rank

Sometimes, the document that the user is looking for, doesn't exist in the selected class and then the user must perform additional work. We refer to this situation as the "Out-Class". After failing to obtain target document in the selected class, the user may choose one of the following options [2]:

a) *Scroll* through the classes and the documents in each class in rank order,
or
b) *Revert*to the standard IR display and sequentially scroll through the ranked result set. In this case, the documents are ranked again.In this paper we propose equation 2 for re-ranking the documents.

$$Rank(d_j) = (\sum_{i=1}^{i=9} P(q \epsilon c_i) * P(d_j \epsilon c_i)) * score(j) \quad (2)$$

Rank of each document is dependent on the following parameters:

- The probability of the document belongs to c_i.
- The probability of user query belongs to c_i.
- The score of document. Score(d_j) is the score of document(d_j) that is as output of the standard IR system.

It should be noted that the out-class/revert rank is a hybrid search strategy that begins with a classification-based strategy and reverts to a ranked-list strategy if the user doesn't find the target document in the selected class. This hybrid strategy is different from the presentation in cluster-based search engines, where the user is presented with the ranked listing in a main window and the clusters in another [2].

V. CLASSIFICATION

We have not talked until now about how classification is performed. In order to classifying Persian text documents, it is necessary to apply some lingual preprocessing, such as: Text segmentation, normalization, tokenization, elimination of stop words and word stemming.

Text segmentation, one of the primary activities in text preprocessing; is the process of recognizing boundaries of text constituents, such as paragraphs, sentences, phrases and words. Word segmentation also known as tokenization focuses on recognizing word boundary delimiters, punctuation marks, written forms of alphabet and affixes. The developed tokenizer determines words boundaries as explained in [17].

To achieve the goal of Persian text classification, after removing the xml tags, whole words of each document are extracted by using a Persian tokenizer. Then in word segmentation process, all stop words are eliminated from the extracted tokens. The main reason for eliminating stop words is that they frequently occur in all corpora, and they usually don't have any added value in the process of text classification.

There are some letters such as 'ي' (*i*) and 'ك' (*k*) for which we have two Unicode (one for Persian and one for Arabic). In Persian text both are used. In normalizing step, we have to unify their occurrences. In this paper, the encoding of all text files is converted to UTF-8.

In addition, there are some imported sounds such as "Tanwin" and "Hamza" from Arabic which we use in some imported words in Persian. These words may be written in some different forms. For example 'پاییز' and 'پائیز' are forms of writing the word 'fall', so one of the lingual preprocessing is unification of these words. In the last step of lingual preprocessing, suffixes and prefixes of each word is removed and stems of word extracted by Persian stemmer.

In the experiments, we assume three cases for document classification:

In the first case, we assume classification is performed based on a k-nearest neighbor (KNN) classifier [18]. K-Nearest Neighbor is one of the most popular algorithms for text categorization. To classify a new document, the system finds the k nearest neighbors among the training documents, and uses the categories of the k nearest neighbors to weight the category candidates. One of the drawbacks of KNN algorithm is its efficiency, as it needs to compare a test document with all samples in the training set. In addition, the performance of this algorithm greatly depends on two factors, that is, a suitable similarity function and an appropriate value for the parameter k. Compared to other text categorization methods such as Bayesian classifier, KNN does not rely on prior probabilities, and it is computationally efficient [19].

In the second case, we assume we have an SVM based classifier on 9 top level categories of the Hamshahri dataset. SVM is a popular and traditional approach to text categorization. The idea of SVM is that if a distribution of training examples is not linearly separable, these examples are mapped into another space where their distribution is linearly separable, as illustrated in the left side of figure 4. SVM optimizes the weights of the inner products of training examples and its input vector, called Lagrange multipliers [20], instead of those of its input vector, itself, as its learning process. It defines two hyper-planes as a boundary of two classes with a maximal margin, as illustrated in the right side of figure 4.

After that a separate lexicon prepared for each category to determine common words in that class. These lexicons include terms and their frequency in the related category. Also a general lexicon is made for all of words in the whole of documents. In this paper, we use TF-IDF [21] (a kind of augmented DF) as a feature selection criterion for choosing the common words in each lexicon.

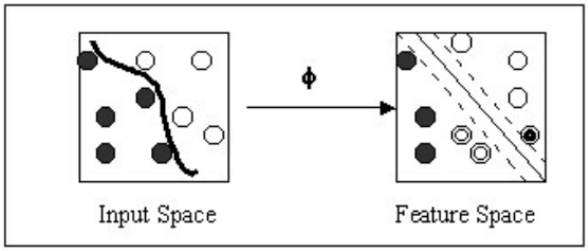


Figure 4. Mapping vector space in SVM

In order to representing the documents, we use a vector for each document with m elements where m denotes the number of features which are mostly the text words; also we use TFCRF method for weighting the words in each document. The TFCRF [21] considers both the distribution of the feature within different documents and its distribution within different categories for weighting a feature.

Finally we have a matrix that number of rows equal to the number of sample document in dataset and number of column is equal to the total number of words in general vocabulary. The value of each cell represents the weight of related word in related document. If a term doesn't exist in a document, the related value in matrix will be zero. We apply our selected machine learning algorithm on this matrix, as described in the next section

In the third case, we assume we have language modeling classifier to predict the class of each document [22]. The simplest and most successful basis for language modeling is the *n*-gram model. Note that by the chain rule of probability we can write the probability of any sequence as [23]:

$$P(w_1 w_2 \dots w_T) = \prod_{i=1}^{T} P(w_i | w_1 \dots w_i w_{i-1}) \qquad (3)$$

An *n*-gram model approximates this probability by assuming that the only words relevant to predicting $P(w_i|w_1 \dots w_i w_{i-1})$ are the previous *n-1* words; that is, it assumes the Markov *n*-gram independence assumption:

$$P(w_i | w_1 \dots w_{i-1}) = P(w_i | w_{i-n+1} \dots w_{i-1}) \qquad (4)$$

A straightforward maximum likelihood estimate of *n*-gram probabilities from a corpus is given by the observed frequency:

$$P(w_i|w_{i-n+1}\dots w_{i-1}) = \frac{\#(w_{i-n+1}\dots w_i)}{\#(w_{i-n+1}\dots w_{i-1})} \quad (5)$$

where #(.) is the number of occurrences of a specified gram in the training corpus. In [24], Cavnar and Trenkle summarize Zipf's Law as "*The n^{th} most common word in a human language text occurs with a frequency inversely proportional to n*". That is, $f \propto \frac{1}{r}$, where *f* is the frequency of the word and *r* is the rank of the word in the list ordered by the frequency [25]. Therefore, some mechanism for assigning non-zero probability to novel *n*-grams is a central and unavoidable issue. Some standard approaches to smoothing probability estimates to cope with sparse data problems (and to cope with potentially missing *n*-grams) are add-one, absolute discounting and back-off estimator [26,27].

VI. EXPERIMENT

The dataset used in our experiments is derived from Hamshahri Corpus. Hamshahri dataset [22] produced according to CLEF standard in Tehran university research group. Hamshahri is one of the most popular daily newspapers in Iran that has been publishing for more than 20 years. Hamshahri corpus is a Persian test collection that consists of news texts from this newspaper since 1996 to 2007. This corpus contains more than 300,000 news articles about variety of subjects.Hamshahri 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. Therefore, in order to avoid ambiguities, classes are merged to 9 final categories prior to training. Mentioned text documents have been stored in XML format and UTF-8 standard.

The Persian dataset we used in Persian classification is a part of Hamshahri dataset. Our dataset consists of 9000 Persian documents of different lengths that belong to 9 categories. The categories are: Literature and Art, Social, Science and Culture, Miscellaneous, Politics, Sport, Natural Environment, Economy, Tourism. For each category we randomly select about 1000 sample documents. By using an XML parser, we saved each document of news in a separate text file.

For evaluating the IR system, we use the collection contains a topic set of 50 queries and their judgment that was created by 25 different users during summer 2009[16]. Topics of this version of Hamshahri were created using the University of Tehran Information Retrieval Evaluation system (UTIRE). Each topic consists of three parts: Title, Description and Narrative. Title contains up to two or three words which give the main idea of the topic. Description contains a full sentence or question describing the topic in short. Narrative contains a broader description of the topic including examples and perhaps mentions aspects that should not be counted as relevant.

The underlying IR system is based on the open-source search software, Lucene[28,29]. The default document ranking algorithm from Lucene was used.

The evaluation is done by using the standard TrecEval tool which is provided by NIST [30] and used in TREC evaluations.

VII. EXPERIMENTAL RESULTS

We performed two sets of experiments. In the first set we assess the results accuracy of three mentioned classifiers on Hamshahri dataset. In the second set, we compare the quality results of the standard IR System to proposed classification-based IR system and the typical classification-based IR system.

A. Classifiers Assessment

A.1. KNN

In the first step, we apply KNN algorithm and examine this issue to classification process from different aspects such as the number of selected features, the number of neighbors and distance metric.

Table1 shows the importance of selected features number. The more number of features makes the classifier efficiency increase. According to this table, when the number of features is more than 4000, error rises.

TABLE 1. CORRECT RATE AND ERROR RATE FOR DIFFERENT NUMBER OF FEATURES

K=3, Distance Metric=cosine , Rule=nearest		
Number of features	**Correct Rate (Precision)**	**Error Rate**
10	0.6975	0.3025
100	0.8950	0.1050
500	0.9450	0.0550
1000	0.9475	0.0525
2000	0.9613	0.0388
3000	0.9700	0.0300
4000	**0.9738**	**0.0262**
5000	0.9712	0.0288

In Table2, we show the effect of the K parameter. K indicates the number of nearest neighbors that we consider for predicting the class of a new sample. The mentioned Table shows whatever K value is less, classifier result get improved.

TABLE 2. CORRECT RATE & ERROR RATE FOR DIFFERENT VALUE OF K

N=1000, Distance Metric=cosine , rule=nearest		
k	**Correct Rate**	**Error Rate**
1	**0.9888**	**0.0112**
3	0.9475	0.0525
5	0.9313	0.0688
7	0.9213	0.0788
9	0.9163	0.0838
11	0.9113	0.0887

Another factor that we examine is distance metric. This metric is a measure of similarity or dissimilarity that can be used to organized samples according to their degree of relation to another. In our experiments, we find the cosine function is the best for determining the similarity of a new test sample with samples of train data. According to Table3, correlation is useful metric too.

TABLE 3. CORRECT RATE AND ERROR RATE FOR DIFFERENT DISTANCE METRIC

N=1000, k=3, rule=nearest		
Distance Metric	**Correct Rate**	**Error Rate**
Euclidean	0.6700	0.3300
City-block	0.4525	0.5475
Cosine	**0.9475**	**0.0525**
Correlation	0.9437	0.0563

A.2. SVM

In the next step, we examine the effect of different factors in SVM algorithm on classification process. For example, we apply this algorithm on the various feature vector length. According to Table4, when the length of feature vector increased, the accuracy of classifier goes up. It is value to mention by using the more features, the test process takes more time.

TABLE 4. EVALUATION METRICS FOR VARIOUS FEATURE VECTOR LENGTH

Feature Vector Length	Precision	Recall	F-Measure
10	0.7846	0.7015	0.7407
100	0.8784	0.8938	0.8860
200	0.8983	0.9496	0.9232
300	0.8987	0.9717	0.9338
400	0.8989	0.9761	0.9360
1000	**0.8996**	**0.99**	**0.9426**

Also, we measured the effect of different kernel function on classifier performance. The selection of an appropriate kernel function is important, since the kernel function defines the feature space in which the training set examples will be classified. As long as the kernel function is legitimate, an SVM will operate correctly even if the designer does not know exactly what features of the training data are being used in the kernel-induced feature space. Table5 shows the evaluation results of applying the various kernel functions.

TABLE 5. EVALUATION METRICS FOR DIFFERENT KERNEL FUNCTION

Kernel Function	Precision	Recall	F-Measure
Linear	0.7865	0.8750	0.8284
Polynomial	**0.8918**	**0.9895**	**0.9381**
RBF	0.8715	1	0.9313
Quadratic	0.8127	0.8652	0.8381
MLP	0.7865	0.8750	0.8284

As we saw in Table5, when we apply polynomial kernel function, we achieve to highest precision and F-measure, but recall of RBF kernel function is high among the other functions.

A.3. N-Gram Language Modeling

A.3.1. Influence of Linguistic Preprocessing

In order to investigating the effect of lingual preprocessing in classification performance, we get the accuracy without and with considering the preprocessing. The intention of linguistic preprocessing in this paper is normalizing, eliminating the stop words, tokenizing and stemming.

Table 6 and Table 7 show these results respectively:

TABLE6. CLASSIFICATION ACCURACY WITHOUT CONSIDERING LINGUISTIC PRE-PROCESSING

Number of Dataset Sample	1-gram	2-gram	3-gram	4-gram
1000	0.47	0.81	0.84	0.846
3000	0.49	0.84	0.85	0.85
5000	0.51	0.87	0.87	0.88
9000	**0.61**	**0.90**	**0.92**	**0.925**

TABLE7. CLASSIFICATION ACCURACY WITH CONSIDERING LINGUISTIC PREPROCESSING

Number of Dataset Sample	1-gram	2-gram	3-gram	4-gram
1000	0.42	0.87	0.91	0.92
3000	0.43	0.87	0.92	0.93
5000	0.54	0.89	0.96	0.96
9000	**0.67**	**0.96**	**0.98**	**0.984**

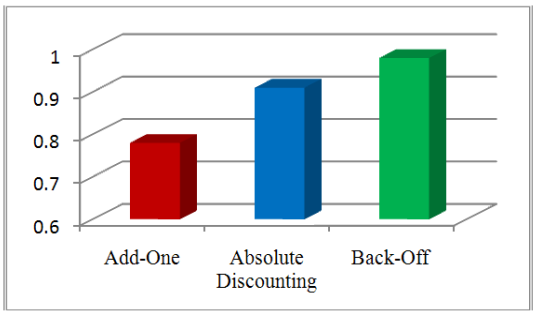
A.3.2. Influence of the n-gram Order

The order n is a key factor in *n*-gram language modeling. An order n that is too small will not capture sufficient information to accurately model word dependencies. On the other hand, a context n that is too large will create sparse data problems in training. In our experiments, we did not observe significant improvement when using higher order *n*-gram models (n>3). In fact, we observed an immediate decrease in performance for the word level model, due to the early onset of sparse data problems. For solving this problem, we use of smoothing methods, hence, the accuracy of results which presented in Table 6 and Table 7 calculated by applying the back-off smoothing. Also if more training data were available, the higher order models may begin to show an advantage. For example, in the larger dataset (average 1000 documents per class for training) we observe an obvious increase in classification performance with higher order models (Table 7). However, it is valuable to mention when n becomes too large, over-fitting will begin to occur.

A.3.3. Influence of Smoothing Techniques

Smoothing plays a key role in language modeling.

FIGURE 5. 3-GRAM CLASSIFIER ACCURACY WITH DIFFERENT SMOOTHING METHODS



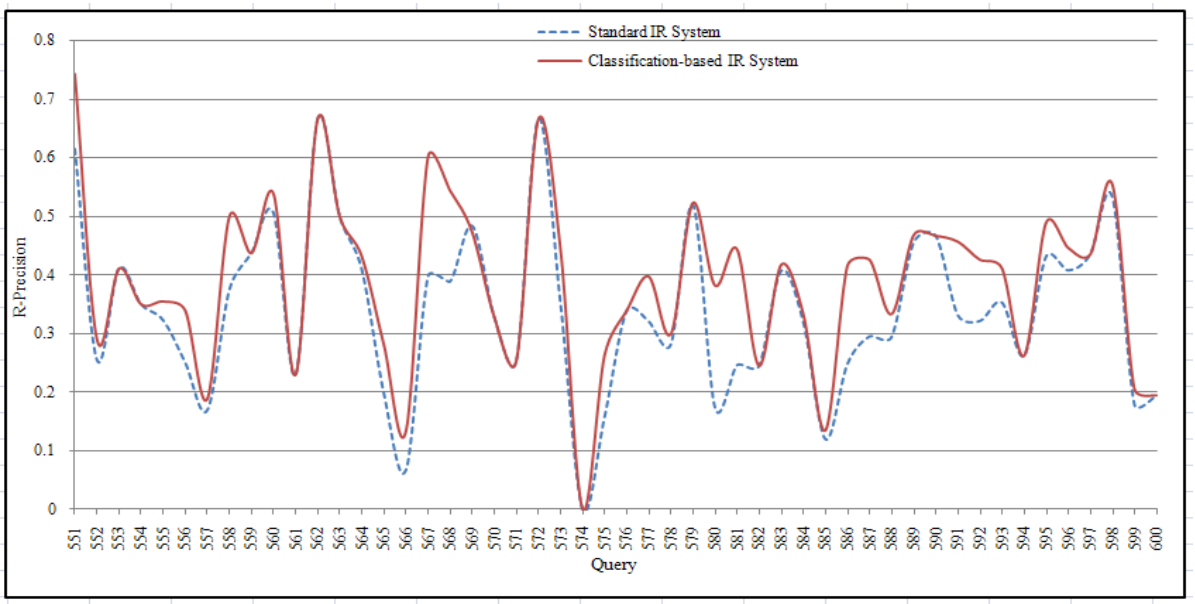


Figure 6. R-Precision for all 50 queries in traditional system and classification-based retrieval system

In the case we have examined, add-one smoothing is obviously the worst smoothing technique, since it systematically overfits much earlier than the more sophisticated smoothing techniques. Back-off smoothing makes the better accuracy in our dataset. Fig 5 shows these results.

A.4. Comparison of the Used Classification Methods

Table 8 shows the summary of the three discussed methods at a glance. It should be noted that for each method, the best is considered appropriate.

As we saw in Table 8, although all three algorithms show acceptable results for Persian text classification, the performance of KNN and 3-gram language modeling classification are better in comparison to SVM. Since in KNN algorithm, it needs to compare a test document with all samples in the training set, this algorithm is less efficient than the 3-gram classifiers.

TABLE8. COMPARISON OF PRECISION OF COMMON TEXT CLASSIFICATION APPROACH FOR PERSIAN NEWS

Method	Features	Precision
SVM	Num of Features=1000, Kernel Function=Polynomial	**89.96**
KNN	K=1, Distance Metric=cosine, rule=nearest, Num of Features=4000	**98.88**
N-gram Language Modeling	N=3, With Language Preprocessing, back-off smoothing	**98**

B.Classification-based IR System Assessment

In this section, we compared the performance of traditional IR system with classification-based IR search implemented in [2] and the proposed system. Using target testing, we quantified the benefits of grouping results in comparison to a standard IR system in terms of the rank of the target documents in the result set. The evaluation is done by using the standard TrecEval tool. For each system, we consider the top 100 retrieved documents for each query.

Figure 6 shows the R-Precision for all 50 queries in three mentioned systems. The R-Precision is defined as the precision for the first R results, where R is the total number of relevant documents for the query.

Figure 7 depicts the mean value of R-Precision for 50 queries in each retrieval system.

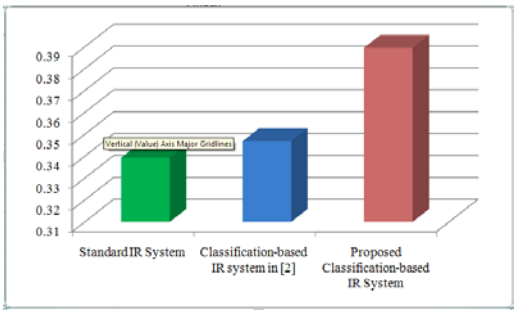


Figure 7. R-Precision average for three systems

Precision and recall are single-value metrics based on the whole list of documents returned by the system. For systems that return a ranked sequence of documents, it is desirable to also consider the order in which the returned documents are presented. By computing a precision and recall at every position in the ranked sequence of documents, one can plot a precision-recall curve, plotting precision p(r) as a function of recall r.

For each query, the interpolated precision is measured at the 11 recall levels of 0.0, 0.1, 0.2, ..., 1.0. For each recall level, we then calculate the arithmetic mean of the interpolated precision at that recall level for each information need in the test collection. Figure 8 depicts interpolated precision and recall of both systems.

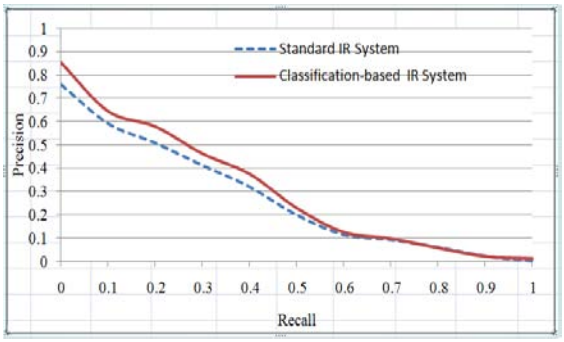


Figure 8. Recall-Precision curve

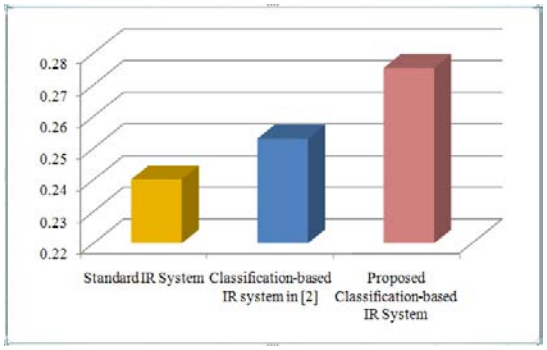


Figure 9. Comparison of average precision for three system

In figure 9, we show the average precision for all 50 queries in traditional system and classification-based retrieval system. Average precision computes the average value of p(r) over the interval from r=0 to r=1:

$$AveP = \frac{\sum_{r=1}^{N}(P(r) * rel(r))}{number\ of\ relevant\ documents} \quad (6)$$

p@n is the precision at top n returned results, which is defined as:

$$p@n = \frac{\#of\ relevant\ docs\ in\ top\ n\ results}{n} \quad (7)$$

where $rel(d_i)$ is binary, which is set to *1* when d_i is relevant to the query. Figure 10 compare this measure for both systems:

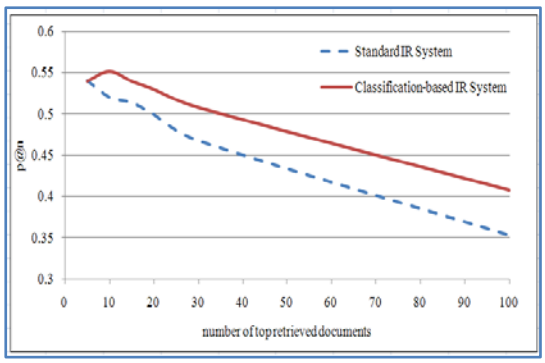


Figure 10. p@n for both system

CONCLUSION

In this paper, we have introduced a hybrid model of information retrieval system that by applying the classification on documents is able to increase the quality of search results. The proposed classification-based IR system begins with a classification-based strategy and reverts to a ranked-list strategy if the user doesn't find the target document in the selected class. We applied query classification in addition to document classification to gain the additional insight of the users' intent.

This may imply that a classification-based system may be more beneficial for informational queries, where the user will probably inspect several search results, rather than for navigational queries, which are similar to known item queries that target a single web page. Such a system could also be useful for novice users who are more likely to generate poor queries.

The experiment results on Hamshahri corpus show the performance of the proposed hybrid system is better than the baseline IR systems and the classification-based IR system implemented in [2].

In the future we intend to consider user interaction. By using the user feedback, system can better understand his query and returned the more relevant results.

REFERENCES

[1] X.G. Qi, "Web page classification and hierarchy adaption", A dissertation presented to the graduate and research committee of Lehigh university, 2012

[2] Zh. Zhu, I.J. Cox, M. Levene, "Ranked-Listed or Categorized Results in IR: 2 Is Better Than 1", In Proceeding of NLDB, P. 111-123, 2008.

[3] Zh. Zhu, "Improving Search Engines via Clasification", A dissertation submitted to Birkbeck College, university of London, 2011

[4] B. Cao, J.T. Sun, E. Wei Xiang, D. Hao Hu, Q. Yang, Zh. Chen, "PQC: Personalized Query Classification", Proceeding of the 18th ACM Conference on Information and Knowledge Management, China, November 2-6, 2009

[5] M.A. Hearst, J.O. Pedersen, "Reexamining the cluster hypothesis: scatter/gather on retrieval results", In Proceedings of SIGIR-96, 19th ACM International Conference on

Research and Development in Information Retrieval, pp. 76-84, 1996

[6] K. Shin, S. Han, "Fast Clustering Algorithm for Information Organization", In CICLing 2588 of LNCS, P. 619-622, Springer, 2010.

[7] D. Xing, G.R. Xue, Q. Yang, Y. Yu, "Deep Classifier: Automatically Categorizing Search Results into Large-Scale Hierarchies", Proceedings of the Internatinal Conference on Web Search and Web Data Mining, USA, February 11-12, 2008.

[8] S. Osinski, D. Weiss, "Carrot 2: Design of a flexible and efficient web information retrieval framework". In Proceedings of the third International Atlantic Web Intelligence Conference, Berlin. LNCS, pp. 439–444, 2005

[9] H.J. Zeng, Q.C. He, Z. Chen, W.Y. Ma, J.W. Ma, "Learning to cluster web search results", In Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 210–217, 2004

[10] H. Chen, S. Dumais, "Bringing order to the web: automatically categorizing search results", In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 145–152, 2000.

[11] A. Singh, K. Nakata, "Hierarchial classification of web search results using personalized ontologies", In Proceedings of the 3rd International Conference on Universal Access in Human-Computer Interaction, HCI International , 2005.

[12] D. Xing, G.R. Xue, Q. Yang, Y. Yu, "Deep classifier: automatically categorizing search results into larg-scal hierarchies", In proceeding of the international conference on web search and data mining, Pags 139-148, ACM, 2008.

[13] M. Tsuruta, Y. Umemura, H. Sakai, Sh. Masuyama, "A web search result classification system based on the degree of suitability for specialists", 4th NTCIR workshop, 2004.

[14] Ch. Chekuri, M.H. Goldwasser, "Web search using automatic classification", In Proceedings of the Sixth International World Wide Web Conference, 1997.

[15] B. Kules, J. Kustanowitz, B. Shneiderman, "Categorizing Web Search Results into Meaningful and Stable Categories Using Fast-Feature Techniques", In Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries, USA, June 11-15, 2006.

[16] http://ece.ut.ac.ir/dbrg/hamshahri/

[17] M. Shamsfard, S. Kiani, Y. Shahedi, "STeP-1: Standard Text Preparation for Persian Language", CAASL3 Third Workshop on Computational Approaches to Arabic Script-based Languages, Canada, 2009.

[18] M. Farhoodi, A. Yari, "Applying machine learning algorithms for persian text classification", 2nd International Conference on Data Mining and Intelligent Information Technology Applications, 2010.

[19] Y. Liao, V. Rao Vemuri, "Using Text Categorization Techniques for Intrusion Detection", Proceedings of the 11th USENIX Security Symposium, 2002

[20] S.M. Farshchi, "Document Classification, a Novel Neural-based Classifier", internationa Journal of Engineering and technology, Vol.1, No.2, 2011.

[21] M. Maleki, A. Abdollahzadeh Barfroush, "TFCRF: A Novel Feature Weighting Method Based on Class Information in Text Categorization", 19th international conference on Computer, Information and Systems Science and Engineering ICKM, 2007.

[22] M. Farhoodi, A. Yari, A. Sayah, "N-Gram Based Text Classification for Persian Newspaper Corpus", 7th International Conference on Digital Content, Multimedia Technology and its Applications, 2011

[23] F. Peng, X. Huang, "Machine Learning for Asian Language Text Classification", 2006

[24] W.B. Cavnar and J.M. Trenkle, "N-Gram-Based Text Categorization", In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, 1994.

[25] M. Mansur, N. UzZaman, M. Khan, "Analysis of N-Gram Based Text Categorization for Bangla in a Newspaper Corpus", 2006.

[26] S. Ramasundaram, S.P. Victor, "Text Categorization by Backpropagation Network", International Journal of Computer Applications, 2010

[27] Stanley F. Chen, Joshua Goodman, "An Emperical Study of Smoothing Techniques for Language Modeling", 1999

[28] http://lucene.apache.org/

[29] http://en.wikipedia.org/wiki/Lucene

[30] National Institution of Standards and Technology: http://trec.nist.gov/trec_eval/

Mojgan Farhoodi got her B.Sc. degree in Computer Engineering from Tarbiat Moallem University of Tehran. She also received her M.Sc. degree in IT engineering from Amirkabir University of Technology. She is working now at IT Research Faculty of CyberSpace Research Institute (CSRI). Her research interests include Web Information Retrieval, Natural Language Processing (NLP) and Data Mining. Currently she is working on a Quranic question answering system.

Saeed Shiry Ghidary is an assistant professor in Amirkabir University of Technology. He received his B.Sc. degree in Electronic Engineering from Amirkabir University of Technology in 1990, his M.Sc. degree in Computer Architecture from same university in 1994 and his PhD. in Artificial Intelligent Systems from Kobe University in 2002. His research interests include Machine Learning, AI, Robotics, Mechatronics, Machine Vision, Cognitive Science and Brain Modeling.

AliReza Yari received his B.Sc. degree in Control System Engineering in 1993 from the University of Tehran, Iran, and his M.Sc. and Ph.D. degrees in systems engineering in 2000 from Kitami Institute of Technology, Japan. He is currently doing research in Information Technology Research Faculty of CyberSpace Research Institute (former ITRC). His research interests include web information retrieval and language specific search engine, e.g. Persian search engine. He is also working on application of natural language processing (NLP) for improvement of search engines.