Stemming and Lemmatization: A Comparison of Retrieval Performances

Vimala Balakrishnan and Ethel Lloyd-Yemoh, *Member, IACSIT*

***Abstract*—The current study proposes to compare document retrieval precision performances based on language modeling techniques, particularly stemming and lemmatization. Stemming is a procedure to reduce all words with the same stem to a common form whereas lemmatization removes inflectional endings and returns the base or dictionary form of a word. Comparisons were also made between these two techniques with a baseline ranking algorithm (i.e. with no language processing). A search engine was developed and the algorithms were tested based on a test collection. Both mean average precisions and histograms indicate stemming and lemmatization to outperform the baseline algorithm. As for the language modeling techniques, lemmatization produced better precision compared to stemming, however the differences are insignificant. Overall the findings suggest that language modeling techniques improves document retrieval, with lemmatization technique producing the best result.**

***Index Terms*—Document retrieval, language models, lemmatization, stemming.**

I. Introduction

The increase in size of data and information collections over the past couple of years made it necessary for tools to be developed in order to access information with much ease. Over the years, information retrieval methods have been developed and enhanced to assist users in looking for the right information. Information retrieval focuses on getting or providing users with easy access to the information they need. It does not only look for the right information but represents it in a manner that is easily understandable to users, stores the information in an orderly manner and organizes it in such a way that it can be easily retrieved at a later time [1]. Basically, information retrieval can be defined as *"a problem-oriented discipline, concerned with the problem of the effective and efficient transfer of desired information between human generator and human user"* [2].

Various mechanisms have been developed over the years to assist users in retrieving information. Common ones include the Boolean model, which uses queries with precise semantics coupled with binary decisions. In this model, a document is retrieved based on a binary decision of either a document being relevant or non-relevant [3]. The vector space model on the other hand compares user queries with documents found in collections and computes the extent to which these two are similar. It then ranks the retrieved documents according to their degrees of similarities [4]. The vector space model is mostly used in information filtering, indexing and relevance rankings.

Today, the use of Internet all over the world resulted in the information size to increase and made it possible for large volumes of information to be retrieved at any given time. Thisalso means that both relevant and non-relevant information will be retrieved [5], thereby slowing down the retrieval process. However, speed and relevancy are very essential in the retrieval of information and information seekers look for ways to improve this aspect of the retrieval process. This eventually resulted in the birth of language models. Although a lot of studies have been done in this area, there is still a high demand for retrieval improvements. There are still a lot of non-relevant documents being retrieved even with stemming or lemmatization techniques being applied to search queries. Studies based on stemming and lemmatization techniques have reported improved document retrievals, however it would be interesting to assess their performances by way of a comparison. The current study hence aims to 1) compare the document retrievals using stemming and lemmatization techniques, and 2) compare the stemming and lemmatization techniques against a baseline ranking algorithm (i.e. with no language processing).

The remainder of the paper is structured as follows: the related works are discussed in the following section. This is then followed by the research design which focuses on the stemming and lemmatization techniques, experiment setup and the evaluation metrics used. The results and discussion follow next.

II. Related Work

In the language model, users create a query to describe the information that they need and the system will choose keywords from the query that are deemed to be relevant. These keywords will be matched againstthe documents in a collection. When similarities are found between the given query and a document in the collection, that document is retrieved and then matched against the rest of the retrieved documents for ranking purposes [1]. There are two procedures that usually help to improve the language models by quickening the search process, and these are stemming and lemmatization.

Stemming is one of the techniques used in information retrieval systems to make sure that variants of words are not left out when text are retrieved [5]. The process is used in removing derivational suffixes as well as inflections (i.e. suffixes that change the form of words and their grammatical functions) so that word variants can be conflated into the same roots or stems.Stemming mechanisms have been used in a lot of language research areas such as Arabic [6],

Manuscript received January 16, 2014; revised March 14, 2014. This study was supported by the University of Malaya (RP002B – 13ICT).

The authors are with the Faculty of Computer Science and Information Systems, University of Malaya, Kuala Lumpur, Malaysia (e-mail: vimala.balakrishnan@um.edu.my, ethel_lloyd@siswa.um.edu.my).

cross-lingual retrieval [7] and multi-language manipulations [8].

There are various stemming algorithms that have been developed to ensure that words are reduced to their root forms, thereby reducing the size of document dictionary. This is because one root or stem can be used to represent many variants of terms used in a particular language. Although this approach helps in retrieving more relevant documents, there is the possibility of either under-stemming (where two words belonging to the same conceptual group are converted to two different stems or roots, e.g. a search for the word "run" not containing documents which have "running" and "ran" in them), or over-stemming (where two words belonging to different conceptual group are converted to the same stems or roots, e.g. when a search for the word "new" includes a search result containing the word "news").

Stemming techniques are many, including the Paice/Husk stemmer [9], Porter's stemmer [10] and Lovin's stemmer [5].

In the Paice/Husk stemmer, a file is created which holds a set of rules, and these rules are read by an array which implements the rules until a final stem is achieved. It accepts and processes a rule if the word specifies an ending which matches the last letters of the word [9]. The Lovin's stemmer was developed to deal with both information retrieval and computational linguistics problems. The Lovins stemmer is a single pass, context-sensitive algorithm which only removes one suffix from a word by utilizing a list of 250 suffixes and removing the longest suffix that it finds attached to the given word. The stemmer ensures that when a word has been stemmed, it is at least three characters long [5]. The Porter's stemmer was used in the current study, and is discussed in the next section.

Lemmatization on the other hand uses vocabulary and morphological analysis of word and tries to remove inflectional endings, thereby returning words to their dictionary form. It checks to make sure that things are done properly by analyzing if query words are used as verbs or nouns. Lemmatization also helps to match synonyms by the use of a thesaurus so that when one searches for "hot" the word "warm" is matched as well. In the same light a search for "car" will produce "cars" as well as "automobile".The lemmatization technique has been used in several languages for information retrieval.For instance, Ozturkmenoglu and Alpkocak [11] compared three different lemmatizers to retrieve information on a Turkish collection.Their results showed that lemmatization indeed improves the retrieval performance utilizing only a minimum number of terms in the system. Additionally, they also found that the performance of information retrieval was better when the maximum length of lemmas is used. In 2012, Gupta *et al*. [12] combined stemming and partial lemmatization and tested their model on the Hindi language. Their model yielded significant improvements compared to the traditional approaches.

Both stemming and lemmatization play very important roles when it comes to increasing relevance and recall capabilities of a retrieval system. When these techniques are used, the number of indexes used is reduced because the system will be using one index to present a number of similar words which have the same root or stem. For instance, when the word "industrialize" is lemmatized, its index can be used for "industrious, industry", etc.

III. Research Design

A. Stemmer and Lemmatizer

Fig. 1 depicts the data flow diagram for a search query that goes through the stemming process. A userenters the search query via the interface. The query is then passed to thesearch engine which will in turn invoke the Porter's stemming algorithm. The stemmingalgorithm is applied to the search query and the resulting stemmed text is returned to thesearch engine. The next step is for the search engine to pass the stemmed text to thedatabase so that it can be matched against the documents that are available in thecollection. This results in the selection of matching data or documents which will bepassed to the search engine and displayed to the user for viewing.

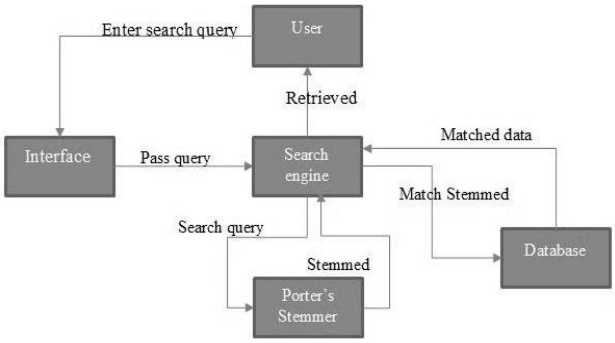


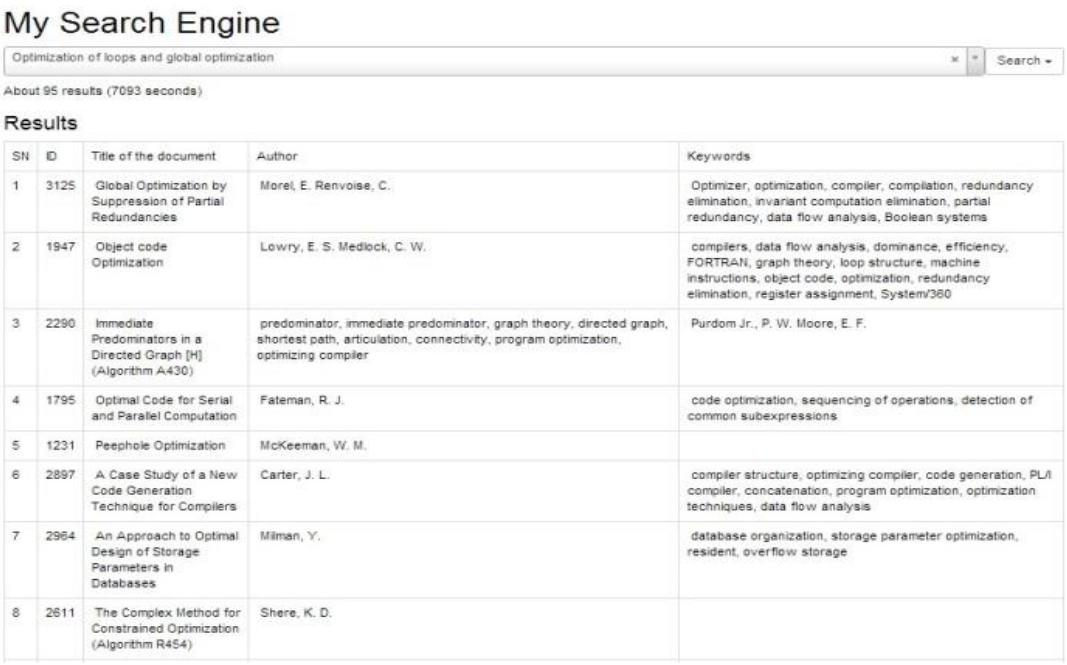
Fig. 1. Data flow diagram for stemming.

The Porter's stemmer is one of the widely used stemmers in information retrieval [10]. When the stemming function of the system is called, it will check the keyword and follow a set of rules. Firstly it will remove all stop words (i.e. a list of words specified by the system to be ignored). These are generally words that frequently occur in search queries, such as "and", "the", etc. The prototype designed in our study contains 430 of these words. The next step will be to remove endings that make the keyword plural (e.g. -s, -es), past tense (-ed), and continuous tenses (-ing).The stemmer then moves on to check and convert double suffices to single suffice. Other suffices such are -ic, -full, -ness,-ant, -ence, just to mention a few are removed as well.

As for creating the lemmatizer, a prebuilt lemmatizer provided by LemmaGen was used in this study. LemmaGenwas particularly chosen as it provides multilingual support, and does not rely on sentence structure of the text which is being processed (i.e. it can be applied on each word separately, and thus can be used to lemmatize search query words). The latter is a very influential characteristic as the proposed search engine might have just one query word or a sentence structure.

B. Nking

The *tf-idf* (term frequency-inverse document frequency) was used as the baseline ranking algorithm [4]. The algorithm checks theretrieved document to see how frequent the words in the search query appears in thedocument. The larger the number of times a query word appears in a document, themore relevant that document is perceived to be in relation to the search query [4].


My Search Engine

Optimization of loops and global optimization

Search

About 95 results (7093 seconds)

Results

SN	ID	Title of the document	Author	Keywords
1	3125	Global Optimization by Suppression of Partial Redundancies	Morel, E. Renvoise, C.	Optimizer, optimization, compiler, compilation, redundancy elimination, invariant computation elimination, partial redundancy, data flow analysis, Boolean systems
2	1947	Object code Optimization	Lowry, E. S. Medlock, C. W.	compilers, data flow analysis, dominance, efficiency, FORTRAN, graph theory, loop structure, machine instructions, object code, optimization, redundancy elimination, register assignment, System/360
3	2290	Immediate Predominators in a Directed Graph [H] (Algorithm A430)	predominator, immediate predominator, graph theory, directed graph, shortest path, articulation, connectivity, program optimization, optimizing compiler	Purdom Jr., P. W. Moore, E. F.
4	1795	Optimal Code for Serial and Parallel Computation	Fateman, R. J.	code optimization, sequencing of operations, detection of common subexpressions
5	1231	Peephole Optimization	McKeeman, W. M.	
6	2897	A Case Study of a New Code Generation Technique for Compilers	Carter, J. L.	compiler structure, optimizing compiler, code generation, PL/I compiler, concatenation, program optimization, optimization techniques, data flow analysis
7	2964	An Approach to Optimal Design of Storage Parameters in Databases	Milman, Y.	database organization, storage parameter optimization, resident, overflow storage
8	2611	The Complex Method for Constrained Optimization (Algorithm R454)	Shere, K. D.	

Fig. 2. Baseline (*tf-idf)* result.

Term frequency $tf_{t,d}$ describes how often a query term t appears in a document d. The term frequency is used as follows:

$$\log(1 + tf_{t,d}) \tag{1}$$

df refers to document frequency and relates to the number of document that contains the search keyword. The inverse document frequency *(idf)* describes the relevance of the search term in relation to all the documents in the collection, as depicted in (2):

$$idf_t = log_{10} \frac{N}{df_t} \tag{2}$$

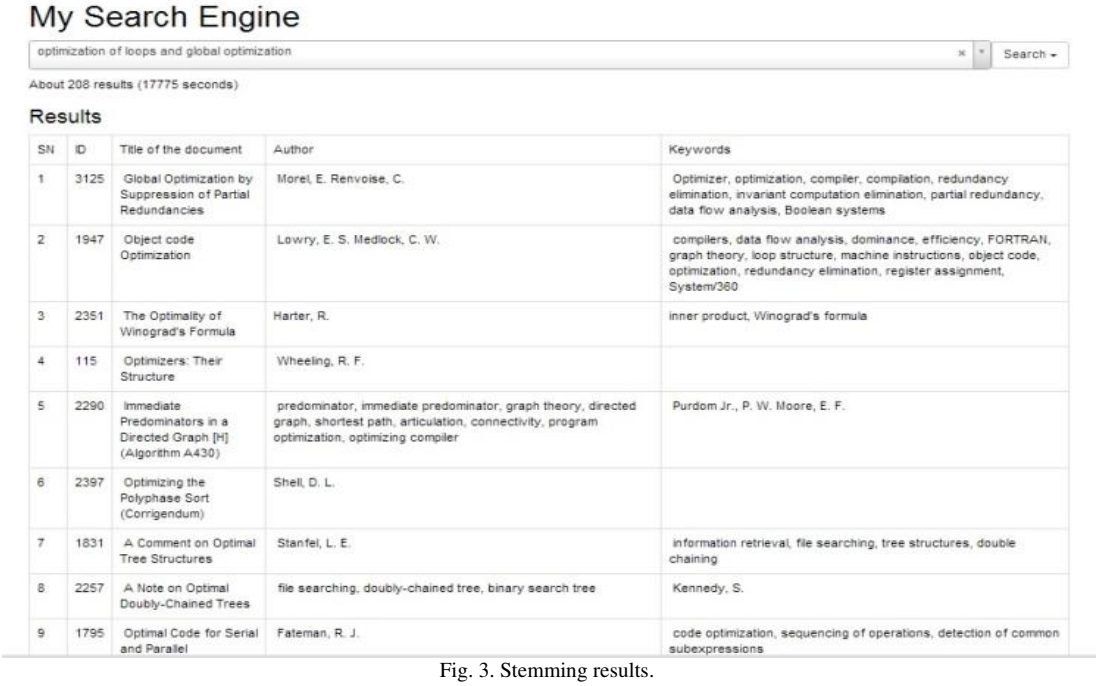
where N is the number of documents in the collection.

tf-idf therefore will be the multiplication of the term frequency and inverse document frequency as in (3) below.

$$\log(1 + tf_{t,d}) \text{ X } idf_t = log_{10} \frac{N}{df_t} \tag{3}$$

C. Evaluations

A prototype search engine was developed using the API approach that involved creating a back end using Visual Studio 2012. The engine's front-end was divided into two parts, namely the SearchEngine.API and SearchEngine.Web. The SearchEngine.API was sectioned into three major parts: Requests, Responses and Services. The request classes contain the algorithms that will implement all the codes that will be executed with the search engine. The response classes entail the instructions as to what the system should display when a request is made whereas the service classes entail the codes that need to be run once the request command is issued. The SearchEngine.Web basically was used to implement the user interface.


My Search Engine

optimization of loops and global optimization

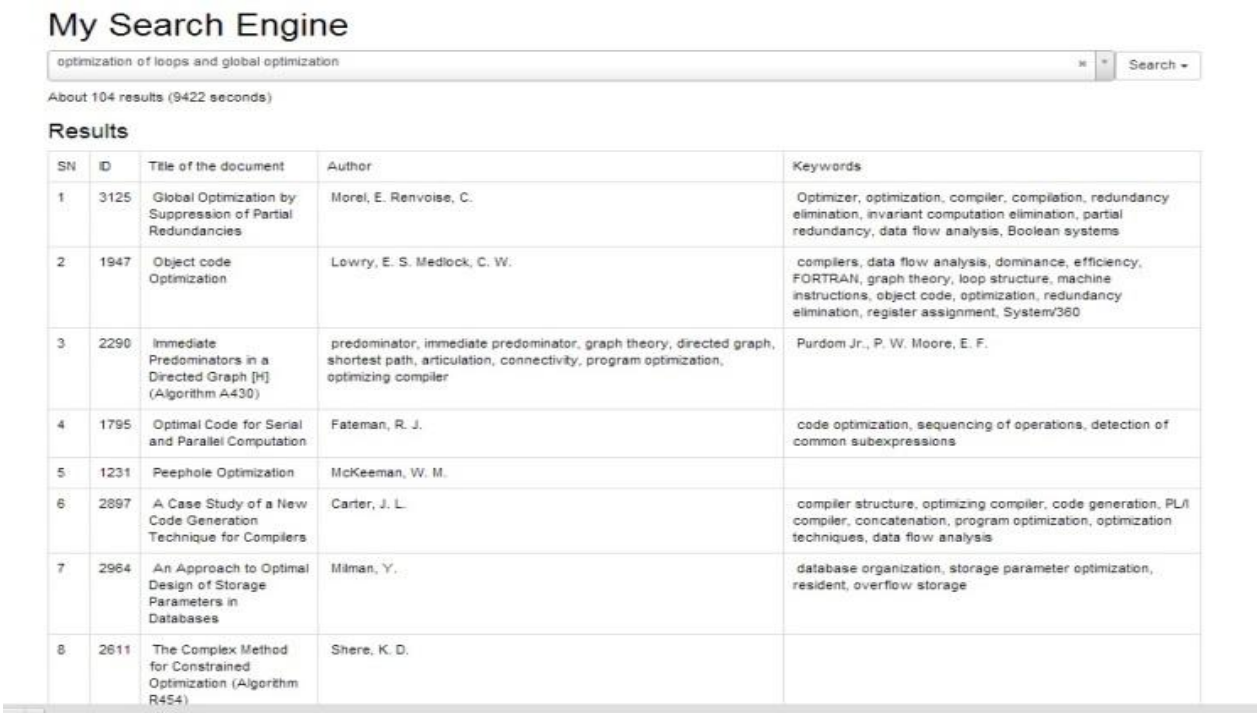
Search

About 208 results (17775 seconds)

Results

SN	ID	Title of the document	Author	Keywords
1	3125	Global Optimization by Suppression of Partial Redundancies	Morel, E. Renvoise, C.	Optimizer, optimization, compiler, compilation, redundancy elimination, invariant computation elimination, partial redundancy, data flow analysis, Boolean systems
2	1947	Object code Optimization	Lowry, E. S. Medlock, C. W.	compilers, data flow analysis, dominance, efficiency, FORTRAN, graph theory, loop structure, machine instructions, object code, optimization, redundancy elimination, register assignment, System/360
3	2351	The Optimality of Winograd's Formula	Harter, R.	inner product, Winograd's formula
4	115	Optimizers: Their Structure	Wheeling, R. F.	
5	2290	Immediate Predominators in a Directed Graph [H] (Algorithm A430)	predominator, immediate predominator, graph theory, directed graph, shortest path, articulation, connectivity, program optimization, optimizing compiler	Purdom Jr., P. W. Moore, E. F.
6	2397	Optimizing the Polyphase Sort (Corrigendum)	Shell, D. L.	
7	1831	A Comment on Optimal Tree Structures	Stanfel, L. E.	information retrieval, file searching, tree structures, double chaining
8	2257	A Note on Optimal Doubly-Chained Trees	file searching, doubly-chained tree, binary search tree	Kennedy, S.
9	1795	Optimal Code for Serial and Parallel	Fateman, R. J.	code optimization, sequencing of operations, detection of common subexpressions

Fig. 3. Stemming results.


My Search Engine

optimization of loops and global optimization

Search

About 104 results (9422 seconds)

Results

SN	ID	Title of the document	Author	Keywords
1	3125	Global Optimization by Suppression of Partial Redundancies	Morel, E. Renvoise, C.	Optimizer, optimization, compiler, compilation, redundancy elimination, invariant computation elimination, partial redundancy, data flow analysis, Boolean systems
2	1947	Object code Optimization	Lowry, E. S. Medlock, C. W.	compilers, data flow analysis, dominance, efficiency, FORTRAN, graph theory, loop structure, machine instructions, object code, optimization, redundancy elimination, register assignment, System/360
3	2290	Immediate Predominators in a Directed Graph [H] (Algorithm A430)	predominator, immediate predominator, graph theory, directed graph, shortest path, articulation, connectivity, program optimization, optimizing compiler	Purdom Jr., P. W. Moore, E. F.
4	1795	Optimal Code for Serial and Parallel Computation	Fateman, R. J.	code optimization, sequencing of operations, detection of common subexpressions
5	1231	Peephole Optimization	McKeeman, W. M.	
6	2897	A Case Study of a New Code Generation Technique for Compilers	Carter, J. L.	compiler structure, optimizing compiler, code generation, PL/I compiler, concatenation, program optimization, optimization techniques, data flow analysis
7	2964	An Approach to Optimal Design of Storage Parameters in Databases	Milman, Y.	database organization, storage parameter optimization, resident, overflow storage
8	2611	The Complex Method for Constrained Optimization (Algorithm R454)	Shere, K. D.	

Fig. 4. Lemmatization results.

The retrieval performances were tested and comparisons were made using the Communications of the Association for Computing Machinery (CACM) collection. The collection contains 3204 documents, 64 queries and the relevance judgements for the documents. The queries were tested and filtered in order to choose queries that require language processing. This resulted in 15 queries.

Fig. 2 below shows the screen display for "*optimization of loops and global optimization"* queryusing the baseline technique. A total of 95 documents were retrieved for this particular query.

Similarly, when the same query was used for stemming a total of 208 documents were retrieved. This far exceeds the number of documents retrieved by the baseline technique though the relevance is undetermined.

Finally, lemmatization produced 104 documents for the same query. Fig. 3 and Fig. 4 depict the screen displays for stemming and lemmtization, respectively. From these figures, it can also be noted that although the query was the same, different results were produced due to the varying processing techniques.

The results were then compared against the relevance judgements provided in the CACM collection, and the relevance precisions were calculated.

D. Evaluation Metrics

In order to assess the performance of the system, the relevant documents retrieved during the evaluation were matched against the relevant judgements provided along with the CACM collection. Mean Average Precision (MAP) was used to evaluate document relevancy at the top 10 and 20 document levels.

MAP was calculated by dividing the average precisions with the number of queries (i.e. 15 in this study). MAP is sensitive to the entire ranking of the documents retrieved for a search query. As it also combines both recall-oriented and precision-oriented aspects of the search engine, the overall performance of the search engine can be evaluated efficiently. In general, MAP can be represented as follows:

$$\frac{\#\ relevant\ documents\ retrieved}{Total\ relevant\ documents} \quad (4)$$

Additionally, precision histograms were also created to compare the algorithms' performances on each of the 15 queries. Pair-wise comparisons were also made to assess the significance of the differences. Results were considered to be significant at $p < 0.05$.

IV. Results and Discussion

A. Mean Average Precisions

Table I shows the MAP for top 10 and 20 documents for all the three techniques.

TABLE I: MAP for Top 10 and 15 Documents

Techniques	@10	@20
Baseline	0.601	0.490
Stemming	0.614	0.510
Lemmatization	0.623	0.544

From the table, it can be noted that both stemming and lemmatization performed better than the baseline technique at both the document levels. This indicates that when queries are processed using language modeling techniques, they yield documents that are more relevant compared to queries which are not processed. This is similar with studies that have reported language models to improve document retrievals [6]–[11].

A comparison between stemming and lemmatization indicates that lemmatization outperformed stemming. Pair-wise comparisons however revealed that the precision differences between these techniques to be insignificant. This is probably because lemmatization is more advanced in the

sense that it takes care of additional analysis that is not supported by stemming. For instance, lemmatization looks at the synonyms of a word unlike stemming. This may result in more relevant documents.

B. Histograms

The histograms for all the 15 queries are shown in this section. For comparisons against the baseline algorithm, the histograms for top 20 documents are shown. Fig. 5 shows the histogram for stemming and baseline. It can be noted that stemming performed better than the baseline for 60% (i.e. 9/15) of the queries. The remaining 40% were on the same level.

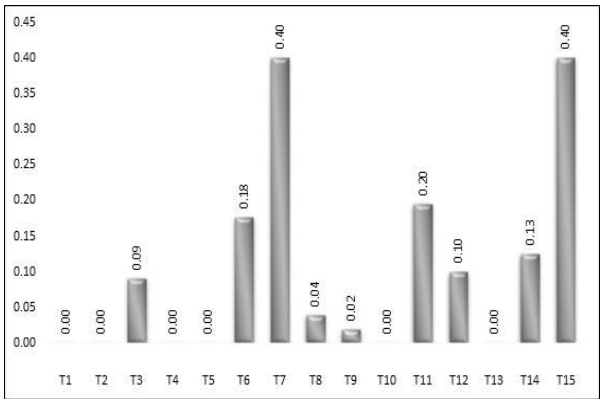


Fig. 5. Stemming-baseline histogram.

Similarly, Fig. 6 shows that lemmatization performed better than the baseline for 40% (i.e. 6/15) queries. The baseline performed better than lemmatization for a single quey (i.e. T9) whilst the rest were retrieved at the same precision levels. Both histograms for stemming and lemmatization show that the performance matches their precisions as indicated in Table I, in which stemming and lemmatization performed better than the baseline algorithm.

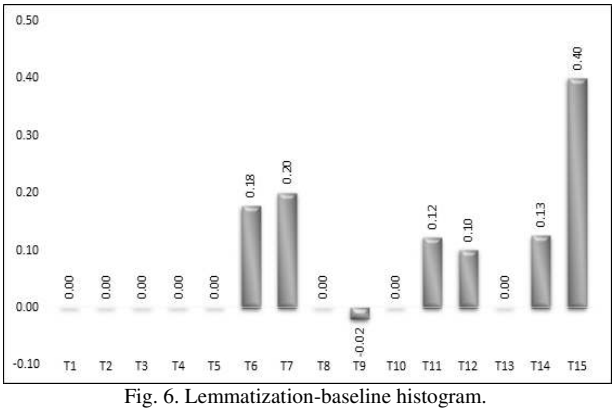


Fig. 6. Lemmatization-baseline histogram.

Fig. 7 and Fig. 8 depict the histograms for stemming against lemmatization at top 10 and 20 document levels, respectively. Although most of the queries were retrieved at the same level, lemmatization performed slightly better than stemming (i.e. 13%). We believe this is due to the nature of the test collection that was used in this study whereby not many queries required lemmatization process to take place. Although the differences are insignificant, nevertheless lemmatization outperformed stemming.

Overall, the study found language processing techniques improve the relevancy of document retrievals compared to the baseline algorithm. Lemmatization on the other hand, yielded more relevant results when compared to stemming. The study is not without its limitations, with the main drawback being the test collection. During the evaluation, it was found that most of the queries were not suitable to be used for a language model as they do not contain items that require stemming or lemmatization. Future studies should look into using other test collections.

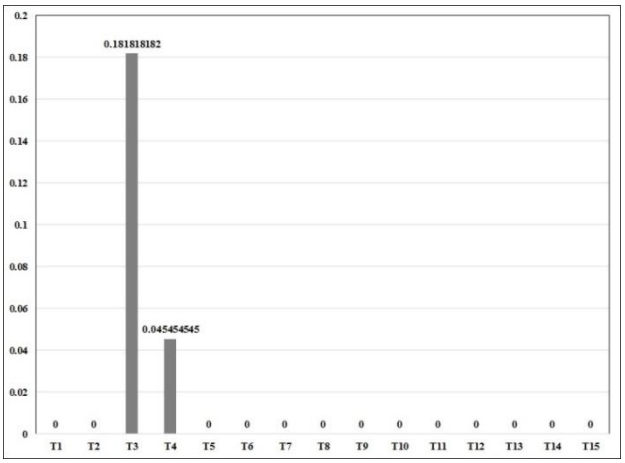


Fig. 7. Lemmatization-stemming for top 10.

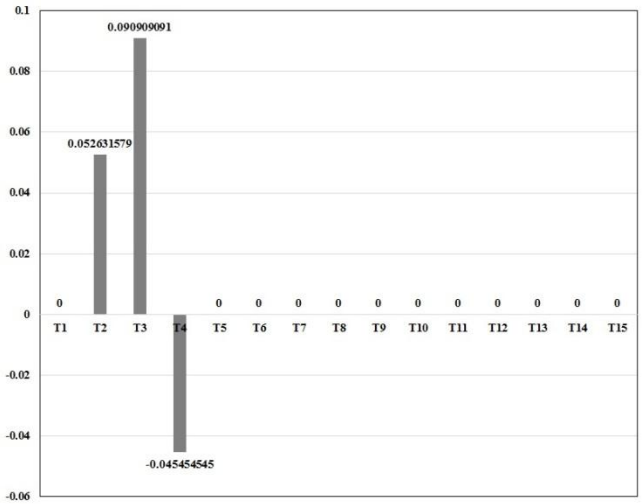


Fig. 8. Lemmatization-stemming for top 20.

REFERENCES

[1] G. Chowdhury and S. Chowdhury, *Introduction to digital libraries*, Facet publishing, 2002.

[2] N. J. Belkin, "Anomalous states of knowledge as a basis for information retrieval," *Canadian Journal of Information Science*, vol. 5, pp. 133-143, 1980.

[3] H.S. Heaps, *Information Retrieval, Computational and Theoretical Aspects,* Academic Press, 1978.

[4] R. Baeza-Yates and B. Ribeiro-Neto, *Modern information retrieval*, vol. 463, New York: ACM press, 1999.

[5] J. B. Lovins, "Development of a stemming algorithm," *Mechanical Translation and Computational Linguistics*, vol. 11, pp. 22-31, 1968.

[6] L. S. Larkey, L. Ballesteros, and M. E. Connell, "Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis," in *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 2002, pp. 275-282.

[7] J. Xu, A. Fraser, and R. Weischedel, "Empirical studies in strategies for Arabic retrieval," in *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 2002, pp. 269-274.

[8] M. Wechsler, P. Sheridan, and P. Schäuble, "Multi-Language Text Indexing for Internet Retrieval," in *Proc. 5th RIAO Conference*, *Computer-Assisted Information Searching on the Internet*, vol. 5 pp. 217-232, 1997.

[9] D. A. Hull, "Stemming algorithms: A case study for detailed evaluation," *Journal of the American Society for Information Science*, vol. 47, pp. 70-84, 1996.
[10] R. Hooper, and C. Paice. (December 2013). The Lancaster stemming algorithm. [Online]. Available: http://www.comp.lancs.ac.uk/computing/research/stemming/
[11] O. Ozturkmenoglu and A. Alpkocak, "Comparison of different lemmatization approaches for information retrieval on Turkish text collection," *Innovations in Intelligent Systems and Applications (INISTA) International Symposium*, pp. 1-5, 2012.
[12] D. Gupta, R. Kumar, R. Yadav, and N. Sajan, "Improving Unsupervised Stemming by using Partial Lemmatization Coupled with Data-based Heuristics for Hindi," *International Journal of Computer Applications*, vol. 38, pp. 1-8, 2012.

V. Balakrishnan received her PhD in the field of ergonomics in 2009 from Multimedia University, Malaysia. Both her master and bachelor degrees were from University of Science, Malaysia.

She is currently affiliated with the Faculty of Computer Science and Information Technology, University of Malaya as a Senior Lecturer. Most of her research works are in the field of data engineering, opinion mining, information retrieval and health informatics.

Dr. Balakrishnan is also a member of the Medical Research Support (Medicres) group, IACSIT and Global Science and Technology Forum.

E. Llyod-Yemohattained received her bachelor's degree from FTMS KL (University of East London) and is currently pursuing her master's degree in the field of management information systems at Faculty of Computer Science and Information Technology, University of Malaya, Malaysia.

Her research work is mainly on information retrieval, particularly involving language modeling techniques.