


# Lexical paraphrasing and pseudo relevance feedback for biomedical document retrieval

Muhammad Wasim<sup>1,2</sup>  · Muhammad Nabeel Asim<sup>2</sup> ·  
Muhammad Usman Ghani<sup>1,2</sup> · Zahoor Ur Rehman<sup>3</sup> ·  
Seungmin Rho<sup>4</sup> · Irfan Mehmood<sup>5</sup>

Received: 15 January 2018 / Revised: 28 March 2018 / Accepted: 24 April 2018  
Published online: 4 June 2018  
© Springer Science+Business Media, LLC, part of Springer Nature 2018

**Abstract** Term mismatch is a serious problem effecting the performance of information retrieval systems. The problem is more severe in biomedical domain where lot of term variations, abbreviations and synonyms exist. We present query paraphrasing and various term selection combination techniques to overcome this problem. To perform paraphrasing, we use noun words to generate synonyms from Metathesaurus. The new synthesized paraphrases are ranked using statistical information derived from the corpus and relevant

---

✉ Irfan Mehmood  
irfanmehmood@ieee.org; irfan@sejong.ac.kr

Muhammad Wasim  
wasim@kics.edu.pk

Muhammad Nabeel Asim  
nabeel.asim@kics.edu.pk

Muhammad Usman Ghani  
usman.ghani@kics.edu.pk

Zahoor Ur Rehman  
zahoor@ciit-attock.edu.pk

Seungmin Rho  
korea.smrho@gmail.com

<sup>1</sup> Department of Computer Science and Engineering, University of Engineering and Technology, Lahore, Pakistan

<sup>2</sup> Al-Khwarizmi Institute of Computer Science, University of Engineering and Technology (UET), Lahore, Pakistan

<sup>3</sup> Department of Computer Science Institute of Information Technology, Attock, Pakistan

<sup>4</sup> Department of Media Software, Sungkyul University, Anyang, Korea

<sup>5</sup> Department of Software, Sejong University, Seoul, Korea

documents are retrieved based on top n selected paraphrases. We compare the results with state-of-the-art pseudo relevance feedback based retrieval techniques. In quest of enhancing the results of pseudo relevance feedback approach, we introduce two term selection combination techniques namely Borda Count and Intersection. Surprisingly, combinational techniques performed worse than single term selection techniques. In pseudo relevance feedback approach best algorithms are IG, Rochio and KLD which are performing 33%, 30% and 20% better than other techniques respectively. However, the performance of paraphrasing technique is 20% better than pseudo relevance feedback approach.

**Keywords** Pseudo relevance feedback · Biomedical document retrieval · Query expansion · Query paraphrasing · Information retrieval

## 1 Introduction

Informational Retrieval is the process of finding documents related to the particular user query from a set of documents also known as corpus. The user feeds a query to the search engine and information retrieval system returns the documents that probably satisfy the user information need. In the field of Biomedical and Bioinformatics, with the discovery of new diseases and their cures, biomedical data is growing rapidly. Therefore, this is attracting the attention of researchers towards the improvement of biomedical data retrieval systems. The performance of biomedical retrieval systems heavily depends upon the vocabulary of user queries and most of the time when user poses a query he just utilize a few keywords as he does not have enough domain knowledge about the concept which he wants to retrieve. At this point, term mismatch problem occurs and user may face retrieval failure.

Term mismatch is a common phenomenon of natural language which occurs when different people name the same concept or thing differently. In information retrieval, it happens when term in user defined query does not match with the terms used in relevant document even if both user defined and document specific terms are referring the same entity. For example, a user puts a query containing keywords *medical practitioner* and corpus is full of relevant documents but all the documents contain the words *doctor*, *physician* etc. It can be seen that all the terms are conveying same information but these are named differently due to which mismatch problem will occur and these documents which are more relevant to the query as compared to others will not be retrieved.

Term mismatch problem can be tackled by reformulation of user query. Query paraphrasing and expansion are widely used techniques for query reformulation. Query expansion can be further categorized into local and global query expansion. In global query expansion, knowledge sources and dictionaries like (WordNet, PubMed) are used to generate candidate expansion terms. In local query expansion, statistical information is used to find candidate expansion terms from corpus. In this approach, documents are retrieved based on user query and top k retrieved documents are considered relevant. To select candidate expansion terms from top retrieved documents, different selection techniques like as Chi-square, Information Gain, Kullback Leibler Divergence (KLD) and Dice are used.

In Query Paraphrasing technique, word mismatch problem is alleviated by generating different *Lexical Paraphrases* of the user query. Lexical paraphrases are those paraphrases which have the same meaning as of query but the key terms of the query are replaced by their synonyms. Synonyms of query words in biomedical domain can be acquired through metathesaurus. Paraphrases are generated by replacing the query words with the found synonyms after applying Parts of Speech (PoS) tagging over the specified query. Once all

paraphrases are generated, these are statistically analyzed using the corpus and a certain score is assigned to each paraphrase. These assigned scores are used to rank the synthesized paraphrases as only top paraphrases are fed to the search engine in order to improve the performance of retrieval system.

In this paper, we introduce a paraphrasing technique for query reformulation in order to improve the performance of biomedical document retrieval. Furthermore, we use different term selection algorithms and introduce two combination based term selection techniques (BordaCount, Intersection) for query expansion in pseudo relevance feedback. Finally, we compare the performance of paraphrasing and pseudo relevance feedback.

The paper is divided into the following sections. Previous work of presented algorithms is described in Section 2. In Section 3, methodology of pseudo relevance feedback approach is discussed along with term selection methods and document ranking models. Section 4 reveal the methodology of query paraphrasing technique. Subsequent Sections 5, 6 discuss the motivational example, experimental setup and results respectively. Last section presents the conclusion and future work.

## 2 Related work

Query expansion techniques have been proposed to overcome word mismatch problem in an information retrieval system. Local Query expansion [15, 27], global query expansion [17, 31], query paraphrasing [41, 42] and Word Sense Disambiguation [20, 21, 25, 38] are most commonly used techniques for query reformulation. In local query expansion, statistical information is used to find candidate expansion terms from corpus. In information retrieval, widely used document retrieval models are Okapi BM25 [37], language models [16], TF-IDF [30]. Similarly in local query expansion most widely used term selection techniques are Chi-Square [3], Kullback Leibler Divergence [28], Lavrenko Relevance Feedback [32], Information Gain [18], Co-occurrence based feedback [36] and Robertson Selection Value. Some work also has been done to improve the performance of pseudo relevance feedback by combining different term selection algorithms such as Jagendra et al proposed a Borda count method to combine different term selection algorithm [35]. However, mentioned work is not directly related to biomedical retrieval.

In 2017, Bouadjenek and Verspoor [7] proposed an effective method for retrieval of biomedical data using query reformulation. The proposed method basically used pseudo relevance feedback methodology after transformation of query for single field to multi field. This multi filed query is then augmented with terms using two state-of-the-art strategies: biomedical lexicon and Rocchio method. To evaluate the integrity of proposed method, they compared their query reformulation method with other baseline methods using bioCADDIE dataset which contains unstructured and structured meta data from a set of 20 individual repositories. Baseline 1 used the description field only for querying the terms and baseline 2 utilized both title and description field. They claimed that multi field query formulation method outperformed state-of-the-art two baselines methods by 3% in terms of Mean Average Precision (MAP).

In global query expansion, knowledge sources and dictionaries like (WordNet, Metathesaurus) are used to generate candidate expansion terms. For the first time in 1997, Stairmand introduced global query expansion method by using WordNet to find synonyms of query terms. Experimentally he could not prove integrity of proposed method and he claimed that in some restricted conditions results can be improved by using global query expansion [39]. In 2008, significantly improved results were presented by Hui Fang [13].

Fang proposed a method of determining the similarity between two terms through WordNet and then utilized this scheme for query expansion [13].

Now-a-days researchers are interested in combining both global and local query expansion approaches in order to take advantage of both strategies. In 2016, Abdulla et al. [1] combined local and global query expansion terms. He used PubMed [29] and MetaMap thesaurus [24] to extract terms relevant to user query. To select terms from locally retrieved documents, they proposed Lavrenko Relevance Feedback (LRF) [19] and Most Frequent Terms (MFT) [40].

Word Sense Disambiguation appends the synonyms of the keywords in the query to expand it and helps to avoid the retrieval of irrelevant information. Recent research using wordsense disambiguation (Schütze and Pedersen, Mitra, Mihalcea, Lytinen) showed the improvement in performance of information retrieval system. However, the practical results presented in (Sanderson [34] and Gonzalo [14]) indicate that there is still a large room for improvement in information retrieval performance.

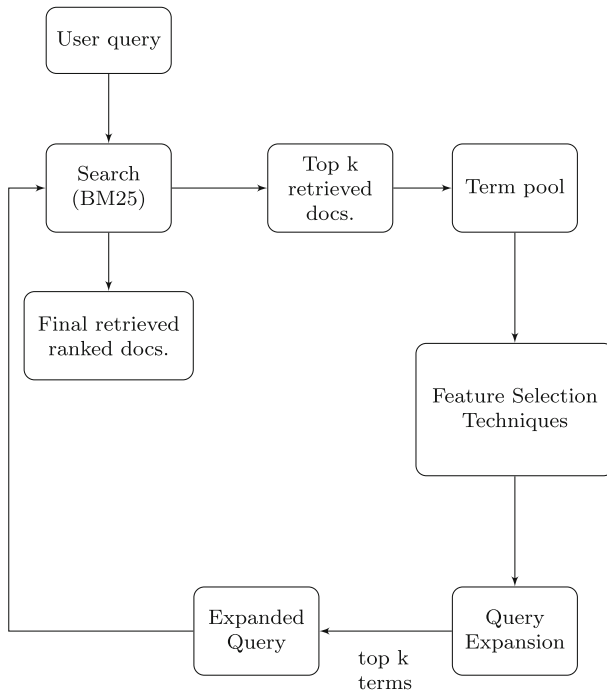
In query paraphrasing, query is reformulated with the words that convey exactly or nearly the same information as query words. The goal of query paraphrasing is to enhance the performance of IR by retrieving documents which are more relevant to the query as described in [4–6]. Recently, many researchers such as Marton et al., [23], Callison-Burch [9] and Bannard [4] contributed to enhance the performance of IR system by suggesting improvement in query paraphrasing. In their work, they generated paraphrases by finding relationship between words in corpus.

This paper discuss query paraphrasing technique and query expansion methods of a full fledged pseudo relevance feedback system in context of biomedical retrieval. In addition, performance of both paradigms are compared and evaluated. In query paraphrasing approach, several semantically similar but lexically different paraphrases are generated by replacing query words with their equivalent synonyms. Only top paraphrases are selected which are found through the ranking of assembled paraphrases, in order to query over search engine. Besides, results of paraphrasing approach is compared against pseudo relevance feedback system which has several query expansion methods. Moreover, two introduced combinational techniques (Borda Count, Intersection) of pseudo relevance feedback system are also taking part in comparison.

### 3 Methodology for pseudo relevance feedback (PRF)

Previous research studies show that a lot of work has been done on document ranking models (such as Okapi BM25, TFIDF, Language models like uni-gram, bi-grams, n-grams) to improve the performance of information retrieval [22]. In IR biomedical domain, Okapi BM25 retrieval model has been extensively used for document retrieval [11]. Due to better accuracy of Okapi BM25 amongst all retrieval models, we used it as document retrieval model in proposed work. Figure 1 depicts the work-flow of query expansion using pseudo relevance feedback. It clearly illustrates that pseudo relevance feedback performance mainly depends upon two key parameters namely document retrieval model and term selection technique.

In the process of pseudo relevance feedback, query provided by the user is sent to the retrieval model. Retrieval model rank documents based on the similarity of words in query and documents. From ranked documents, we make a threshold which splits the corpus into two classes of documents: documents relevant to the user query, documents irrelevant to the user query. Term selection algorithms rank the terms present in relevant class document according to their similarity or co-occurrence with query words. Only the top  $n$  terms



**Fig. 1** Proposed methodology for pseudo relevance feedback

are used to expand the query which is sent back again to retrieval model for document retrieval. We discuss retrieval model and term selection algorithms in subsequent subsections in detail. Furthermore, information about experimental methodology and evaluation of pseudo relevance feedback is discussed in Section 7.

### 3.1 Okapi BM25

The name *Okapi BM25* is combination of 3 different entities: Okapi, BM and 25, in which BM stands for “Best Match”, okapi denotes a system’s name to which this function was implemented for the first time and 25 denotes the 25th trial of the estimation of Poisson function distribution. Okapi BM25 is a probabilistic model that not only assigns weights to documents but also rank them according to their relevance against particular query. It has been widely used in biomedical domain for retrieval of information. Its Mathematical expression is given as the following equation:

$$\begin{aligned}
 BM25(d, q) & \stackrel{rank}{=} \sum_{t \in q} \log \frac{N - n_t + 0.5}{n_t + 0.5} \\
 & \times \frac{(k_1 + 1)n_{d,t}}{k_1 \times ((1 - b) + b \times \frac{|d|N}{\sum_{i \in C} |d_i|}) + n_{d,t}} \times \frac{(k_3 + 1)n_{q,t}}{k_3 + n_{q,t}} \tag{1}
 \end{aligned}$$

Where

- $N$  is the total documents present in the corpus
- $n_t$  denotes the number of documents having term  $t$

- $n_{d,t}$  denotes the number of occurrence of term  $t$  in document  $d$
- $k_1$  and  $k_3$  are the parameters that are used to weight the effect of term  $t$  frequency in document  $d$  and query  $q$  respectively
- $b$  is used as tuning constant to control normalization

Important features of BM25 ranking formula are summarized below:

- The first term in the equation corresponds to inverse document frequency i.e documents having more rare terms will be ranked high. It reduces the effect of query terms that occur in many documents and are unable to discriminate the documents.  $0.5$  is added for smoothing in order to avoid mismatch problems where relevant documents have not a single term matching the query terms.
- The second term controls the weight of the query terms that occur more frequently in document, and also does normalization. The purpose of normalization is to cancel the effect of length of document. Since lengthy documents having greater likelihood have more chances of high ranking against any query even if that long document is not relevant to the query.  $b$  in this term is employed to control the effect of normalization. The value of  $b$  lies in between  $0$  and  $1$ . If it is set to zero, no normalization is performed and if set to  $1$ , full effect of length normalization is utilized.
- The last term is employed for tuning the impact of frequently occurring terms in query. As in longer queries, some terms occur more than once. This repetition could be taken as these terms hold more significance towards the document ranking. Therefore, this factor is added in order to assign high weight to those query terms that occur more than once in frequency. The value of  $k_3$  ranges from  $0$  to a very large value such as  $10,000$ . Zero value suggests only one instance of each query term contributes to the ranking whereas a very large value suggests query terms contribute as often as they occur.

### 3.2 Term selection methods

In domain of query expansion, we regard all the unique terms as features and filtering out such terms is known as *term selection* [2]. However, out of all the terms present in the term pool, only a handful of terms are important and relevant for the purpose of query expansion. Rest of the terms are either irrelevant for this purpose or may not help in retrieving more relevant documents. In this section we will discuss several methods used for term selection in query expansion such as KLD, LRF, PRF etc.

#### 3.2.1 Kullback-Leibler divergence (KLD)

KLD [28], a well known technique in information theory [12], has been continuously used in speech processing and natural language processing. It basically measures the difference between the terms distribution in top retrieved relevant documents  $R$  and the whole document collection  $C$ . Thus, to calculate the KLD score for candidate terms following equation is used:

$$KLD(f) = P_R(f) \times \log \frac{P_R(f)}{P_C(f)} \quad (2)$$

where  $P_R(f)$  is the probability of term or feature  $f$  present in (top retrieved document collection) as shown in the following equation:

$$P_R(f) = \frac{\sum_{D \in R} TF(f/D)}{\sum_{D \in R} \sum_{f \in D} TF(f/D)} \quad (3)$$

$P_C(f)$  is the probability of feature  $f$  present in whole collection of documents  $C$ , given as:

$$P_C(f) = \frac{\sum_{D \in C} TF(f/D)}{\sum_{D \in C} \sum_{f \in D} TF(f/D)} \tag{4}$$

### 3.2.2 Cooccurrence

This is the one of the suitable approach for term selection in query expansion. It investigates the relationship between query terms and corpus of documents. In this approach, we assign score to each candidate term based on its co-occurrence with original query terms provided by the user [36]. We can easily calculate the association of two terms using different co-occurrence coefficients such as:

$$Dice(f_i, f_j) = \frac{2 \times n_{ij}}{n_i + n_j} \tag{5}$$

$$Jaccard(f_i, f_j) = \frac{2 \times n_{ij}}{n_i + n_j - n_{ij}} \tag{6}$$

$n_i$  and  $n_j$  are the number of the documents having  $f_i$  and  $f_j$  terms respectively, whereas  $n_{ij}$  gives the number of documents containing both  $f_i$  and  $f_j$  terms together.

This approach can also be used to find similarity between the candidate terms and the user query terms. But there exists a problem of query drift as it may add couple of terms that are similar to the query terms. In order to tackle this problem, we use a concept named as inverse document frequency(idf) and calculate a co-degree coefficient of candidate terms as shown in the following equation:

$$Co - degree(Q_i, t) = \log_{10}(co(Q_i, t) + 1) \times \left( \frac{idf(t)}{\log_{10}(D)} \right) \tag{7}$$

whereas  $idf(t)$  is defined as:

$$idf(t) = \log_{10} \left( \frac{N}{N_C} \right) \tag{8}$$

where  $N$  shows the number of documents present in corpus,  $D$  is the number of top retrieved documents,  $Q_i$  is the  $i^{th}$  query term,  $t$  is the candidate expansion term, and  $N_C$  shows the number of documents in corpus having term  $t$ . And  $co(Q_i, t)$  shows the number of co-occurrences between  $Q_i$  and  $t$  in the top ranked documents. The above mentioned equation shows how similar is a candidate term to a single query term. Now, we will compute how similar is a candidate term to all query terms as mentioned in the following equation:

$$Co - occurrence_{Final}(Q, t) = \prod_{q_i \in Q} (Co - degree(Q_i, t)) \tag{9}$$

Now, this (9) is used to find the score of candidate terms for query expansion and such type of expansion is known as Co-occurrence based query expansion.

### 3.2.3 Information gain(IG)

It is basically the amount of information gained by knowing the value of the attribute, which is the difference between entropy distribution before and after the split. It is a parameter used to determine the degree of class prediction through the absence or presence of terms in a corpus [18]. Suppose  $A = \{a_1, a_2\}$  be the set of two classes. First, set of documents initially

retrieved after passing the query feed by the user. Second, set of non-relevant documents for the same query. IG score of a term or feature  $f$  can be calculated as:

$$IG(f) = - \sum_{i=1}^{|A|} P(a_i) \log P(a_i) + P(f) \sum_{i=1}^{|A|} P(a_i|f) \log P(a_i|f) + P(\bar{f}) \sum_{i=1}^{|A|} P(a_i|\bar{f}) \log P(a_i|\bar{f}) \tag{10}$$

whereas  $P(f)$  is the probability of feature  $f$  present in documents,  $\bar{f}$  shows the term  $f$  doesn't occur it is calculated as  $(\bar{f} = 1 - P(f))$ ,  $P(f_i)$  is the probability of the  $i^{th}$  value of class,  $P(a_i|f)$  is the conditional probability of the  $i^{th}$  value of class given that  $f$  happens and  $P(a_i|\bar{f})$  is the conditional probability of the  $i^{th}$  value of class given that  $f$  doesn't happen. This value of information gain coefficient obtained is exploited to measure the value of a term with respect to all classes. This kind of query expansion approach is known as IG based query expansion.

### 3.2.4 Probability relevance feedback (PRF)

In this technique, we calculate the probability of a term in relevant documents divided by the probability of term in non-relevant documents [32]. A term with higher probability in relevant class is significant for query expansion. This PRF score of a term is calculated as follows:

$$PRF = \frac{P_{rel}}{P_{non-rel}} \tag{11}$$

where  $P_{rel}$  and  $P_{non-rel}$  are the probability of term in relevant and non-relevant documents respectively.

### 3.2.5 Chi-square probability ( $\chi^2$ )

This is a well known and old chi-square approach as most of the previous work has been done using this probability based approach. Claudio et al. [10] utilized this technique in his work. This techniques is defined as the following equation:

$$\chi^2 = \frac{[P(t_R) - P(t_C)]^2}{P(t_C)} \tag{12}$$

where  $P(t_R)$  and  $P(t_C)$  are the probability of term occurring in relevant documents and in whole corpus respectively.

### 3.2.6 Levrenko relevance feedback (LRF)

This technique uses the formula adopted from Lavrenko Relevance Model [32]. This is basically a language model rather than classical probabilistic ones. The LRF score for the query expansion terms is calculated by using the following formula:

$$score(t) = \sum_{allR} \log \frac{P(t|M_R)}{P(t|G)} \tag{13}$$

In the above equation, there are probabilities.  $P(t|G)$  is the probability of occurrence of the term  $t$  in whole collection. Whereas,  $P(t|M_R)$  can be found using the formula below:

$$P(t|M_R) = \lambda \times \frac{TF(t, R)}{\sum_{t \in R} TF(t, R)} + (1 - \lambda) \times P(t|G) \tag{14}$$



Where  $TF(t, R)$  is the frequency of the term in relevant document R and the denominator is the summation of all the term frequencies for a relevant document. The  $\lambda$  is an adjustable parameter. Researchers have found that  $\lambda$  equals to 0.6 shows best results.

### 3.2.7 Combined term selection techniques

We now present linear combination which has been previously used in biomedical domain and two other less known combination techniques i.e. Borda Count and term intersection.

### 3.2.8 Borda based query expansion

Borda Count is an aggregation method in which list of candidates are ranked in order of preference. In 2015, Sing et al. used this technique to select candidate terms for query expansion [35]. The proposed approach combined the results of given individual feature selection methods using Borda combination technique. Lists of candidate terms are obtained from all the methods. These ranked list are then used to select the final QE terms. For Borda Ranking, a voting mechanism is used which votes all the candidate terms based on their ranks. This voting aggregates the ranked scores of a term from all method and assigns it a collective score. The method is describes using an example below:

Consider ranked lists of the term obtained from individual methods:

- Candidate QE terms obtained from method 1: B, A, C, D
- Candidate QE terms obtained from method 2: A, B, C, D
- Candidate QE terms obtained from method 3: B, A, C
- Candidate QE terms obtained from method 4: B, A, D, C
- Candidate QE terms obtained from method 5: A, C

And now the voting mechanism is applied. For  $n$  number of terms in QE list, the highest scoring term will receive the score of  $n$ , the second highest as  $n - 1$ , and so on. Since there are total 4 terms obtained from the QE techniques, the term present at the highest rank will be given the vote of 4. Term scores from all the methods combined are given as:

- Term Score(A) =  $3 + 4 + 3 + 3 + 4 = 17$
- Term Score(B) =  $4 + 3 + 4 + 4 + 1.5 = 16.5$
- Term Score(C) =  $2 + 2 + 2 + 1 + 3 = 10$
- Term Score(D) =  $1 + 1 + 1 + 2 + 1.5 = 6.5$

Thus the final ranked list will be A, B, C, and D.

### 3.2.9 Intersection combination

In 2008, Perez et al., presented a combination technique technique to select terms named as intersection combination [28]. In this technique firstly, terms are selected by individual term selection algorithms then from lists of selected terms those terms are extracted which occur in all lists.

- First of all, we create a list of candidate terms from all the methods separately.
- The intersection of these list will give the terms that are retrieved commonly by all types of methods.

This way we are able to filter out the noise of all feature selection techniques. As this technique requires a list of words obtained from each of the techniques, so a larger candidate list is obtained from all techniques. The method is describes using an example below:

Let us take a hypothetical dataset and term selection algorithms. Dataset consist of two documents  $D_1$  and  $D_2$  with five distinct terms  $t_1, t_2, \dots, t_5$ . We ranked above mentioned terms using three different algorithms A, B and C.

- Candidate terms ranked by algorithm A:  $t_2, t_3, t_4, t_1, t_5$
- Candidate terms ranked by algorithm B:  $t_2, t_1, t_3, t_5, t_4$
- Candidate terms ranked by algorithm C:  $t_5, t_2, t_3, t_1, t_4$

After the generation of candidate term lists from given methods, top three terms are obtained from each of these ranked lists as given below:

- Candidate terms ranked by algorithm A:  $t_2, t_3, t_4$
- Candidate terms ranked by algorithm B:  $t_2, t_1, t_3$
- Candidate terms ranked by algorithm C:  $t_5, t_2, t_3$

Now, intersection is computed between the terms which means those terms are selected which are common in all lists. In the above mentioned example, we can see the terms  $t_2$  and  $t_3$  are common in these three lists. So, these two terms are finally selected for query expansion.

## 4 Methodology for query paraphrasing

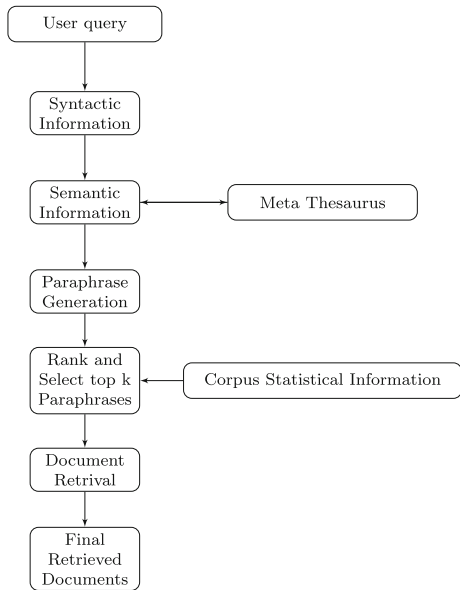
Query paraphrasing is a multi step process. We start with applying POS tagging to user query that yields syntactic information. Syntactic information helps us to identify the functional role of words in a sentence. Syntactic information is further used to find semantic information containing variants and synonyms found against head words of particular user query in the biomedical dictionary (metathesaurus). In order to make sure better retrieval of documents and generation of paraphrases, we keep the query static and iteratively replace only noun or verb present in the user query with the noun or verb synonym that is found in metathesaurus. Newly generated paraphrases get ranked on the basis of statistical information provided by the corpus. Top n paraphrases were selected iteratively and queried using solr search engine for retrieval of final documents (Fig. 2).

### 4.1 Paraphrase construction and ranking procedure

In order to generate lexical paraphrase for a query, three types of information (syntactic, semantic, statistical) about the query sentence are considered. These types are explained as following:

**Syntactic Information** – It is the information about the syntax of the language. Such information explains how words of a language can be combined to form sentences. In other words, it is the grammatical information. This information is needed in order to find the correct synonym for the words. For example, there are two sentences like “Name the country”, and “What is the name of country?” In order to find the correct synonym for the word “name”, it should be known that where does it stand in sentence i.e whether it is being used as a noun or verb. In first sentence, it is being used as verb rather than noun, so a verb synonym should replace it in the first sentence. Similarly we need to

**Fig. 2** Methodology for paraphrasing



find a noun synonym of the same word in order to paraphrase second sentence. Using wordnet some of the possible synonyms for the word “name” as a verb are “identify”, “mention” etc. Similarly for “name” as a noun, synonyms are as “title”, “figure” etc. To find syntactic information from a sentence PoS tagger is used. Multiple tools can be used to perform part of speech tagging. Brill’s PoS tagger (Brill [8]) is one such source that is used in this research.

**Semantic Information** – After identifying syntactic information, next step is to find the semantic information. Semantic information is related to the meaning of the language. In other words, semantic information tells that which concepts of the language are being mapped to a particular word. This information is usually shown through different types of synonyms for that word.

As there can be multiple different types of synonyms related to one word, so in order to avoid unnecessary data explosion, only few are considered. These few semantics are synonyms, attributes, pertainyms, and seealsos (Miller et al. [26]). For example, the synonym of word “low” is “shallow” and its attribute can be “depth” and seealsos can be “short”. Similarly a pertainym for japanese is Japan. In this research Metathesaurus is used to find semantic information of the biomedical terms.

**Statistical Information** – Statistical information of the corpus is also used for paraphrase generation purpose. Corpus is first lemmatised. **Lemma** is the term that is used to define the base concepts of language. For example, lemma for the word “better” will be “good”. Similarly for helping verbs such as “was”, “were”, “is”, “are” etc, the lemma word will be “be”.

Corpus that is being used in this research is first lemmatized. Then for each lemma, its frequency is counted in the whole corpus, and maintained in a lemma dictionary. Similar procedure is performed for each possible lemma pairs. Their frequency is calculated in a five word window. However the lemma pairs that occur only once in the corpus are discarded. This is how the statistical information of the corpus is saved. This information is used in later steps for paraphrase ranking.

In order to generate paraphrases from the query  $q$ , below mentioned steps are followed:

- Query is tokenized, tagged and lemmatized.
- For each lemma, possible synonyms are generated and stop-words are ignored.
- The different combinations of these lemma synonyms then generate multiple possible phrases.

### 4.2 Paraphrase ranking

As described above, for one query  $q$  multiple possible combinations of synonyms can be generated, and hence many paraphrases. However, some of these paraphrases do not correspond with the original query and the information that is present in corpus. Therefore, there must be a mechanism that allow us to consider only those paraphrases that are more close to user query not just syntactically but semantically as well. This ranking is done by checking how common a paraphrase is in the corpus by using probabilistic estimations of lemma pairs.

For the phrase  $t$ , lemma pair  $l(a, b)$  is considered such that lemma  $l_b$  comes after  $l_a$ . The score is calculated based on the fact that how common these lemma pairs are in the corpus. This can be done by determining the probabilities of these lemma pairs. The total score for the paraphrase  $P_t$  can then be calculated as  $P(l_1, l_2, \dots, l_n)$  where  $n$  denotes total number of lemma pairs present in paraphrase  $t$ .

$$P(l_1, l_2, \dots, l_n) = P(l_n|l_{n-1}) \times \dots \times P(l_2|l_1) \times P(l_1) \tag{15}$$

We conclude that by using the above estimation, poor results were obtained. There could be two possible reasons for the generation of poor results:

- While calculating conditional probabilities, we were only considering two lemma  $l_a$  and  $l_b$ , and ignoring other lemmas in query.
- One common lemma is not being counted accurately because of its lemma pair which is approximately infrequent. Suppose,  $l_b$  appears 100 times in corpus and lemma pair  $l_{a,b}$  appears 30 times only, then  $P(l_a|l_b)$  will be 0.3. On the other hand, if another frequent lemma  $l'_a$  appears in corpus 500 times and lemma pair containing  $l'_a|l'_b$  appears 50 times, then  $P(l'_a|l'_b)$  will be 0.10. So the first probability is higher than the second one. The case should have been opposite because the second lemma pair is more frequent in the corpus.

Considering above two reasons, the idea of using conditional probability estimation is discarded. The concept of joint probability estimation is being used in further calculations as it caters the above two problems.

By using joint probability estimate, the probabilities in the above example will be:  $P(l_a, l_b) = 30$  whereas in second case it will be  $P(l'_a, l'_b) = 50$  which is more closer to correct estimate. Now the score for paraphrase  $P$  can be calculated as:

$$P(l_1, l_2, l_3, \dots, l_n) = \prod_{a=1}^n \prod_{b=a+1}^n P(l_a, l_b) \tag{16}$$

This probability can be directly estimated by only considering the frequency of lemma pair as follows:

$$P(l_1, l_2, l_3, \dots, l_n) = \prod_{a=1}^n \prod_{b=a+1}^n frq(l_a, l_b) \tag{17}$$

$$Score(P) = \prod_{a=1}^n \prod_{b=a+1}^n frq(l_a, l_b) \quad (18)$$

Once the scores for all the paraphrases of query  $q$  are obtained, we rank them in decreasing order. And only pick the paraphrases having the highest score.

### 4.3 Key parameters

In this section we discuss some important parameters that can affect the overall score of the paraphrase.

- **Tagging Accuracy:**

Part of Speech tagging is an important factor in this research. If words are not correctly tagged at the initial stage, then this will have negative impact on the obtained synonyms and eventually on the whole paraphrase generation and ranking process. Therefore, highly accurate tagging is needed. Using Brill's tagger [8], an overall accuracy of 84% was obtained. In order to make sure the tagging is 100% accurate, we manually remove all the wrongly tagged part of speech.

- **Weight of Word Order ( $W_{order}$ ):**

Word order is the parameter that is used to consider the relative lemma pair order while calculating the frequencies. For example, if  $(l_a, l_b)$  is a lemma pair, then whether its frequency should be calculated in this order only or also in the reverse order too i.e.,  $(l_b, l_a)$ . To determine this, the following equation is formed:

$$frq(l_a, l_b) = frq(l_a - > l_b) + W_{order} \times frq(l_b - > l_a) \quad (19)$$

where,

$frq(l_a - > l_b)$  = expressing the order  $((l_a, l_b))$

$frq(l_b - > l_a)$  = expressing the order  $((l_b, l_a))$

If  $W_{order} = 0$ , then reverse order is ignored

If  $W_{order} = 1$ , then reverse order is counted

- **Absent Pairs:**

During calculations, it is quite possible that a lemma-pair, under consideration, is not in the list. As a result, zero frequency will be returned and it will make the whole term for a paraphrase equal to zero. To account its effect, a new factor i.e., Absent Frequency ( $AbFq$ ) is introduced.  $AbFq = 0.1$  for a lemma-pair not present in the dictionary. It means that the score will be divided by 10 for such cases. Similarly, there is another factor, Absolute Adjacent Divisor,  $AbAjFq$ . This factor takes into account the effect of adjacent lemma pairs on this algorithm. It is actually the divisor of  $AbFq$  and determines the factor how much to divide  $AbFq$  for a lemma-pair that is not present. For example,  $AbAjFq = 10$  will have the effect of  $AbFq \div AbAjFq = 0.01$  on the score due to adjacent absent lemma pair.

- **Words Co-Location**

The research was performed under two different settings, namely **CoLoc** setting, and **NoLoc** setting. In first setting we consider co-locations of the lemma pairs under consideration. Consider the word "folic acid". If we are considering the CoLoc setting, then we'll consider both of them as one word and look for synonyms considering it as one word. CoLoc setting will return synonyms such as "vitamin M", "vitamin Bc"

and such. Whereas in NoLoc setting, both the words will be considered as separate individual words, yielding synonyms such as “folic lsd” and etc.

The effect of these two settings was observed on the document retrieval process. It was noticed that number of documents retrieved using NoLoc setting were higher than the ones obtained in CoLoc setting. One possible theory behind such a behavior is the fact that in NoLoc setting, the frequency of the lemma or lemma pair that is important for retrieval process is preserved in paraphrases. The keyword “folic” will stay “folic” in all paraphrases. This impacts retrieves more such documents as compared to the other setting in which paraphrase contains synonyms such “vitamin M” and “vitamin Bc”. Because of this fact, number of documents retrieved in NoLoc setting is slightly higher than CoLoc one.

- **Query Length**

Different query lengths were experimented. Using the optimal settings of the above mentioned parameters, it was found that shorter queries ranging from 6 to 11 words retrieved more number of documents

#### 4.4 Document retrieval

Once possible paraphrases of query are generated, ranked, and selected, the next step is to use those paraphrases in combination with original query to retrieve possible documents. Lets say we originally chose 19 paraphrases, including query q, we have total  $T$  sentence  $(P_0, P_1, P_2, \dots, P_T)$  where  $P_0$  represents original query q.

Following are the steps taken to score and retrieve documents.

- Extract content lemma for each paraphrase. Let  $n$  be the total lemmas present in paraphrase  $P_i$ .
- For each lemma from paraphrase  $P_a$ , its Term Frequency Inverse Document Frequency *tfidf* (Salton and McGill [33]) score is calculated for each retrieved document. Let  $tfidf(D_k, l_{a,b})$  be the score of document  $D_k$  retrieved for lemma  $l_{a,b}$ . If document  $D_k$  is retrieved by more than one lemma of paraphrase  $P_a$ , then its *tfidf* scores are added and we get:

$$\sum_{b=1}^n tfidf(D_k, l_{a,b}) \tag{20}$$

To get the total score of Document  $D_k$  for paraphrase  $P_a$ , the equation 4 will be multiplied with equation 6:

$$DocScr(D_k, P_a) = Scr(P_a) \times \sum_{b=1}^n tfidf(D_k, l_{a,b}) \tag{21}$$

- Now we need to consider the effect of all other paraphrases on document  $D_k$ . We use all other paraphrases in the same way and calculate total score of document  $D_k$  by adding the individual scores of paraphrases. Let there be total  $T$  paraphrases (including query q), then mathematically score of a document can be calculated as

$$DocScr(D_k) = \sum_{a=1}^T \{Scr(P_a) \times \sum_{b=1}^n tfidf(D_k, l_{a,b})\} \tag{22}$$

## 5 Motivational example

To illustrate the proposed strategy of paraphrasing, we take a hypothetical example. In this particular example, we consider a corpus of six documents available at<sup>1</sup> containing information about biological disease and its cure. We take a query “*HIV and the GI tract, recent reviews?*” from OHSUMED dataset.

Now we consider the following steps sequentially in order to generate query paraphrases:

1. In the very first step, tokenization and POS tagging are applied over the query as given below:

HIV < NN > and < CC > the < DT > GI < NNP > tract < NN >, recent < JJ > reviews < NNS >

Where,

- NN: stands for Noun
- NNP: stands for Proper noun(singular)
- NNS: stands for Noun(plural)
- CC: stands for Conjunction
- DT: stands for Determiner
- JJ: stands for Adjective

After POS tagging, lematization and removal of all stop words are performed to get the following terms:

hiv, tract, gi, recent, review.

2. In the second step, all possible synonyms of these terms are obtained using metathesaurus. The synonyms for the term HIV, tract, GI, recent and reviews are as follow:

- For HIV[noun], we have:
  - 1: Immunological disorder
  - 2: immune deficiency syndrome
  - 3: Sexually transmitted example
  - 4: Sexually transmitted infection
- For GI tract, we have:
  - 7: digestive tract
  - 8: alimentary canal
  - 9: alimentary tract
  - 10: digestive tube
- For recent[adj], variants are:
  - 11: recent
  - 12: recency
  - 13: current

<sup>1</sup><https://github.com/wasimbhalli/DSL-BiomedicalQueryExpansion/blob/master/resources/corpus-motivational-example>

- For review[noun], we have:

14: reviews  
 15: assess  
 16: analysis

3. After synonym generation, we make paraphrases of given query. In paraphrase generation, at a time in query we replace one word with synonym and take all other static. For the sake of simplicity of example, we take the following synonyms:

- HIV: Sexually transmitted infection
- GI tract: alimentary tract
- GI tract: digestive tract
- recent: recency

From these above mentioned synonyms, following phrases are generated:

- (a) sexually transmitted infection gi tract recent reviews?
- (b) hiv alimentary tract recency reviews?
- (c) sexually transmitted infection digestive tract recent reviews?
- (d) hiv gi tract recent reviews?
- (e) hiv alimentary tract recent reviews?
- (f) sexually transmitted infection gi tract recency reviews?
- (g) sexually transmitted infection alimentary tract recent reviews?
- (h) hiv digestive tract recent reviews?
- (i) hiv digestive tract recency reviews?
- (j) sexually transmitted infection digestive tract recency reviews?
- (k) sexually transmitted infection alimentary tract recency reviews?
- (l) hiv gi tract recency reviews?

Generally, number of paraphrases are usually calculated by exploiting the following formula:

$$\text{Number of Paraphrases} = 2^n$$

where,

$$n = \text{Number of Synonyms}$$

If we consider the above mentioned formula, number of generated paraphrases are supposed to be sixteen as we have used four synonyms but as we have used the two synonyms of single term “gi tract”, one for the term hiv and one for the term recent, number of generated paraphrases are twelve.

4. After the generation of query paraphrases, these phrases are ranked in descending order. This ranking is done because all the generated phrases are not perfect. In order to select only those paraphrases that corresponds to original query, we find statistical information from the corpus. For this information collection, we make pairs of terms in query phrases and find their frequency in the corpus by using seven word window. For example, for the query phrase: *hiv gi tract recent reviews?*, (hiv, gi), (hiv, tract), (hiv, recent),



(hiv, reviews), (gi, tract), (gi, recent), (gi, reviews), (tract, recent), (tract, reviews) and (recent, reviews) pairs are generated. Frequency of each pairs is found from the corpus as:

$$\begin{aligned}
 frq(hiv, gi) &= 0 \\
 frq(hiv, tract) &= 0 \\
 frq(hiv, recent) &= 0 \\
 frq(hiv, reviews) &= 0 \\
 frq(gi, tract) &= 3 \\
 frq(gi, recent) &= 0 \\
 frq(gi, reviews) &= 0 \\
 frq(tract, recent) &= 0 \\
 frq(tract, reviews) &= 0 \\
 frq(recent, reviews) &= 0
 \end{aligned}$$

$$Score(P) = \prod_{a=1}^n \prod_{b=a+1}^n frq(l_a, l_b) \quad (23)$$

By putting the frequency count of each pair in above mentioned equation, we find score for query paraphrase. For given query paraphrase, zero score will be obtained.

$$Score(P) = 0 \times 0 \times 0 \times 0 \times 3 \times 0 \times 0 \times 0 \times 0 \quad (24)$$

$$Score(P) = 0 \quad (25)$$

To overcome this problem we assign an absent frequency score 0.1 to the non-adjacent pairs such as (tract, reviews) and for adjacent lemma pairs such as (tract, recent) we divide 0.1 by some adjacent factor as in our case say 10 and then use this 0.01 value for adjacent pairs. Now, the score will be calculated as:

$$Score(P) = 0.01 \times 0.1 \times 0.1 \times 0.1 \times 3 \times 0.1 \times 0.1 \times 0.01 \times 0.1 \times 0.1 \quad (26)$$

$$Score(P) = 3 \times 10^{-12} \quad (27)$$

Similarly, we calculate the score for each query paraphrase. Final score of each paraphrase along with their rank is shown in Table 1.

Table 1 rank the paraphrases according to their scores like the paraphrase “g” has assigned the highest rank (1st).

- From these ranked paraphrases, we select top 10 paraphrases for document retrieval.
- In final step, query and selected paraphrases are used to retrieve and rank the documents. To keep the example simple, we briefly describe the process of document retrieval for paraphrase (d). We calculate tf-idf score of all documents retrieved against each term present in paraphrase (d). In this paraphrase: *hiv gi tract recent reviews?*, we have 5 terms. We give these five terms one by one to tf-idf document retrieval model. Scores of all retrieved documents against each term are given below:

- For hiv, retrieved documents with tf-idf scores are:

$$\text{Document} = \text{Doc1}, \text{Score} = 0.2488$$

**Table 1** Scores of query paraphrases and their ranking

Paraphrase	Score ( $10^{-12}$ )	Rank	Query Paraphrases
a	0.432	7th	sexually transmitted infection gi tract recent reviews?
b	2	4th	hiv alimentary tract recency reviews?
c	0.00072	11th	sexually transmitted,infection digestive tract recent reviews?
d	3	3rd	hiv gi tract recent reviews?
e	2	5th	hiv alimentary tract recent reviews?
f	0.433	6th	sexually transmitted infection gi tract recency reviews?
g	4.32	1st	sexually transmitted infection alimentary tract recent reviews?
h	0.01	8th	hiv digestive tract recent reviews?
i	0.01	9th	hiv digestive tract recency reviews?
j	0.00216	10th	sexually transmitted infection digestive tract recency reviews?
k	4.3	2nd	sexually transmitted infection alimentary tract recency reviews?
l	3	3rd	hiv gi tract recency reviews?

- (b) For gi, retrieved documents with tf-idf scores are:

$$\text{Document} = \text{Doc1}, \text{Score} = 0.3048$$

- (c) For tract, retrieved documents with tf-idf scores are:

$$\text{Document} = \text{Doc1}, \text{Doc2}, \text{Doc4}, \text{Score} = 0.21104, 0.1378, 0.1218$$

- (d) For the term recent, no documents are retrieved by tf-idf:

- (e) For the term “reviews”, no documents are retrieved by tf-idf:

We then add the scores of all retrieved documents and multiply the yielded sum of each document with the paraphrasing score of (d) mentioned in Table 1

For instance:

$$\text{Sum of Doc1 (sd1)} = 0.76464$$

$$\text{Sum of Doc2 (sd2)} = 0.1378$$

$$\text{Sum of Doc3 (sd3)} = 0$$

$$\text{Sum of Doc4 (sd4)} = 0.1218$$

$$\text{Sum of Doc5 (sd5)} = 0$$

$$\text{Sum of Doc6 (sd6)} = 0$$

$$\text{Paraphrasing score of (d)} = 3 \times 10^{-12}$$

$$\text{Score of Doc1} = (\text{sd1}) \times \text{Paraphrasing score of (d)}$$

$$\text{Score of Doc2} = (\text{sd2}) \times \text{Paraphrasing score of (d)}$$

$$\text{Score of Doc3} = (\text{sd3}) \times \text{Paraphrasing score of (d)}$$

$$\text{Score of Doc4} = (\text{sd4}) \times \text{Paraphrasing score of (d)}$$

$$\text{Score of Doc5} = (\text{sd5}) \times \text{Paraphrasing score of (d)}$$

$$\text{Score of Doc6} = (\text{sd6}) \times \text{Paraphrasing score of (d)}$$

Similarly, we repeat the above mentioned document retrieval process for remaining nine paraphrases and ultimately we add the scores of each document in quest of final sum

to rank the retrieved documents. After querying through nine paraphrases, we get the final scores of each document listed after ranking:

Score of Doc2= 12.0509  
Score of Doc1= 9.8901  
Score of Doc4= 3.1398  
Score of Doc3= 1.5801  
Score of Doc5= 0  
Score of Doc6= 0

### **Pseudo Relevance Feedback:**

Contrarily, to demonstrate the work flow of pseudo relevance feedback, we consider the following parameters:

Query: “HIV and the GI tract, recent reviews?”  
Term Selection Method: KLD

Now we take into account following steps sequentially:

- (a) In the first step, tokenization, POS-tagging and removal of stop words are performed to get the following query words:
  - hiv
  - tract
  - gi
  - recent
  - review
- (b) In the second step, we feed yielded query words in a document ranking model named “Okapi BM25” and it retrieve four documents with the following rank:
  - Doc1
  - Doc2
  - Doc4
- (c) In the third step, we select only top three documents from the set of retrieved documents and rank the terms by assigning a particular score through a term selection technique (e.g KLD) to all the terms of selected documents.
- (d) In the fourth step, we expand the specified query by selecting these top five ranked terms (vaccines , clinical, gi, hiv, pathogenic)
- (e) Finally, we submit the expanded and new formulated query again to Okapi BM25 to retrieve following final ranked documents:
  - Doc1
  - Doc3
  - Doc2
  - Doc4

## **5.1 Discussion**

In above discussed motivational example, expected result of the specified query “HIV and Gi tract recent reviews” is the information regarding recent advancement and study about the role of gastrointestinal (GI) in pathogenesis of sexually transmitted infection and antiretroviral therapies, vaccines which directly effect the immunity of the digestive tract.

In mentioned corpus, doc2 has the content which is most closest to expected results followed by doc1, doc4 and doc3 in terms of relevancy whereas doc5 and doc6 are totally irrelevant. For query paraphrasing, all documents are correctly ranked by exploiting proposed methodology whereas pseudo relevance feedback has just yielded these three documents (doc1,doc2,doc4) as relevant to the query. Pseudo relevance feedback approach has missed to retrieve doc3 due to word mismatch problem as pseudo relevance feedback only consider the occurrence of query words within corpus documents. Moreover, even after employing query expansion, we get all four documents however there rank is not correct at all (doc1,doc3,doc2,doc4).

## 6 Experimental setup and results

In this section, we describe the experimental setup used by pseudo relevance feedback and query paraphrasing system to improve biomedical retrieval

An open source standalone enterprise search platform knowns as “Solr” (<http://lucene.apache.org/solr/>) is used for experimentation. It has features of full text search, open interfaces of several standards (i.e Json, Xml, Http), smart filtering and real time indexing as well. To parse the text of dataset in addition to solr, we use java code available at github repository.<sup>2</sup> There exist many document retrieval models but we employ most widely used Okapi BM25, a probabilistic weighting model with following default parameters  $k_1$  (1.2d),  $k_3$  (8d) and  $b$  (0.75d). We implement several term selection algorithms in the underline architecture of solr for the discovery of relevant terms of specified query. In next sections, we briefly describe dataset and evaluation measure used for experimentation. Furthermore we discuss the results of paraphrasing and pseudo relevance feedback in different sections. Lastly, we compare the performance of paraphrasing and pseudo relevance feedback.

### 6.1 Dataset and evaluation measure

This section describe the dataset and evaluation measure employed for the assessment of different algorithms(i.e Term selection, Document retrieval, Paraphrasing technique).

We use OHSUMED ([http://trec.nist.gov/data/t9\\_filtering/readme](http://trec.nist.gov/data/t9_filtering/readme)) dataset which is a sub collection of the MEDLINE (<https://www.nlm.nih.gov/pubs/factsheets/medline.html>) database. It consists of 348,566 documents containing the fields of abstract, author, publication type, MESH indexing and source. OHSUMED dataset has 62 queries and 39806482 terms.

To evaluate the integrity of presented algorithms, “Mean Average Precision” (MAP) is utilized. There exist some other evaluation measures such as recall and F1 measure but these measures are mostly used for classification applications to evaluate the integrity of classifiers. In information retrieval, almost all the competition tracks and researchers use MAP to evaluate retrieval models. Mathematical equation of average and mean average are discussed below:

**Average precision** This measure compares the documents ranked by retrieval model with predefined set of documents ranked by domain experts against particular query.

$$\text{Average } P = \frac{\sum_{r=1}^D (P(r) \times \text{rel}(r))}{C_{rt}} \quad (28)$$

<sup>2</sup><https://github.com/darthcodus/Lucene-TREC-OHSUMED>

where

- $r$  is a rank
- $D$  denotes the number of retrieved documents
- $\text{rel}(r)$  is a function that tells whether a document is relevant or not (binary)
- $P(r)$  stands for precision

**Mean average precision** It summarizes the ranking results from multiple queries by averaging the *AverageP*.

$$MAP = \frac{\sum_{q=1}^Q \text{AverageP}(q)}{|Q|} \quad (29)$$

We used python script to evaluate the performance of information retrieval provided by Aaron Cohen used in trecgenomic track.

## 6.2 Summary of query paraphrasing system performance

We generated paraphrases for each query and then ranked them based on statistical information. Top  $k$  paraphrases are selected iteratively for each query such that  $k$  varies with the difference of 5, starting from 10 and going upto 30. These top paraphrases were queried using Solr search engine to retrieve final documents. For sake of efficient analysis, we calculated MAP (Mean Average Precision) using available python script released in 2007 TREC competition. Table 2 clearly summarize the performance of proposed paraphrasing system on custom defined benchmark.

We concluded that performance of the system just kept increasing slightly till the third test point (20) and after that it kept conversing the trend by getting decreased slightly. This happened because we rank the paraphrases according to the statistical information of corpus, hence only top paraphrases are more relevant to user query. Table 2 clearly indicates that paraphrasing technique start causing the problem of query drift after 20 number of paraphrases due to high extent of irrelevancy of generated paraphrases with the query. This leads to decrease in performance of paraphrasing approach after the selection of 20 paraphrases. Moreover, we use only “OHSUMED” dataset for experimentation as it contain only abstracts instead of extensive text and making it easy to find statistical information from corpus. Besides, other available benchmark dataset for information retrieval in biomedical domain such as “MEDLUNE” is pretty large. This makes it almost impossible for all researchers and practitioners to find statistical information from corpus.

## 6.3 Performance of pseudo relevance feedback system

This section summarizes the performance of seven term selection algorithms(Chi-Square, KLD, RSV, CoDice, IG, LRF, PRF,ROCCHIO) along with introduced combination based term selection algorithms, used to rank the terms for the task of query expansion. As expected, three out of seven term selection algorithms (KLD, IG, Rocchio) performed way better than the rest. In quest of improving the results of pseudo relevance feedback even

**Table 2** Mean average precision of paraphrases for different values of  $k$

Top Paraphrases	10	15	20	25	30
MAP	0.29018	0.2982	0.2991	0.2931	0.2702

furthermore, we use combination based term selection techniques (Borda Count, Intersection) on three mentioned best performing metrics.

The performance of IR system using pseudo relevance feedback system depends upon two factors: number of top retrieved relevant documents and number of candidate expansion terms. Table 3 summarize the performance of seven term selection algorithms and introduced combination based term selection techniques over defined set of documents(10,15,20,25,30) and terms(5,10,15,20,25,30). All results are generated by keeping the number of documents (e.g Number of documents = 10) static and varying the number of terms (5,10,15, 20, 25, 30). Considering only the following pairs of test points (Number of documents = 10)(Number of terms = 5,10,15,20,25,30) in the table, it can be summarize that information gain (IG) has outclassed the rest by marking the best performance at certain pair of test points (No of documents=10, No of terms=25) whereas, Rocchio marks the worst performance at following pair of test points (No of documents=10, No of terms=30). Likewise, on next pairs of test points (Number of documents = 15)(Number of terms = 5,10,15,20,25,30), KLD has performed way better then the rest at this pair of test point (No of documents=15, No of terms=15) and PRF was worst amongst all at this pair of test point (No of documents=15, No of terms=30) . However, with the increase in number of documents, Rocchio's performance is also increased as Rocchio overshadow the rest three times at these pairs of test points (No of documents=20,25,30 No of terms= 15,20,30) and PRF prove worst amongst all over mentioned pairs of test points. In case of overall performance, IG and Rocchio surpass the rest two times each at following pairs of test points (No of documents=10 , 30), (No of documents= 20, 25) respectively. In addition, RSV has shown great performance at following test point (No of documents= 15). On the other hand, with the increase in number of documents, combination based term selection technique Intersection's performance kept decreasing and BordaCount has shown a mix behavior. However, in overall comparison, both combination based term selection techniques did not outperform any single term selection algorithm at all. Bold values in Table 3 represents the highest value obtained by a term selection technique for specific document test point against all term test points.

Table 4 summarize the performance difference of nine term selection algorithms by varying number of terms (5,10,15,20,25,30) against static number of documents. We use standard deviation to assess the performance fluctuation. Each cell of the table depicts standard deviation, computed at specific number of documents (i.e No of documents = 10) by iterating over the number of terms defined in a following set (5,10,15,20,25,30). For instance, ChiSquare mark the deviation of 0.0125, when number of terms (5,10,15,20,25,30) are varied against static number of documents (No of documents = 10). As the table suggests, ChiSquare and RSV kept raising their performance and stability levels with the increase in number of documents against the defined set of terms, except on a single test point (No of documents = 15) where they mark a slight increase in deviation value. On the other hand, all remaining term selection algorithms and introduced combinational techniques have shown a mix behavior regarding performance fluctuation and stability over defined set of documents (10,15,20,25,30).

Table 5 reveals the performance of nine term selection algorithms in terms of percentages. Each cell of the table indicates a sum computed by counting the total number of times a particular algorithm overshadow the rest against static number of documents (i.e No of documents=10) and varied number of terms (5,10,15,20,25,30). For instance, first tuple highlights that IG outperformed all other algorithms thrice whereas RSV and KLD surpass the performance of other algorithms only once, when number of terms are iteratively varied from the defined set (5,10,15,20,25,30) against particular number of documents (No

**Table 3** Mean average precision (MAP) of nine term selection algorithms used for pseudo relevance based query expansion, where  $F_{BD}$  represents feedback documents and  $F_{BT}$  shows selected candidate terms

$F_{BD}$	$F_{BT}$	Term Selection Algorithms								
		ChiSquare	KLD	RSV	CoDice	IG	PRF	Rocchio	Borda count	Intersection
10	5	0.2472	0.2792	0.2797	0.2560	0.2814	0.2292	<b>0.1982</b>	0.2656	0.2395
	10	0.2541	<b>0.2849</b>	0.2840	0.2564	0.2823	0.2260	0.1793	0.2799	0.2393
	15	0.2658	0.2832	0.2826	0.2680	0.2820	0.2289	0.1716	<b>0.2832</b>	<b>0.2432</b>
	20	0.2700	0.2843	<b>0.2857</b>	<b>0.2681</b>	0.2840	0.2282	0.1564	0.2828	0.2388
	25	0.2741	0.2830	0.2818	0.2631	<b>0.2864</b>	<b>0.2398</b>	0.1480	0.2810	0.2311
	30	<b>0.2804</b>	0.2820	0.2819	0.2628	0.2857	0.2394	0.1426	0.2826	0.2294
15	5	0.2361	0.2799	0.2811	0.2579	0.2812	0.2286	0.2683	0.2644	<b>0.2395</b>
	10	0.2595	0.2923	0.2895	0.2701	<b>0.2968</b>	0.2296	0.2932	0.2853	0.2384
	15	0.2680	<b>0.2982</b>	<b>0.2979</b>	0.2703	0.2923	0.2223	<b>0.2964</b>	<b>0.2921</b>	0.2370
	20	0.2700	0.2946	0.2945	<b>0.2745</b>	0.2942	0.2242	0.2929	0.2909	0.2362
	25	<b>0.2796</b>	0.2910	0.2923	0.2742	0.2916	0.2285	0.2936	0.2862	0.2355
	30	0.2788	0.2884	0.2884	0.2684	0.2870	<b>0.2343</b>	0.2907	0.2861	0.2302
20	5	0.2420	0.2801	0.2809	0.2477	0.2846	0.2239	0.2721	0.2616	<b>0.2395</b>
	10	0.2455	<b>0.2922</b>	0.2911	0.2606	0.2902	0.2233	0.2902	0.2866	0.2384
	15	0.2565	0.2921	<b>0.2921</b>	<b>0.2638</b>	<b>0.2926</b>	0.2208	<b>0.2976</b>	<b>0.2914</b>	0.2390
	20	0.2583	0.2888	0.2895	0.2580	0.2885	0.2210	0.2945	0.2878	0.2385
	25	0.2668	0.2860	0.2863	0.2595	0.2837	0.2171	0.2884	0.2886	0.2329
	30	<b>0.2727</b>	0.2813	0.2806	0.2579	0.2806	<b>0.2248</b>	0.2852	0.2863	0.2282
25	5	0.2429	0.2791	0.2789	0.2445	0.2756	<b>0.2232</b>	0.2731	0.2657	<b>0.2394</b>
	10	0.2419	<b>0.2936</b>	<b>0.2906</b>	0.2444	0.2857	0.2229	0.2905	0.2780	0.2388

**Table 3** (continued)

<i>F<sub>B<sub>D</sub></sub></i>	<i>F<sub>B<sub>T</sub></sub></i>	Term Selection Algorithms									
		ChiSquare	KLD	RSV	CoDice	IG	PRF	Rocchio	Borda count	Intersection	
15	15	0.2445	0.2848	0.2850	<b>0.2488</b>	<b>0.2913</b>	0.2209	0.2912	0.2837	0.2380	
	20	0.2542	0.2863	0.2857	0.2473	0.2906	0.2204	<b>0.2955</b>	<b>0.2876</b>	0.2400	
25	25	<b>0.2647</b>	0.2857	0.2841	0.2479	0.2874	0.2174	0.2893	0.2831	0.2414	
	30	0.2645	0.2796	0.2810	0.2452	0.2817	0.2216	0.2868	0.2825	0.2369 30	
30	5	0.2414	0.2759	0.2760	0.2407	0.2772	0.2200	0.2653	0.2677	<b>0.2394</b>	
	10	0.2481	<b>0.2885</b>	<b>0.2865</b>	<b>0.2424</b>	0.2827	<b>0.2220</b>	0.2843	0.2734	0.2388	
15	15	0.2460	0.2831	0.2832	0.2379	0.2890	0.2191	<b>0.2947</b>	0.2821	0.2327	
	20	0.2524	0.2775	0.2765	0.2329	0.2894	0.2192	0.2893	<b>0.2848</b>	0.2360	
25	25	0.2567	0.2811	0.2811	0.2345	0.2862	0.2154	0.2877	0.2843	0.2350	
	30	<b>0.2579</b>	0.2796	0.2786	0.2341	<b>0.2907</b>	0.2140	0.2840	0.2809	0.2359	



**Table 4** Performance fluctuation in terms of standard deviation at variant number of expanded terms for nine term selection algorithms

Standard Deviation of Term Selection Algorithms									
FBD	ChiSquare	KLD	RSV	CoDice	IG	PRF	Rocchio	Borda Count	Intersection
10	0.0125	0.0020	0.0021	0.0053	0.00201	0.0061	0.0210	0.0068	0.0054
15	0.0161	0.0063	0.00578	0.0061	0.0056	0.0042	0.0104	0.0101	0.0033
20	0.0119	0.0052	0.0050	0.0055	0.0045	0.0028	0.0089	0.0110	0.0046
25	0.0106	0.0053	0.0041	0.0019	0.0059	0.0021	0.0077	0.0077	0.0016
30	0.0064	0.0045	0.0041	0.0039	0.0051	0.0030	0.0101	0.0068	0.0025

of documents=10). Performance percentages are computed using following mathematical expression:

$$Performance\ in\ Percentage = \frac{|Winning\ Sum|}{Total\ Number\ of\ Test\ Points} \times (100) \quad (30)$$

whereas,

- Total number of test points =  $|D| \times |T|$
- $|D|$  = Total number of document instances = 5
- $|T|$  = Total number of term instances = 6

As the table suggests, it’s pretty easy to understand that IG is a clear winner as it performs the best with the highest percentage of 33.33% amongst all in overall comparison of the performance. Likewise, Rocchio is the second best performer marking the percentage of 20.00% whereas, ChiSquare, CoDice, PRF all fail terribly to compete in overall performance comparison by revealing the lowest percentage of 00.00% except KLD who marks the reasonable performance of 20%. Similarly, combinational techniques (Intersection, BordaCount) did not perform up to the mark even though BordaCount’s performance is little bit better then the worst performing Intersection by the figure of 10.00%. On the other hand, performance of paraphrasing technique can be computed after analyzing comparison graph Fig. 3. As the graph refers, paraphrasing technique marks it’s peak performance over three different test points (No of documents= 15, 20, 25). Hence, performance of paraphrasing technique in percentage can be calculated using following equation:

$$Performance\ in\ Percentage = \frac{|Number\ of\ Peak\ Points|}{Total\ Number\ of\ Test\ Points} \times (100) \quad (31)$$

whereas,

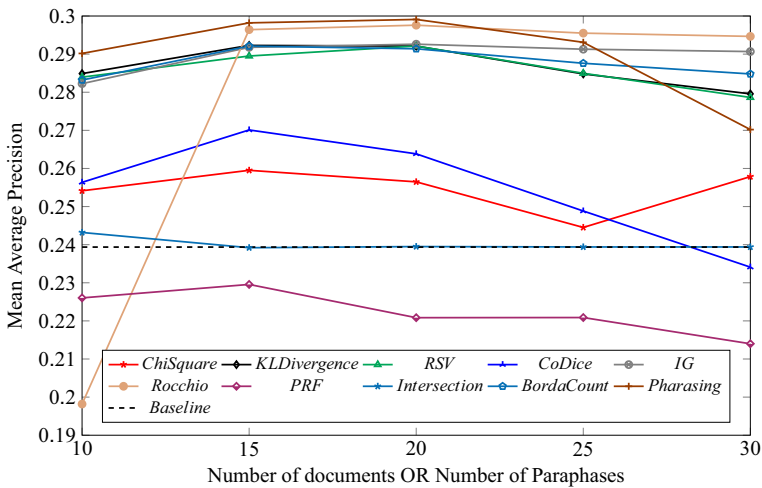
- Number of Peak Points = 3
- Total number of test points =  $|D|$
- $|D|$  = Total number of document instances = 5

Performance of Paraphrasing Technique = 60% Performance of Pseudo Relevance Feedback System in Percentage = 40%

Considering above mentioned performance figures, we can say that paraphrasing approach performs 20% better then pseudo relevance feedback system.

**Table 5** Percentage of nine term selection algorithms

Performance of Term Selection Algorithms in Percentage									
FBD	ChiSquare	KLD	RSV	CoDice	IG	PRF	Rocchio	Borda Count	Intersection
10	0	1	1	0	3	0	0	1	0
15	0	1	1	0	2	0	2	0	0
20	0	1	0	0	1	0	2	2	0
25	0	2	0	0	1	0	3	0	0
30	0	1	0	0	3	0	2	0	0
Percentage	0.00%	20.00%	6.67%	0.00%	33.33%	0.00%	30.00%	10.00%	0.00%



**Fig. 3** peak results of feature selection techniques with combinational methods and paraphrasing

### 6.4 Comparison

In this section, graphically we compare the results of pseudo relevance feedback and paraphrasing. All feature selection techniques do not produce their peak results at the same defined set of parameters which are number of top retrieved relevant documents (10,15,20,25,30) and candidate expansion terms (5,10,15,20,25,30) that get merge with query.

That is why, to lay out the clear picture of the performance of PRF system and better comparison, Fig. 3 shows peak results only against the best parameters for all techniques found through exhaustive testing as highlighted in the Table 3. For instance, taking into account following pair of test points (Number of documents = 10)(Number of terms = 5,10,15,20,25,30) ChiSquare produces it’s best result at (Number of documents = 10, Number of terms = 30), KLD (Number of documents = 10, Number of terms = 10), whereas RSV at (Number of documents = 10, Number of terms = 20) and so on. Hence, we consider only these points as a benchmark for test in graphical representation. In addition, results of paraphrasing technique over following test points (Number of documents = 10,15,20,25,30) have also been added in the graph.

Mentioned pictorial representation summarize the performance of both PRF against the number of documents and Paraphrasing system against the number of paraphrases in terms of mean average precision. As the graph suggests, it’s pretty easy to understand that Rocchio and paraphrasing technique outperformed the rest but in a straight comparison, Rocchio is a clear winner because it went up gradually despite under performing at start (No of doc=10) to meet the highest starting level set it by Paraphrasing system. On the other hand, even after the good start by getting flattened out and keeping steady at some test points, paraphrasing performance start getting decrease after a dramatic dropped of at certain test point (No of doc=20). Moreover, Probability Relevance Feedback (A Term selection Technique) performance’s is even worst than the Baseline. Surprisingly, the idea of upgrading the performance of PRF system upto paraphrasing system through newly introduced combinational techniques have been terribly failed as Borda count did not make a huge difference in the results and Intersection performed as bad as baseline. As a result, we conclude that Paraphrasing

system outperforms Rocchio until 15 number of paraphrases/15 number of documents but after that both perform equally good and kept increasing slightly until 20 number of paraphrases/20 number of documents. After that, all of a sudden paraphrasing performance kept decreasing.

## 7 Conclusion and future work

We introduced couple of techniques to improve the performance of biomedical document retrieval process: Combination techniques(Borda Count, Intersection) in Pseudo-Relevance Feedback and lexical paraphrasing technique. We compared the performance of combinational algorithms with individual term selection algorithms. To evaluate the performance of paraphrasing technique, we compared it with pseudo relevance feedback metrics. Surprisingly, in biomedical domain all introduced combinational techniques underperformed as compared to the other domains where they are performing quite well. Out of 7 term selection algorithms, IG, Rocchio and KLD performed pretty well whereas all other term selection algorithms failed to show a reasonable performance as a whole. Besides, paraphrasing performed 20% better than pseudo relevance feedback system, which is a reasonable rise in performance. Conclusively, performance of paraphrasing technique is way better than the pseudo relevance feedback system and it mainly depends upon two factors: ranking of newly generated paraphrases and the number of top chosen paraphrases. Moreover, paraphrasing technique requires high computational cost to calculate statistical information which play a key role in paraphrase ranking. In future, we shall be working with *word2vec* to calculate statistical information for paraphrase re-ranking.

**Acknowledgments** This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2016R1D1A1A09919551.

## References

1. Abdulla AAA, Lin H, Bo X, Banbhani SK (2016) Improving biomedical information retrieval by linear combinations of different query expansion techniques. *BMC Bioinformatics* 17(7):238
2. Asim MN, Rehman A, Shoaib U (2017) Accuracy based feature ranking metric for multi-label text classification. *Int J Adv Comput Sci Appl* 8(10):369–378
3. Asim MN, Wasim M, Ali MS, Rehman A (2017) Comparison of feature selection methods in text classification on highly skewed datasets. In: 2017 First International conference on latest trends in electrical engineering and computing technologies (INTELLECT). IEEE, pp 1–8
4. Bannard C, Callison-Burch C (2005) Paraphrasing with bilingual parallel corpora. In: Proceedings of the 43rd Annual meeting on association for computational linguistics. Association for Computational Linguistics, pp 597–604
5. Barzilay R, Lee L (2003) Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In: Proceedings of the 2003 conference of the North American chapter of the association for computational linguistics on human language technology, vol 1. Association for Computational Linguistics, pp 16–23
6. Barzilay R, McKeown KR (2001) Extracting paraphrases from a parallel corpus. In: Proceedings of the 39th annual meeting on association for computational linguistics. Association for Computational Linguistics, pp 50–57
7. Bouadjenek MR, Verspoor K (2017) Multi-field query expansion is effective for biomedical dataset retrieval. *Database*, 2017
8. Brill E (1992) A simple rule-based part of speech tagger. In: Proceedings of the workshop on speech and natural language. Association for Computational Linguistics, pp 112–116

9. Callison-Burch C (2008) Syntactic constraints on paraphrases extracted from parallel corpora. In: Proceedings of the conference on empirical methods in natural language processing. Association for Computational Linguistics, pp 196–205
10. Carpineto C, Romano G (1999) Towards more effective techniques for automatic query expansion. *Res Adv Technol Digit Lib*, 851–852
11. Claveau V (2012) Unsupervised and semi-supervised morphological analysis for information retrieval in the biomedical domain. In: COLING-24th International conference on computational linguistics
12. Cover TM, Thomas JA (1991) Entropy, relative entropy and mutual information. *Elem Inf Theory* 2: 1–55
13. Fang H (2008) A re-examination of query expansion using lexical resources. In: Proceedings of ACL-08: HLT, pp 139–147
14. Gonzalo J, Verdejo F, Chugur I, Cigarran J (1998) Indexing with wordnet synsets can improve text retrieval. arXiv:[cmp-ig/9808002](https://arxiv.org/abs/cmp-ig/9808002)
15. Harman DK (1995) The 3rd text retrieval conference (trec-3). NIST Special Publication, pp 500–225
16. Hiemstra D (2001) Using language models for information retrieval
17. Jinxi X, Bruce Croft W (1996) Query expansion using local and global document analysis. In: Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval. ACM, pp 4–11
18. Lee C, Lee GG (2006) Information gain and divergence-based feature selection for machine learning-based text categorization. *Inf Process Manag* 42(1):155–165
19. Lemur project. <http://lemurproject.org/lemur/indriquerylanguage.php>
20. Lin D (1998) Automatic retrieval and clustering of similar words. In: Proceedings of the 17th international conference on computational linguistics, vol 2 Association for Computational Linguistics, pp 768–774
21. Lytinen S, Tomuro N, Repede T (2000) The use of wordnet sense tagging in faqfinder. In: Proceedings of the AAAI00 Workshop on AI and Web Search
22. Majumder P, Mitra M, Chaudhuri BB (2002) N-gram: a language independent approach to ir and nlp. In: International conference on universal knowledge and language
23. Marton Y, Callison-Burch C, Resnik P (2009) Improved statistical machine translation using monolingually-derived paraphrases. In: 2009 Proceedings of the conference on empirical methods in natural language processing: vol 1. Association for Computational Linguistics, pp 381–390
24. Metamap a tool for recognizing umls concepts in text. <http://metamap.nlm.nih.gov>. Accessed 2015
25. Mihalcea R, Moldovan DI (1999) A method for word sense disambiguation of unrestricted text. In: Proceedings of the 37th annual meeting of the association for computational linguistics on computational linguistics. Association for Computational Linguistics, pp 152–158
26. Miller GA, Beckwith R, Fellbaum C, Gross D, Miller KJ (1990) Introduction to wordnet: an on-line lexical database. *Int J Lexicograph* 3(4):235–244
27. Mitra M, Singhal A, Buckley C (1998) Improving automatic query expansion. In: Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval. ACM, pp 206–214
28. Pérez-Agüera JR, Araujo L (2008) Comparing and combining methods for automatic query expansion. arXiv:[0804.2057](https://arxiv.org/abs/0804.2057)
29. Pubmed help. <http://www.ncbi.nlm.nih.gov/books/nbk3827>. Accessed 2015
30. Ramos J et al (2003) Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning, vol 242, pp 133–142
31. Roy D, Paul D, Mitra M, Garain U (2016) Using word embeddings for automatic query expansion. arXiv:[1606.07608](https://arxiv.org/abs/1606.07608)
32. Salton G, Buckley C (1997) Improving retrieval performance by relevance feedback. *Read Inf Retrieval* 24(5):355–363
33. Salton G, McGill MJ (1983) Introduction to modern information Philadelphia, Pa. American Association for Artificial Intelligence Retrieval
34. Sanderson M (1994) Word sense disambiguation and information retrieval. In: Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval. Springer-Verlag New York Inc., pp 142–151
35. Singh J, Sharan A (2015) Relevance feedback based query expansion model using borda count and semantic similarity approach. *Comput Intell Neurosci* 2015:96
36. Van Rijsbergen CJ (1977) A theoretical basis for the use of co-occurrence data in information retrieval. *J Document* 33(2):106–119
37. Whissell JohnS, Clarke CharlesLA (2011) Improving document clustering using okapi bm25 feature weighting. *Inf Retrieval* 14(5):466–487

38. Xiong N, Vasilakos AV, Yang LT, Song L, Pan Y, Kannan R, Li Y (2009) Comparative analysis of quality of service and memory usage for adaptive failure detectors in healthcare systems. *IEEE J Sel Areas Commun* 27(4):495–509
39. Xiong N, Vasilakos AV, Yang LT, Wang C, Kannane R, Chang C, Pan Y (2010) A novel self-tuning feedback controller for active queue management supporting TCP flows. *Inform Sci* 180(11):2249–2263
40. Xu J, Croft WB (2017) Query expansion using local and global document analysis. In: *ACM SIGIR Forum*, vol 51. ACM, pp 168–175
41. Zhao M, Ohshima H, Tanaka K (2016) Paraphrasing sentential queries by incorporating coordinate relationship. *J Inf Process* 24(4):721–731
42. Zhou Y, Zhang D, Xiong N (2017) Post-cloud computing paradigms: a survey and comparison. *Tsinghua Sci Technol* 22(6):714–732



**Muhammad Wasim** is Assistant Manager at KICS, UET and pursuing his PhD in Computer Science from University of Engineering and Technology, Lahore. He has industry experience of 6+ years and academic background of 6+ years. He has taught various subjects at UET, FC College, and UMT. His research interests include information retrieval, natural language processing, and machine learning.



**Muhammad Nabeel Asim** received his Bachelor degree from University of Management and Technology (UMT) and Master's degree in Electrical Engineering from University of Engineering and Technology, Lahore, Pakistan. Currently working as Research Officer at AlKhawarizmi Institute of Computer Science (KICS) University of Engineering and Technology (UET), Lahore Pakistan. His research interests are Bioinformatics, Artificial Intelligence, Machine Learning, Network Security, and Embedded Systems.



**Dr. Muhammad Usman Ghani** is an Associate Professor in the Department of Computer Science, University of Engineering and Technology, Lahore. His PhD (from Sheffield University, UK) study was concerned with statistical modelling for machine vision signals, specifically language descriptions of video streams. He has been studying on spoken language processing using statistical approaches with applications such as information extraction from speech and speech summarisation. His recent work is concerned with multimedia, incorporating text, audio and visual processing into one frame work.



**Dr. Zahoor Ur Rehman** has experience both in academia and research. He has received his educational and academic training at university of Peshawar, Foundation University Islamabad and UET Lahore, Pakistan. He joined COMSATS institute of information technology as assistant professor in the early 2015. Along with teaching responsibilities, he is an active researcher and reviewers of various conferences and reputed journals.



**Seungmin Rho** Ph.D. is a faculty of Department of Media Software at Sungkyul University in Korea. His current research interests include database, big data analysis, music retrieval, multimedia systems, machine learning, knowledge management as well as computational intelligence.



**Irfan Mehmood** has been involved in IT industry and academia in Pakistan and South Korea for over 6 years. Now serving as an assistant Professor in computer science and engineering department, Sejong University. His sustained contribution at various research and industry-collaborative projects gives him an extra edge to meet the current challenges faced in the field of multimedia analytics. Specifically, he has made significant contribution in the areas of visual surveillance, information mining and data encryption.