

IMPROVING DOCUMENT RELEVANCY USING INTEGRATED LANGUAGE MODELING TECHNIQUES

Vimala Balakrishnan¹, Norshima Humaidi² and Ethel Lloyd-Yemoh³

^{1,3} Department of Information Systems,
Faculty of Computer Science and Information Technology,
University of Malaya, 50603 Kuala Lumpur, Malaysia.

² Faculty of Business and Management, Universiti Teknologi MARA, Puncak Alam Campus.

E-mail: vimala.balakrishnan@um.edu.my¹, norshima958@puncakalam.uitm.edu.my²,
ethel_lloyd@siswa.um.edu.my³

ABSTRACT

This paper presents an integrated language model to improve document relevancy for text-queries. To be precise, an integrated stemming-lemmatization (S-L) model was developed and its retrieval performance was compared at three document levels, that is, at top 5, 10 and 15. A prototype search engine was developed and fifteen queries were executed. The mean average precisions revealed the S-L model to outperform the baseline (i.e. no language processing), stemming and also the lemmatization models at all three levels of the documents. These results were also supported by the histogram precisions which illustrated the integrated model to improve the document relevancy. However, it is to note that the precision differences between the various models were insignificant. Overall the study found that when language processing techniques, that is, stemming and lemmatization are combined, more relevant documents are retrieved.

Keywords: *Information retrieval, document relevancy, language modeling, stemming, lemmatization, mean average precision*

1.0 INTRODUCTION

The use of internet all over the world has caused information size to increase, hence making it possible for large volumes of information to be retrieved by the users. However, this phenomenon also makes it difficult for users to find relevant information, therefore proper information retrieval techniques are needed. 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*” [1]. In short, information retrieval aims to provide users with those documents that will satisfy their information need.

Many information retrieval algorithms were proposed, and some of the popular ones include the traditional Boolean model (i.e. based on binary decisions), vector space model (i.e. compares user queries with documents found in collections and computes their similarities), and probabilistic model (i.e. based on the probability theory to model uncertainties involved in retrieving data), among others. Over the years, information retrieval has evolved to include text retrieval in different languages, and thus giving birth to language models. The language model is particularly concerned with identifying how likely it is for a particular string in a specific language to be repeated [2]. A popular technique used in the language model is the N-gram model which predicts a preceding word based on previous N-1 words [3]. Other popular techniques include stemming and lemmatization.

Stemming relates to deciding which documents in a collection should be retrieved to satisfy a user's need for information. Stemming is used to remove derivational suffixes as well as inflections (i.e. adding inflectional morphemes (i.e. smallest unit of meaning) to a word which indicates grammatical information, so that word variants can be conflated into the same stems). Stop-words are used in eliminating high frequency terms when documents are being searched, and then the stemming procedure can be used to conflate all word variants back to their basic rules using an algorithm [4]. Words found in documents and user queries usually have different possible meanings, and therefore care must be taken when retrieving information. Different variants of a word might appear in a document and it is up to the retrieval system to use natural language processing methods in order to retrieve the required texts in document collections. The lemmatization, in contrary, uses vocabulary and morphological analysis of words to process a query. It removes inflectional endings and analyzes if query words are used as verbs or nouns. Lemmatization also helps to match synonyms using thesaurus, so that when one searches for “hot” the word “warm” is matched as well.

Both stemming and lemmatization play very important roles in increasing relevance and recall capabilities of a retrieval system. When these techniques are used, the number of indexes used is reduced as the system will be using one index to present a number of similar words having the same root. Furthermore, the dictionary size is also reduced as all the distinct terms representing a set of documents will be replaced with a single term. Most of the studies in this field focused on stemming and lemmatization separately, and very few studies have looked into integrating them. The current study aims to propose an integrated model, namely stemming-lemmatization (S-L) model to improve document retrievals. The performance of the S-L model was compared with a baseline algorithm (i.e. with no language processing), and also with stemming and lemmatization models.

The remainder of the paper is structured as follows: the following section explains stemming, followed by lemmatization and the integrated S-L model. The evaluation setup is presented next, followed by results and discussion. The conclusion and future works conclude the paper.

2.0 STEMMING

As stated previously, stemming has a significant effect on both the efficiency and the effectiveness of information retrieval. Studies that have used stemming mechanisms reported better retrieval results, not only for English based texts, but also those in other languages. For instance, an experiment was conducted on multilingual information retrieval using the thesaurus-based query expansion technique, with results indicating the proposed model to be able to retrieve Italian documents when queries are written in German [5]. Similarly, a combination of translation probabilities from different sources was used to improve search results across different languages using a stemming algorithm on the TREC collection [6].

In another study, Ref. [7] developed a stemmer for Tigrinya (i.e. language spoken in the east African countries of Eritrea and Ethiopia) words, by combining rule-based and dictionary based stemming techniques. The study found the hybrid approach to have an average accuracy of approximately 89.3%. A rule-based stemmer for Bengali language which uses stem dictionary for further validation was developed in [8], whereas an enhanced stemmer for Arabic was built by integrating light and dictionary based stemming [9]. The latter study found their stemmer to produce an average accuracy of approximately 96%. A study investigating the effect of stemming on Swedish texts found stemming improved retrieval precisions by at least 15% and relative recall by 18%, depending on the set of rules and the document collection [10].

Finally, researchers in [11] compared the effects of four different stemming scenarios on Turkish texts, that is, without stemming, simple word truncation, the successor variety method and a lemmatizer-based stemmer. The study found all the three stemming techniques produced similar retrieval performances for the Turkish texts. Many other studies have also focused in developing stemmers for languages other than English, such as the Malay language [12], Indonesian language [13], Slovene [14], Turkish [10], German [15] and Dutch [16] etc., with results showing improved retrieval performance when stemmers are used.

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, such as the Paice/Husk, Porter, Lovins, Dawson and Krovetz stemming algorithms. The Porter's stemming algorithm is widely used [17], hence it is selected for the current study.

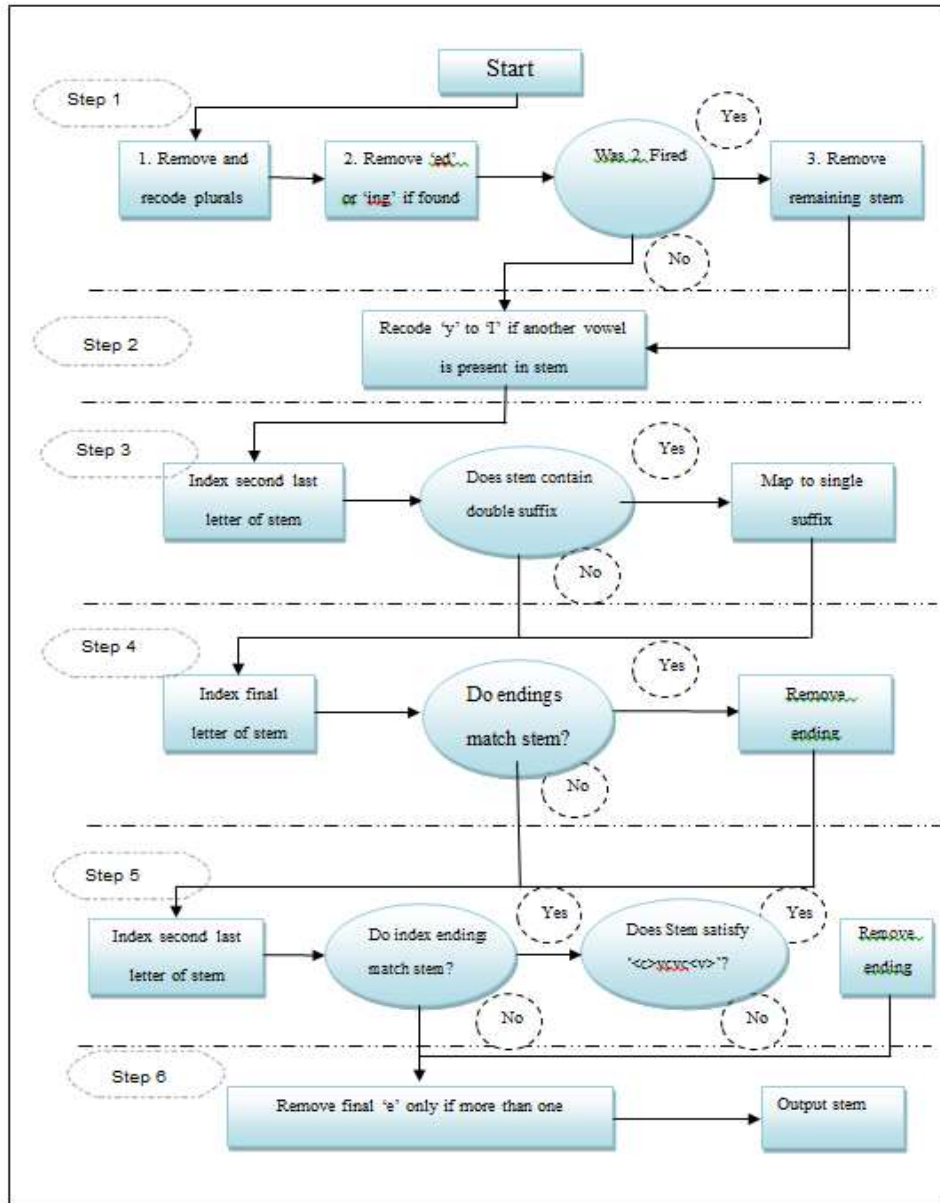


Fig. 1 Porter's stemmer [17]

Porter's stemming uses a linear step stemmer which seeks to remove suffixes of words (see Fig.1). It basically contains six steps, and at each step there are rules that need to be followed. Whenever conditions in the rules are met, the program executes and the new stem goes through the process again. This loop continues until a suitable stem is found [17]. The stemmer deals with vowels (i.e. A, E, I, O, U and Y, only if Y is preceded by a consonant), and consonants. An example of 'Y' being considered as a vowel is in the word "fry", where the consonants will be 'F' and 'R' as compared to the word "buy" where 'B' and 'Y' can be said to be consonants. Consonants are represented by 'C' and vowels by 'V'; therefore words are represented by Formula (1) below:

$$[C] (VC)^m [V] \quad (1)$$

Where

[] \rightarrow optional presence of V and C

(VC) m \rightarrow the number of times VC repeats

In Fig. 1, the stemmer begins at step 1 where past participles and plurals are dealt with. Step 1.1 will work on plurals (i.e. words ending with 'es' or 's'), step 1.2 will be responsible for checking 'ed' and 'ing' and if these exist, it will then check if the stem obtained after this removal is enough to form a word. This leads to the final part where the new stem is transferred to step 2 where 'y' is changed to 'i'. Step 3 transforms double suffixes into single ones, followed by step 4 which removes other endings. Step 5 will check if conditions for the stem are met, and finally the output stem is attained in step 6 [17].

3.0 LEMMATIZATION

Lemmatization basically removes inflectional endings and return the base or dictionary form of a word. It has also been used in several languages for information retrieval. For instance, three lemmatizers were compared using Turkish language [18]. The authors used (i) Oflazer's morphological analyzer (OMA) which uses a two level morphology to build lexicon for a language, (ii) a fixed length truncation in which the first five and seven characters of a word are used as lemma whereas the rest are truncated, (iii) a dictionary-based Turkish lemmatizer and (iv) Zemberek which is a parser that loads binary roots and builds special direct acyclic word graphs with it. Their findings showed that lemmatization improved information retrieval when a minimum number of terms and maximum lemma lengths are used. They also reported their model to have a better performance than other approaches (i.e. no-lemmatization approach, OMA, Zemberek and fixed length truncation).

In a recent medical sector study, lemmatization along with text processing, information extraction and query expansion were used, with results indicating improved precisions of the search results [19]. In another study, a self-learning lemmatizer that could process German documents using a full-form lexicon was designed [20]. Their results showed that the self-learning context-aware lemmatizer for German was a great time saver compared to Wikitionary (i.e. a multilingual web-based project to create a free content dictionary of all words in all languages).

A non-statistical Arabic lemmatizer using different knowledge resources to generate lemma and its relevant features was investigated in [21]. The authors' experimental results revealed their algorithm to achieve an accuracy of 94.8%. Other similar studies on Arabic texts were conducted using light lemmatizers [22, 23]. For example, Ref. [23] presented an approach in which the algorithm searches text words, and finds one of the predetermined stop words to differentiate between verbs and nouns.

In a recent study, a dictionary- and corpus-independent statistical lemmatizer was developed to deal with out-of-vocabulary problem. The proposed algorithm was evaluated on four different datasets in Finnish, Swedish, German and English, and found it to reach 88 – 108% of the gold standard performance of a commercial lemmatizer [24]. Another study on Hungarian language was carried out using NLP-based lemmatization, and compared with simple stemmers [25], with results showing improved mean average precisions for the former technique.

One of the problems of lemmatization is the compound word (i.e. words made of two or more words). This is because compound words will be split into their components during lemmatization, and each component treated as a single word. This may not be accurate in retrievals, therefore attempts have been made to combine both lemmatization and stemming. For instance, some studies found stemming used with clustering algorithms to be beneficial in English texts [26], and also other languages [27, 28]. Gupta et al. [29] combined stemming with partial lemmatization for Hindi language with results indicating significant improvements than other traditional approaches. Another study compared stemming and lemmatization in clustering Finnish text documents, with results indicating the use of lemmatization to be better than stemming [30].

Stemming and lemmatization play important roles in increasing relevance capability of a retrieval system. When these techniques are used, it reduces the number of indexes used as it uses one index to present a number of similar words, which have the same root or stem. A lot of studies have looked into either using stemming or lemmatization and practical combinations, but fully integrating both of these algorithms is lacking. This is what inspired this study on integrating the stemming and lemmatization techniques.

4.0 STEMMING-LEMMATIZATION MODEL

Fig. 2 shows the overall flow for the S-L model.

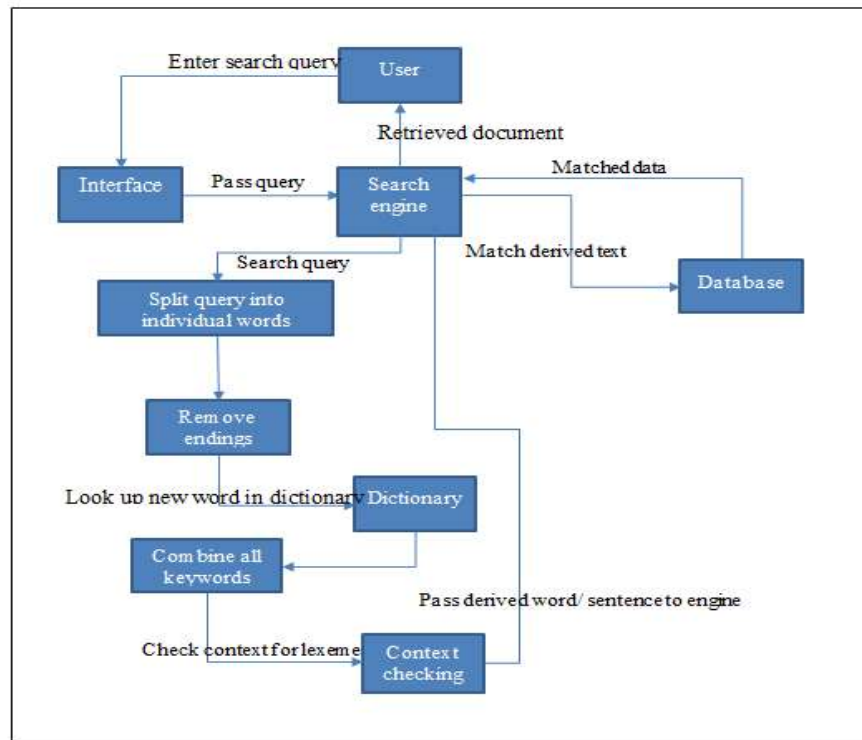


Fig. 2. The Stemming-Lemmatization flow

In the proposed S-L model, the Porter's stemmer was used together with LemmaGen, which is a prebuilt lemmatizer. Fig. 2 shows that when users enter a search query, the system will first look for the whole query text, and the retrieved documents will be kept in a buffer. If the query contains more than one word, the system will then split the query into individual words. All suffixes are removed from each word and the root-word that is returned is passed to the dictionary to ensure that it is an actual root word, and not a stem. Once this is completed, the words will then be combined and their context searched to ensure that the context in which the query was used has not been changed. This is handled by the *Lzma.dll* file that is included in the LemmaGen. Once this check is completed, the derived query is returned to the search engine which will in turn pass it to the database for query-document matching. All documents that match the words in the derived query are then retrieved and displayed to the user.

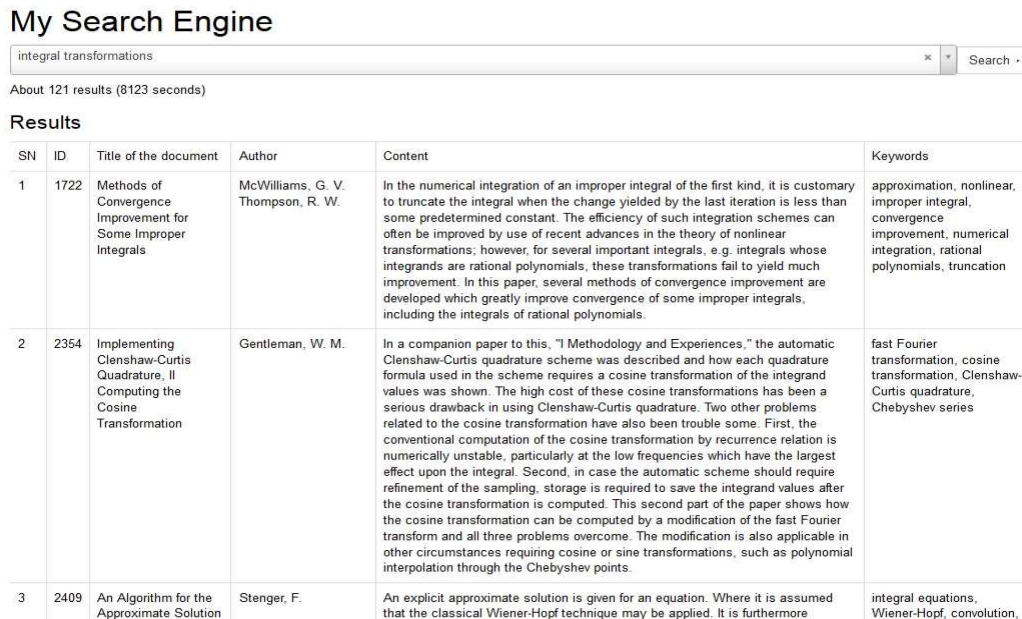
5.0 EVALUATION SETUP

The models in the current study were compared as follows:

- *S-L versus tf-idf* (i.e. the term frequency-inverse document frequency (tf-idf) ranking algorithm was used as the baseline in this study. This traditional ranking algorithm checks the retrieved document to see how frequent the words in the search query appears in the document. The document is considered to be more relevant if it contains more occurrences of the query word. Further details of tf-idf can be found in [31].
- *S-L versus stemming* (i.e. the integrated model is compared against stemming)
- *S-L versus lemmatization* (i.e. the integrated model is compared against lemmatization)

All the models above were compared at top 5, 10 and 15 document levels. The queries were selected manually as not all of them required language processing to take place, thus resulting in final 15 queries.

A prototype search engine was developed in order to evaluate the proposed model. The test collection used was the Communications of the ACM (CACM), which contains 3204 documents, 64 queries, a list of common words that can be used as stop-words and the relevant judgements. Fig. 3 shows a sample screen display for “integral transformations”.



The screenshot shows a search engine interface with the title "My Search Engine". The search bar contains the text "integral transformations" and a "Search" button. Below the search bar, it indicates "About 121 results (8123 seconds)". The results are displayed in a table with the following columns: SN, ID, Title of the document, Author, Content, and Keywords.

SN	ID	Title of the document	Author	Content	Keywords
1	1722	Methods of Convergence Improvement for Some Improper Integrals	McWilliams, G. V. Thompson, R. W.	In the numerical integration of an improper integral of the first kind, it is customary to truncate the integral when the change yielded by the last iteration is less than some predetermined constant. The efficiency of such integration schemes can often be improved by use of recent advances in the theory of nonlinear transformations; however, for several important integrals, e.g. integrals whose integrands are rational polynomials, these transformations fail to yield much improvement. In this paper, several methods of convergence improvement are developed which greatly improve convergence of some improper integrals, including the integrals of rational polynomials.	approximation, nonlinear, improper integral, convergence improvement, numerical integration, rational polynomials, truncation
2	2354	Implementing Clenshaw-Curtis Quadrature. II Computing the Cosine Transformation	Gentleman, W. M.	In a companion paper to this, "I Methodology and Experiences," the automatic Clenshaw-Curtis quadrature scheme was described and how each quadrature formula used in the scheme requires a cosine transformation of the integrand values was shown. The high cost of these cosine transformations has been a serious drawback in using Clenshaw-Curtis quadrature. Two other problems related to the cosine transformation have also been trouble some. First, the conventional computation of the cosine transformation by recurrence relation is numerically unstable, particularly at the low frequencies which have the largest effect upon the integral. Second, in case the automatic scheme should require refinement of the sampling, storage is required to save the integrand values after the cosine transformation is computed. This second part of the paper shows how the cosine transformation can be computed by a modification of the fast Fourier transform and all three problems overcome. The modification is also applicable in other circumstances requiring cosine or sine transformations, such as polynomial interpolation through the Chebyshev points.	fast Fourier transformation, cosine transformation, Clenshaw-Curtis quadrature, Chebyshev series
3	2409	An Algorithm for the Approximate Solution	Stenger, F.	An explicit approximate solution is given for an equation. Where it is assumed that the classical Wiener-Hopf technique may be applied. It is furthermore	integral equations, Wiener-Hopf, convolution,

Fig. 3 Screen display for a search query

6.0 EVALUATION METRICS

The effectiveness of the proposed model was evaluated using precision metric and precision histograms. The average precision (AP) is basically the mean of the precision scores after each relevant document is retrieved, as depicted in Formula (2) below:

$$AP = \frac{\sum_{i=1}^N (P@k \cdot rel(k))}{\text{total relevant documents for a query}} \quad (2)$$

where N is the number of retrieved documents, and $rel(k)$ indicates whether the k^{th} document is relevant or not. The AP is calculated for each query, and then they are averaged over all the queries to produce the mean average precision (MAP).

The precision histograms were created to compare how well a model performs compared to another for each query, based on Formula (3) below: [4]

$$\mathbf{RPrecision}_{(A/B)} = \mathbf{RP}_A(i) - \mathbf{RP}_B(i) \quad (3)$$

where

$\mathbf{RPrecision}_{(A/B)} \rightarrow$ is R-precision of algorithm A over algorithm B.

$\mathbf{RP}_A(i) \rightarrow$ refers to the R-Precision of algorithm A when the i^{th} query is run.

$\mathbf{RP}_B(i) \rightarrow$ refers to the R-Precision of algorithm B when the i^{th} query is run.

7.0 RESULTS AND DISCUSSION

As stated in the preceding section, the models were compared at top 5, 10 and 15 levels for MAP. Table 1 depicts the MAP values for all the models in the study.

Table 1. MAP values for all the models

Algorithms	MAP ₅	MAP ₁₀	MAP ₁₅
tf-idf	0.693	0.609	0.522
Stemming	0.708	0.613	0.568
Lemmatization	0.715	0.623	0.579
S-L	0.723	0.628	0.610

From Table 1, it can be noted that all the language processing models outperformed the baseline algorithm. This is expected as the baseline algorithm returns results based on the search query only, without taking any further processing into consideration such as stemming or lemmatization [20, 29].

Comparisons across the language models show that the integrated S-L model has the highest precisions at all the three document levels as opposed to stemming and lemmatization alone. It can also be noted that the precision values decrease as the recall (i.e. the number of documents retrieved) increase. This is a common phenomenon in information retrieval systems. Further statistical analysis, particularly pair-wise comparisons were made between these techniques and findings revealed the differences to be insignificant at all three levels (i.e. $p > 0.05$).

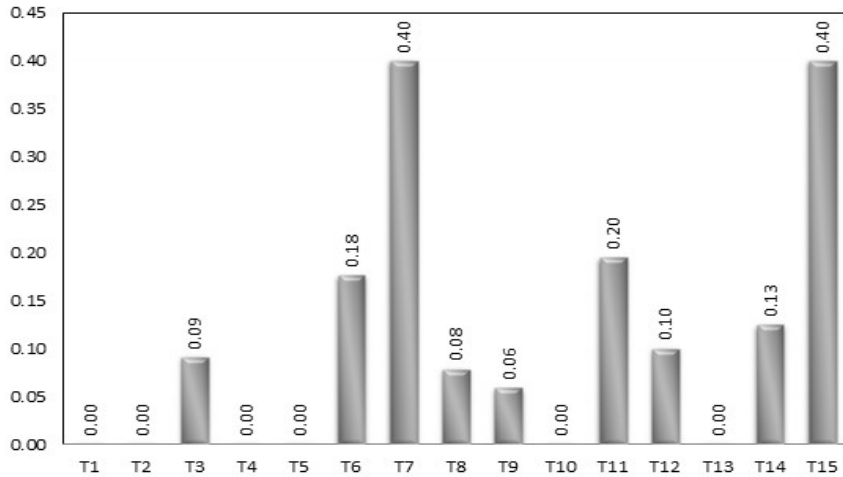


Fig. 4. S-L Model-Baseline (tf-idf)

The precision histograms are only shown between the proposed S-L model against the baseline, stemming and lemmatization at top 10 level. Fig. 4 illustrates the histogram for the S-L model against tf-idf, with the S-L model performing 60% better than the baseline. In other words, nine out of fifteen queries were retrieved with better precisions compared to the baseline algorithm. The rest of the queries were retrieved at the same precisions. This finding tallies with the MAP values depicted in Table 1, in which the S-L model outperformed the baseline algorithm.

Fig. 5 shows the histogram for S-L model against stemming, with the former performing better than stemming for four queries (i.e. T2, T3, T4 and T5). Stemming performed better for T11 (negative bar) whilst the remaining queries were retrieved at the same precision levels.

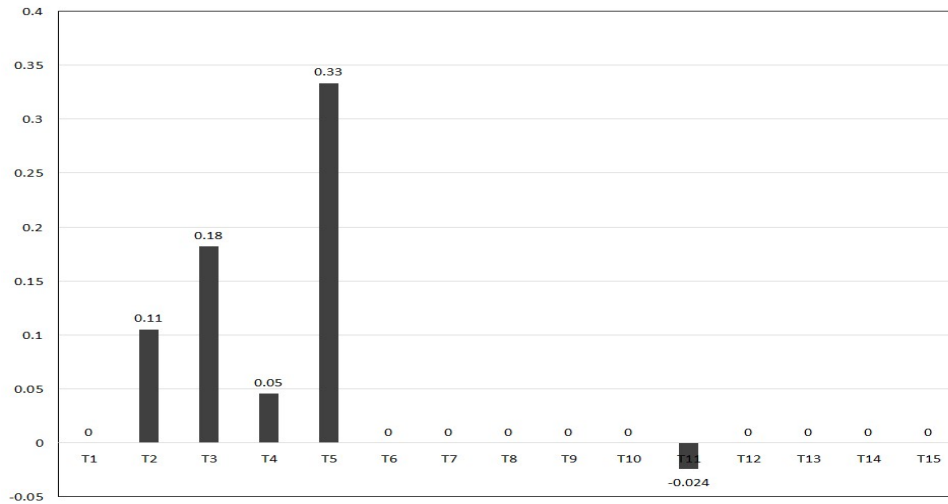


Fig. 5. S-L Model-Stemming

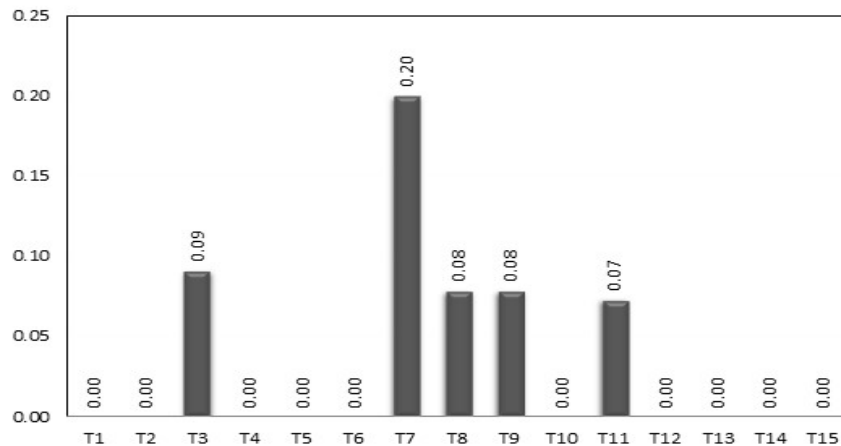


Fig. 6. S-L Model-Lemmatization

Finally, Fig. 6 shows the histogram for S-L model against lemmatization. The S-L model again performed better than lemmatization for approximately 34% of the queries.

Overall the evaluations show that the integrated model work better than stemming and lemmatization at all three document levels, that is, at top 5, 10 and 15. Although the precision differences were insignificant, it is believed that the precisions can be further improved when more queries were used in testing. This is the major drawback of the current study as most of the queries in the CACM collection were not suitable for language models.

8.0 CONCLUSION AND FUTURE WORK

This study looked into the stemming and lemmatization techniques with a further step of combining these two techniques to improve precision in information retrieval. A Stemming-Lemmatization model (S-L model) was created to improve the document relevancy, and assessment of the efficiency of the model in improving document relevancy was made as well. Results generally indicated that when language processing is done, the document relevancy improves. Additionally, it was also found that when both stemming and lemmatization are combined, more relevant documents are retrieved. The main drawback of the study is the lack of appropriate queries provided in the CACM collection, hence future studies should look into the possibility of evaluating the integrated model with more extensive test collections.

9.0 ACKNOWLEDGMENT

The authors extend their heartfelt gratitude to University of Malaya for supporting this study (UMRG – RP028A-14AET)

REFERENCES

- [1] N. J. Belkin, "Anomalous states of knowledge as a basis for information retrieval", *Canadian Journal of Information Science*, Vol. 5: pp. 133-143., 1980
- [2] J. M. Ponte, and W. B. Croft, "A language modeling approach to information retrieval", *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998.

- [3] D. Jurafsky, J. H. Martin, A. Kehler, K. Vander Linden, and N. Ward, “*Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*”, Upper Saddle River: Prentice Hall, 2000
- [4] R. Baeza-Yates, and B. Ribeiro-Neto, B, “ *Modern information retrieval,*” New York, USA: ACM Press, 1999
- [5] P. Sheridan, and J. P. Ballerini, “Experiments in multilingual information retrieval using the SPIDER system”, *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 1996
- [6] J. Xu, A. Fraser, and R. Weischedel, “Empirical studies in strategies for Arabic retrieval”, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 2002
- [7] O. O. Ibrahim and Y. Mikami, “Stemming Tigrinya Words for Information Retrieval,” *Proceedings of COLING 2012: Demonstration Papers*, pp. 345–352, 2012
- [8] S. Sarkar and S. Bandyopadhyay, “Study on Rule-Based Stemming Patterns and Issues in a Bengali Short Story-Based Corpus”, In 7th International Conference on Natural Language Processing (Poster), 2009
- [9] Y. Alhanini, and M. J. A. Aziz, “The Enhancement of Arabic Stemming by Using Light Stemming and Dictionary-Based Stemming,” *Journal of Software Engineering and Applications*, Vol. 4: 522-526, 2011
- [10] J. Carlberger, H. Dalianis, M. Hassel, and O. Knutsson , Improving Precision in Information Retrieval for Swedish using Stemming, 2001, Available at : http://www.nada.kth.se/~xmartin/papers/Stemming_NODALIDA01.pdf
- [11] F. Can, S. Kocerberber, E. Balcik, C. Kaynak, H. C. Ocalan, and O. M. Vursavas, “Information retrieval on Turkish texts”, *Journal of the American Society for Information Science and Technology*, Vol. 59: 407-421, 2008
- [12] F. Ahmad, M. Yusoff, and T. M. T. Sembok, “Experiments with a Stemming Algorithm for Malay Words,” *Journal of The American Society for Information Science*, Vol. 47:909–918, 1996
- [13] B. Nazief and M. Adriani, “Confix Stripping: Approach to Stemming Algorithm for Bahasa Indonesia,” Technical report, Faculty of Computer Science, University of Indonesia, Depok, 1996
- [14] M. Popovic and P. Willett “The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data,” *Journal of the American Society for Information Science*, Vol. 43(5): 384–390, 1992
- [15] M. Braschler, and B. Ripplinger, “How effective is stemming and decomposing for German text retrieval?”, *Information Retrieval*, Vol.7:291-316, 2004
- [16] W. Kraaij and R. Pohlman, “Viewing Stemming as Recall Enhancement,” *In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 40–48, 1996
- [17] M. F. Porter, “An algorithm for suffix stripping”, *Electronic Library and Information Systems*, Vol. 14: pp. 130-137, 1980
- [18] O. Ozturkmenoglu, and A. Alpkocak, “Comparison of different lemmatization approaches for information retrieval on Turkish text collection”, *Innovations in Intelligent Systems and Applications (INISTA) 2012 International Symposium*, IEEE, 2012

- [19] B. King, L. Wang, I. Provalov, and J. Zhou, "Cengage Learning at TREC 2011 Medical Track", 2011, *Proceedings of TREC*, Available at : http://www-personal.umich.edu/~benking/resources/papers/cengage_medical.pdf
- [20] P. Perera, and R. Witte, "A Self-Learning Context-Aware Lemmatizer for German", *Information Systems Frontiers*, Vol. 8: pp. 47-57, 2006
- [21] T. El-Shishtawy and F. El-Ghannam, "Comparison of Different Lemmatization Approaches for Information Retrieval on Turkish Text Collection", 2012, Available at <http://arxiv.org/ftp/arxiv/papers/1203/1203.3584.pdf>
- [22] F. Hammouda and A. Almarimi, "Heuristic Lemmatization for Arabic Texts Indexation and Classification", *Journal of Computer Science*, Vol. 6 (6): 660-665, 2010.
- [23] E. Al-Shammari E., and J. Lin, "A Novel Arabic Lemmatization Algorithm," in *Proceedings of the second workshop on Analytics for noisy unstructured text data*, ACM, 2008
- [24] A. Loponen, and K. Järvelin, "A Dictionary- and Corpus-Independent Statistical Lemmatiser for Information Retrieval in Low-Resource Languages", In: Agosti, M. & al. (Eds.), *Multilingual and Multimodal Information Access Evaluation*, Proceedings of the International Conference in the Cross-Language Evaluation Forum, Heidelberg: Springer, pp. 3-14, 2010
- [25] P. Halacsy, "Benefits of deep NLP-based lemmatization for information retrieval" 2006, Available at <http://ceur-ws.org/Vol-1172/CLEF2006wn-adhoc-Halacsy2006.pdf>
- [26] Moohebat, M., Raj, R.G. , Kareem, S.B.A., Thorleuchter, D., "Identifying ISI-indexed articles by their lexical usage: A text analysis approach", *Journal of the Association for Information Science and Technology*, Vol. 66, No. 3, pp. 501–511. doi: 10.1002/asi.23194
- [27] H. Abu-Salem, M. Al-Omari, and M. W. Evens, "Stemming methodologies over individual query words for an Arabic information retrieval system," *Journal of the American Society for Information Science*, Vol. 50: 524-529, 1999
- [28] M. Rosell, "Improving clustering of Swedish newspaper articles using stemming and compound splitting," In *14th Nordic Conference on Computational Linguistics 2003*
- [29] D. Gupta, Y. R. Kumar 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
- [30] T. Korenius, J. Laurikkala, K. Järvelin, M. Juhola, "Stemming and Lemmatization in the Clustering of Finnish Text Documents, *CIKM'04*, 2004
- [31] G. Salton, and S. Buckley, "Term Weighting Approaches in Automatic Text Retrieval," *Information Processing and Management*, Vol. 24: pp. 513-523 , 1988