An Automatic Query Generation Approach for Arabic Corpus

Mohammed J. Bawaneh
Information Technology Department , Al-Huson University College
Al-Balqa Applied University

ABSTRACT

One of the most common problems that encounter retrieval systems is the query environment which is utilized to evaluate the performance or throughput of them. Usually data sets of queries for a specific corpus are generated using the human experiences. Manual queries are more accurate than automatic one's, but they require a huge effort in the huge corpus. This paper proposes an automatic query generation (AQG) system for Arabic language. The system generates a set of queries of different length that were applied on a query expansion system. The results show the feasibility of these queries compared with manual one in term of average recall and average precision.

General Terms
Information retrieval, Clustering

Keywords
AQG, Entropy Method, Query generation

1. INTRODUCTION

Information systems store huge amount of documents that require some times a huge amount of time to retrieve them so, many algorithms are used to today manipulate these documents such as vector space model, Boolean model, probability model and many others. The main idea of these models is very clear, while the impact of the query on them is less obvious. Traditionally the performance of these models is measured by selecting a corpus which has a predefined set of queries, set of relevant documents to these queries and then computing output of precision, recall, fallout, and other criteria's in aim to construct the comparison between them[1]. Usually in query based algorithms user defines a set of queries, then evaluates the system based on them. Most of the times those queries are unbiased due to their confirmation on user. User may select queries that give the most accurate results, so the decision will decide this algorithm is good, but this algorithm actually correct, to distinguish that a control query generation approach may be utilized. This work presents an AQG approach for Arabic documents, which is used to generate a set of queries called query environment. Queries that are generated may have no meaning for the user but for systems these queries are the suitable one that represents the clusters inside the corpus. Feasibility of query environment can be computed by applying the results to a stem base query expansion system for Arabic corpus. Building AQG is not a simple task due to many problems that may appear. One dubious that may encounter these systems is the clustering of corpus into sets of related documents. Many techniques are used such as KMeans, KMode and stem base clustering, but the first two techniques require an intervention from user to determine the number of clusters [2], so this system ignored them and used the last one. Selection of model that measures relationship between terms and cluster is another problem in these systems. In this work the relationship between term and cluster will be represented by using an entropy method that takes a value between one and zero. Value of one means strong relation while zero means weak relation or no relation. Query building is the process of selecting specific term(s) form a specific corpus that represents documents in the corpus. There are two common ways to build testing query environment. The first one is built manually, in which user defined a set of queries and related documents, these queries are built based on the experiences of user, while the second one is automatic defined query environment, in which the system defines a set queries and related documents utilizing specific criteria.

2. RELATED WORK

Many researches and literature concentrate on query construction to develop effective method and technique to construct query automatically. A lot of methods and techniques have been built to construct queries manually, but only few methods exist for automatically query construction. Today most of researches focus on how to construct query automatically, because if this developed effectively it will improve information retrieval even on the web. However, there are still much rooms for improving existing methods and techniques to construct more relevant queries automatically especially for Arabic language and terms. The Automatically generated queries have many merits, however the main advantage is that for any given evaluation data set several queries can be constructed at minimal cost [3].

Kumaran and Carvalho proposed a support vector machine for ranking queries, the machine predicts a new queries or sub-queries from a predefined set of queries [4].

Bendersky and Croft focused on key concepts of a given documents. They developed an automatic extraction system by using of different types of features on key concepts [5].

Lee and others worked with web text by proposing a learning-based approach. The application ranks the query terms in aim to be used in constructed query. The number of terms in query was done manually [6].

Lacatusu and others proposed a system worked on the query itself. The system breaks down query into more simple queries in order to produce a summarization for query [7].

Nenkova and others worked with multi-document summarization .they proposed a system that finds the main keys of a given documents and sets them in a pool of keys in order to be utilized in or added to the summary of document[8].

3. MATERIALS AND METHODS

Constructing ACQ system requires a several steps to be carried out. These steps start by clustering corpus, selecting manipulation model for corpus and clusters, selecting good storage strategy, selecting a good criteria for sorting terms, building queries from these terms and finally queries are normalized in order to select more related corpus queries.

3.1 Corpus Selection
The used corpus was 242 Arabic documents that talk about different subjects. For simplicity in manipulation and storing the documents were renamed starting from D1 to D242.

3.2 Storing Files
File names and sizes are stored in the storage. File size is utilized in documents clustering process. System selects the file with the largest size; in order to compare it with other files. However, in the initial state all files are considered to be members of cluster zero which represents corpus itself.

3.3 Corpus Clustering
Corpus is distributed into a set of clusters in which each cluster contains a number of related documents. Related documents are obtained using stem base classifier. An Arabic stemming algorithm was used to find the stemmer for each selected word, the algorithm is called Light Stemming [9].

Clustering process starts by selecting the file of maximum size, then comparing the remaining files with it. Each file has a similarity over or equal a specific thresh hold is selected as a member of this cluster. Neither the process continues until nor more files still without cluster. The documents are clustered into a number of clusters which starts from index one. After completing the clustering process some clusters may contain small number of documents, so it needs to be normalized or discarded. Clustering process can be summarized by a set of steps as follow:

```
1. Set cluster number=0
2. Select the file with maximum size from non-selected files
   in corpus and call it center
3. Set cluster number= cluster number + 1
4. Tag the center file as selected in aim to avoid it in next
   selection
5. Find and store stems of center document.
6. Select a non-selected file from corpus and call it member
     Find and store stems of member document.
     Find cosine similarity between center and member
   If similarity >= thresh and similarity > stored similarity then
     Update the stored center and similarity
   End If
7. repeat steps 2—6 until no more files without clusters
8. Normalize clusters
```

3.4 Normalization Process
Clusters indexing start from one and it may ends at one, two, three, or N where N is the number of documents in the cluster. Some clusters may contain small number related documents these clusters must be eliminated and returned back to cluster zero which represents corpus itself. Normalization process can be summarized as follow:

```
1. For each cluster in the corpus do
     If number of files <= normalization factor hold then
         Update document center to zero
     End If
2. Repeat step1until no more clusters
```

3.5 Building Inverted File
Inverted file includes three containers for storing data called Corpus_Container, Cluster_ Container, and Document_ Container. Corpus_ Container was used to store terms and there frequencies in the corpus. Cluster_Container was utilized to store terms, clusters, and frequencies of term in cluster itself, while Document_Container stored term, and its frequency inside the document.

3.6 Building Language Model
Model for source document set is built by concatenating all documents and treating them as single piece of sample text. Corpus requires a language model to represent it mathematically, to accomplish that the next formula will be utilized [10].

$$\mathbf{PML(\,t, D_k) = (freq\ t, D_k) / (\textstyle\sum freq\ D_k),}$$

Where

- t is a term in the corpus vocabulary
- D_k is the document k in the corpus
- Freq D_k, is the total frequency of terms in D_k

PML formula above may give Zero value when a given term not appears in a specific document, so the problem can be solved using the next formula [10].

$$\mathbf{PJM(\,t, D_k) = (1\text{-}ג\ )\ PML(\,t, D_k)\ +\ ג\ PML(\,t, Corpus),}$$

Where

- ג is constant value between 0 and 1(balance factor)

3.7 Sort Terms
After building language model, the system needs to compute the score for each term in the vocabulary. The ulterior formula will be employed to compute the scores for terms [10]. Scores are sorted to be utilized in building queries in next phase.

$$\mathbf{Score(t) =\ PJM(t, D_k)\ log\ \ [PJM(t, D_k)/\ PML(t\ ,Corpus)]}$$

3.8 Query generation
Queries are built from the scored sorted term, in which term of the most contribution is selected. This process passes into two phase's, in initial phase the most four discriminating terms are selected to build initial queries that consist of one, two, three, and four terms. In the next phase queries of length one, two, three, four, and five terms are built from previous queries and remaining terms. The process stills working until no terms have a high score and not selected. The following steps will explain the two mentioned phases.

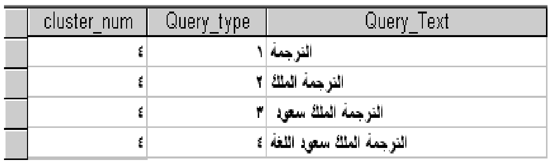
Stage I:

1. Identify the term "T1" that contributes the most related one to the cluster(highest score)
   - 1.1 For single term query Q1= T1
   - 1.2 For two terms query Q2= T1 + T2 where T2 is the next highest
   - 1.3 For Three terms query Q3= T1 + T2 + T3 where T3 is the next highest
   - 1.4 For four terms query Q3= T1 + T2 + T3 + T4 where T4 is the next highest

Stage II:

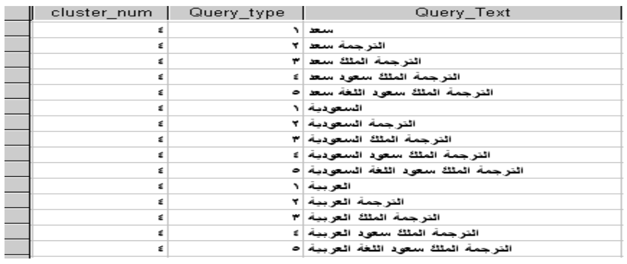
1. Identify the term "Ti" that contributes the most related one to cluster and not already in a query
2. Generate next query as following
   - 2.1 For single term query Q_i= Ti
   - 2.2 For two terms query Q_i= T1 + T_i
   - 2.3 For Three terms query Q_i= T1 + T2 + T_i
   - 2.4 For four terms query Q_i= T1 + T2 + T3 + T_i
   - 2.5 For five terms query Q_i= T1 + T2 + T3 + T4 + T_i
3. Repeat steps 1-- 2 until no term has contribution larger than a specific thresh hold

Figure 1 and Figure 2 show a sample of queries, query type and index of related cluster that result at the end of phase one and two.



cluster_num	Query_type	Query_Text
٤	١	الترجمة
٤	٢	الترجمة الملك
٤	٣	الترجمة الملك سعود
٤	٤	الترجمة الملك سعود اللغة

Figure 1 : Query generation phase one



cluster_num	Query_type	Query_Text
٤	١	سعد
٤	٢	الترجمة سعد
٤	٣	الترجمة الملك سعد
٤	٤	الترجمة الملك سعود سعد
٤	٥	الترجمة الملك سعود اللغة سعد
٤	١	السعودية
٤	٢	الترجمة السعودية
٤	٣	الترجمة الملك السعودية
٤	٤	الترجمة الملك سعود السعودية
٤	٥	الترجمة الملك سعود اللغة السعودية
٤	١	العربية
٤	٢	الترجمة العربية
٤	٣	الترجمة الملك العربية
٤	٤	الترجمة الملك سعود العربية
٤	٥	الترجمة الملك سعود اللغة العربية

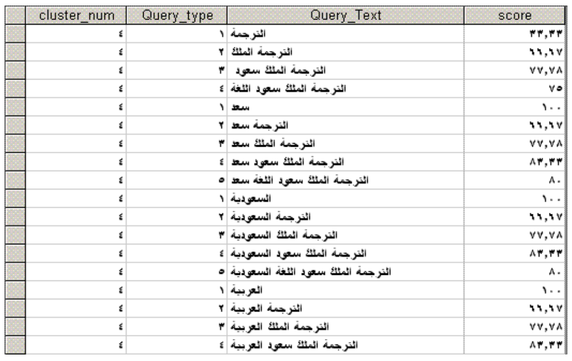
Figure 2 : Query generation phase two

3.9 Queries Normalization

It means that selecting queries are more related to the cluster than other queries and set them as final result. These queries will be used to test the system itself and other systems. Next steps explain how the system normalizes the queries.

1. For each cluster in the corpus do
2. For each query in the cluster do
3. Compute the number of related documents (NRD) in cluster to query
4. Repeat step 2—3 until no more queries in the cluster
5. Compute relativity = NRD / (# terms in query * # of documents in cluster)
6. If relativity >= threshold then accept the queries as an electing query
7. Repeat steps 1—6 until no more cluster in the corpus

Figure 3 gives a brief sample of a normalize queries with thresh hold larger than or equal 85 %



cluster_num	Query_type	Query_Text	score
٤	١	الترجمة	٣٣,٣٣
٤	٢	الترجمة الملك	٦٦,٦٧
٤	٣	الترجمة الملك سعود	٧٧,٧٨
٤	٤	الترجمة الملك سعود اللغة	٧٥
٤	١	سعد	١٠٠
٤	٢	الترجمة سعد	٦٦,٦٧
٤	٣	الترجمة الملك سعد	٧٧,٧٨
٤	٤	الترجمة الملك سعود سعد	٨٣,٣٣
٤	٥	الترجمة الملك سعود اللغة سعد	٨٠
٤	١	السعودية	١٠٠
٤	٢	الترجمة السعودية	٦٦,٦٧
٤	٣	الترجمة الملك السعودية	٧٧,٧٨
٤	٤	الترجمة الملك سعود السعودية	٨٣,٣٣
٤	٥	الترجمة الملك سعود اللغة السعودية	٨٠
٤	١	العربية	١٠٠
٤	٢	الترجمة العربية	٦٦,٦٧
٤	٣	الترجمة الملك العربية	٧٧,٧٨
٤	٤	الترجمة الملك سعود العربية	٨٣,٣٣

Figure 3 : Normalize queries

4. EXPERIMENTS AND RESULTS

4.1 Construction Time

The proposed system was tested and evaluated using 3.0 GHz CPU with 2.0 GB RAM, the execution time took:

- 32 minutes for clustering corpus into clusters.
- 23 minutes to build inverted file for [Corpus-Container, Clusters-Container, and Documents-Container].
- 20 minutes to build models for Corpus, Clusters, Documents and queries
- 16 minutes to normalize queries environment

4.2 The Corpus

The data set or corpus contains 242 documents with average of 500 words per document. To cluster documents a thresh hold of 80% was selected due to different size of tested documents.

4.3 Evaluation

The system was evaluated using 20 normalize queries of different size. They were selected randomly from the results of proposed system. Tested queries are shown in Table 1.

Table 1 : Normalize Tested queries

Index	Cluster	Type	Query
1	1	1	العربية
2	2	1	جامعة
3	3	3	والعمرة مركز جده
4	3	3	والعمرة مركز الحاسب
5	2	2	الحاسوب مؤتمر
6	3	4	والعمرة مركز عمليات العربية
7	3	4	والعمرة مركز عمليات مؤتمر
8	3	3	والعمرة مركز انشاء
9	1	3	السيارات السعودية رقم
10	2	2	الحاسوب المعلوماتية
11	1	3	السيارات السعودية الحاسب
12	2	2	الحاسوب الملك
13	3	5	والعمرة مركز عمليات الحج الانجليزية
14	2	3	الحاسوب التجربة جامعة
15	3	3	والعمرة مركز وتحسين
16	1	2	السيارات العربية
17	1	3	السيارات السعودية الملك
18	2	2	الحاسوب الملك
19	3	3	والعمرة مركز الآلي
20	2	2	الحاسوب لغه

4.4 Results

Performance of the system was computed by using average of recall and precision. Table 2 shows the related documents for each cluster that were determined by the system itself through the clustering process. These documents will be used later on in computing recall and precision for the system. Table 3 displays index of query, index of cluster and indices of retrieved documents. Index of query and index of cluster in table 3 are related to each other. The relationship between them is shown in table 1. Data of table 4 consists of index of query, number of retrieved related documents to query, number of retrieved unrelated documents, precision and recall for each query. This data was built according to data in table 2 and table 3.

Table 2 : Cluster numbers and related files

Cluster	Related Documents
1	D2, D4, D7, D9, D11, D12, D14,D18, D19, D21, D25,D28,D33
2	D1, D5, D6, D10, D23, D24 , D31, D32, D37, D38, D39, D40
3	D3, D8, D13, D17,D20, D22,D26,D27,D29, D30, D34,D35, D36

Table 3 : Retrieved documents for queries

Index	Cluster	Retrieved document
1	1	D1, D2, D4, D11, D25, D28, D29, D30, D31, D33, D38, D39
2	2	D1, D5, D10, D24, D31,D33, D38, D39, D40
3	3	D3, D7, D8, D16, D17,D22, D29, D30, D35
6	3	D3, D7, D8, D16, D17,D22, D29, D30, D35,D37, D38
11	1	D1, D3, D4, D11, D12, D18, D19, D21, D25, D28, D38, D39

Table 4 shows the average recall and precision for the query expansion system using automatic generation. The proposed system gives very good results compared to the manual one.
The variance between recall and precision is not enormous, these differences in recall and precision due to the small size of corpus that were used in evaluating process

Table 4 : Recall and precision for queries

Index	Related doc	Unrelated doc	Precision	Recall
1	6	6	0.5	0.46
2	8	1	0.89	0.67
3	7	2	0.78	0.53
6	7	4	0.64	0.53
11	8	4	0.67	0.62
Average value			0.696	0.562

5. CONCLUSION

Many steps are required to build ACQ system so it needs high speed hardware to accomplish that. Results may not give a specific meaning for users but for system the inverse is true. Manipulating large corpus may give results more nearby to the manual one. Queries environment are more unbiased than manual due to its dependability on the system rather than user opinion. For future work, it would be helpful to use a huge corpus with high performance hardware and additional models for evaluating the system.

6. REFERENCES

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto,“Modern Information Retrieval”, Addison-Wesley 1999.

[2] Richard Dubes and Anil Jain, “Algorithms for clustering data”. Prentice hall. 1988.

[3] Leif Azzopardi and Maarten de Rijke, "Automatic Construction of Known-Item Finding Test Beds", SIGAR'06, August 6-11, 2006, Seattle, Washington, USA. Copyright 2006 ACM 1-59593-369-7/06/0008.

[4] G. Kumaran and V. R. Carvalho, “Reducing long queries using query quality predictors”, In Proc. Of SIGIR, SIGIR '09, pages 564-571, 2009.

[5] M. Bendersky and W. B. Croft,” Discovering key concepts in verbose queries”, In Proc. of SIGIR, SIGIR '08, pages 491{498, 2008.

[6] C.-J. Lee, R.-C. Chen, S.-H. Kao, and P.J. Cheng, “ A term dependency-based approach for query terms ranking”, In Proc. of CIKM, CIKM '09, pages 1267-1276, 2009.

[7] F. Lacatusu et al. LCC's Gistexter at DUC'06, ” Multi-strategy multi-document summarization”, DUC, 2006.

[8] A. Nenkova, L. Vanderwende, and K. McKeown, “ A compositional context sensitive multi-document summarizer”, SIGIR, pages 573{580, 2006.

[9] Mohammed Aljlayl Riyadh College of Technology, Riyadh, Saudi Arabia and Ophir Frieder, Illinois Institute of Technology, Chicago “On Arabic Search: Improving the Retrieval Effectiveness via a Light Stemming Approach”.

[10] Chris Jordan, Carolyn Watter & Qigang Gao: "Using Controlled Query Generation to Evaluate Blind Relevance Feedback Algorithms",JCDL'06, June 11-15, 2006, Chapel Hill, North Carolina, USA. Copyright 2006 ACM 1-59593-354-9/06/0006