Medical Document Retrieval Using NLP

Gokul B J
Department of Computer Technology
Madras Institute of Technology
Chennai, India
gokulbj17082000@gmail.com

Ganesh Kumaar S
Department of Computer Technology
Madras Institute of Technology
Chennai, India
gks@panayan.com

Surya P
Department of Computer Technology
Madras Institute of Technology
Chennai, India
suryaparidevan333@gmail.com

***Abstract*—Search Engines play a major role in the success of Web. Search Engines help the user to find relevant information from a vast pool of data. For this purpose, people use most popular search engines which may not provide an accurate or relevant information. Most of these search engines just provides keywords matched documents from the user query which might not be efficient for the users. In this paper, we present a search engine that helps the ordinary users to search for medical information. Users like to receive relevant documents for each search. Ordinary Search Engine usually fails in finding the relevant document for complex medical query and will be difficult for an ordinary user to use that search engine. So a medical document retrieval system that gives an efficient output of documents related to the user query is a demand on the internet. This gap is being balanced by this project which focuses on providing an accurate/related medical document that the user requires. In this paper, a semantic based search engine is proposed. The proposed search engine can retrieve the document based on the semantic meaning of the query too. The semantic words are found out using word embedding techniques. These retrieved documents are ranked using score based algorithms which will be displayed to the user. Experiment shows that considering the semantic meaning of the query improves the performance of retrieval.**

***Index Terms*—Search Engine, medical query, semantic query, word embedding, document retrieval, ranking**

I. INTRODUCTION

Health-related content is one of the most searched-for topics on the internet, and as such this is an important domain for IR research. There are several dedicated search engines that seek to make this information retrieval more easily accessible. But most of the search engines provide general information in wholesome. So, we are here to provide a domain specific search engine which doesn't provide results based on a bit match of query terms and by the count of page views. We are in the process of designing a domain specific Information Retrieval System which has the best performing Parsing, Indexing and Ranking methods. Here we retrieve documents not only matching query terms, we use synonyms ,UMLS and query terms to retrieve documents. We also tend to train large scale data sets and retrieve a best matching document according to the user query.

The remainder of this paper is organized as follows. In Section II, we discuss some existing related works. The proposed work is introduced in Section III, then it follows our detailed design of components in Section IV. Implementation of our model is discussed in Section V. Simulation results are presented to validate the performance of our proposed method in Section VI. Finally, this paper is concluded in Section VII.

II. RELATED WORKS

The paper [6] works on a scalable approach based on information retrieval of documents from MEDLINE by query expansion using UMLS Metathesaurus. It uses Word Sense Disambiguation (WSD) to identify CUI (Concept Unique Identifiers) for query terms. This model uses the MetaMap tool for recognizing UMLS Concepts in Text and Word Sense Disambiguation (WSD) to identify which sense of a word is used in a sentence. It also Outperforms state-of-the-art methods when identifying terms. This model doesn't take into consideration of syntactic changes which was the drawback we identified.

The objective of the paper [14] is to retrieve documents using UMLS with the capability to retrieve datas to answer yes or no questions, fact-based questions and natural language questions for a biomedical question and answer system . This model also uses the MetaMap tool for recognizing UMLS Concepts in Text and BioPortal which is a comprehensive repository of biomedical ontologies. This model outperforms state of the art models when recognizing the question and classifying the retrieved data accordingly. We could recognize that the model does not take into consideration of relationships between various biomedical entities.

The paper [5] proposes a search engine that retrieves the most similar terms using cosine similarity between a query and word embedding model of the data set. The Data Set for the model is built from Mushaf Al-Tajweed. By Computing the average of vectors, one feature vector of the representing sentence is retrieved.Word2Vec technique for word embeddings using CBOW, Cosine Similarity for finding similarity between two words and KSUCCA table for exploring linguistics are used. It is observed here that words retrieved based on semantic meaning outperforms keyword-based retrieval. Same words with multiple meanings cannot be differentiated in this design which is the major drawback of the model.

The objective of the paper [8] is to retrieve the documents based on the user query from the heterogeneous data which builds a vector space model from the data set. Cosine Similarity for finding similarity between two words and comparing it with Moderated IDF-Cosine Similarity, Rstudio to develop this model and MapReduce Model for processing and distributed computing based on java. This model uses inverse document frequency and IDF modified cosine similarity which calculates effective cosine similarity values between Query and documents. The vector space model used can also be built by considering the semantic meaning of words, which might be a challenge.

The paper [18] works by extracting Biomedical documents , by improving the Biomedical Named Entity Recognition. A biomedical text corpus is extracted to form the data set. Deep Learning NER architecture is used with a character embedding layer and word embedding layer. Here tools like BioW2V using CBOW, BioW2V using skip-gram, DNN (Deep Neural Network), Apache Tika and PetScan are used for the development of the model. The accuracy of the results is comparable with complex BiLM models which adds advantage to the model. The gap found in this model is that the Word Embedding model can be built by using Glove or other context-based embeddings which gives high precision.

The paper [20] describes approaches based on information retrieval of documents using query expansion using the UMLS Metathesaurus. The documents are extracted based on the corpus-based features, resource-based features and term relevance using the extended queries. The use of corpus-based features, resource-based features and term relevance using the extended queries for document extraction adds advantage by increasing the accuracy of documents retrieved.MetaMap for recognizing UMLS concepts in text and MeSH thesaurus is used in this model. The Queries can be further expanded using any word embeddings technique which is a drawback for the model.

Paper [21] deals with BioMedical Information Retrieval using LTR. the objective of the model is to use the Optimal Ranking strategy to improve the performance of Information Retrieval. Unlike traditional Information retrieval, it has specific challenges due to the abundance of medical terminologies. Language Model and Vector Space Model functions are used along with Okapi BM25 and LTR algorithms. This model Outperforms other methods in our framework for biomedical retrievals like the Okapi baseline and NLMinter baseline. Indexing methods other than bm25 and VSM based features can be used because these functions reduce the total performance of the system.

Paper [1] constructs a Query Expansion and LTR based information retrieval system and also proposes new learning to rank (LTR) based query expansion models. This model attempts to tackle the problem of the abundance of terminologies by constructing ranking models, which focus on not only retrieving the most relevant documents but also diversifying the searching results to increase the completeness of the resulting list for a given query.TREETAGGER preprocessing tool is used here with LTR and SVM ranking algorithms . The advantage of the model is that it gives the best-ranked information retrieval for long queries. The challenge is that only Long queries are being concentrated in this model and results of short queries are not considered.

Paper [17] deals with Document ranking methods that induce information from the document's passages. The model aims to address the challenge of utilizing richer sources of passage-based information to improve the efficiency of retrieval. The model uses ClustMRF cluster ranking model, JPDm passage representation., SDM and LTR algorithms . The model helps in retrieving relevant passages that contain a relationship to the queries. Might have used additional passage-based features for better results.

The objective of the paper [11] is to retrieve biomedical documents using the classification and clustering of documents. It uses TF-IDF features that are weighted according to the type of query and term used. This model uses TF-IDF for a statistical measure that evaluates how relevant a word is to a document in a collection of documents and Named Entity Recognition (NER) to classify named entities. This model outperforms other methods like pseudo-relevance feedback and TF–IDF without weighting. The only challenge here is that the model is incapable of automatically determining the weights of the medical terms.

III. PROPOSED WORK

The architecture that we propose aims to provide better search results by using alternate queries. Initially the documents to be indexed will be cleaned and parsed followed by word embedding of keywords and NER (Named entity recognition). Then there will be a total of three queries that will be used to retrieve indexed documents:

- Query based on medical terms
- Query based on similarity
- Original query

For the medical term based query, a CUI (Concept Unique Identifier) and a TUI (Term Unique Identifier) will be found for each word in the query. The CUIs and TUIs will be extracted from medical terminology databases, the UMLS, MeSH, RxNorm, GO and HPO databases will be used. UMLS and MeSh will contain general medical terminologies while RxNorm, GO and HPO will contain terminologies specific to medical drugs, genes and human phenotype respectively.

The second query will be based on similarity. After the stop words and other irrelevant words are removed, the query will then be converted into word embeddings using *fast*Text. These word embeddings will be compared to the *fast*Text word embeddings of keywords from the index and based on cosine similarity to form the second query.

Finally the last query will be the original query after it has undergone pre-processing and stop words and irrelevant words have been removed from the original query. Once it has undergone preprocessing, POS tags will be attached to each term. Once the alternate queries have been formulated then will be used to retrieve the appropriate documents and then ranked and displayed to the user.

IV. DETAILED DESIGN OF COMPONENTS

A. MODULE 1

In the first module, the input data will be parsed and will undergo *fast*Text embedding. For each word, a corresponding POS (Part Of Speech) tag and a NER (Named Entity Recognition) tag will be attached. Then the POS tag, NER tag, *fast*Text word embedding along with the word itself and an unique document id are given for indexing.

PARSING

The input files are data cleansed and parsed using the Beautiful soup library and the lxml parser. The document's abstract text is not present inside a $< sec >$ tag, therefore it is extracted separately and stored into a temporary string followed by a new line. The remaining document data will be extracted based on the $< sec >$ tag and will be appended into the temporary string followed by a new line. Once the whole document is parsed, the temporary string will be written into a new file

POS AND NER TAGGING

For each term extracted from the parsed document, a POS and NER tag is retrieved with the help of spaCy and scispaCy libraries and models. The *en-core-sci-scibert* model is used for POS tagging and the following models are used for NER:

- *en-ner-craft-md* for TAXON and CHEBI entities
- *en-ner-jnlpba-md* for RNA DNA related entities
- *en-ner-bc5cdr-md* for chemical disease related entities
- *en-ner-bionlp13cg-md* for Organism and pathological related entities

Once each term from the document has been associated with a POS and NER tags, they will undergo *fast*Text word embedding.

FASTTEXT WORD EMBEDDING

For each term extracted from the parsed document, preprocessing will happen where stop words and redundant data such as special characters, single characters are removed and multiple white space characters are replaced with a single white space character with the help of regular expressions and Python's re library. Then the data is converted into lowercase and made into a 2-D array which is then tokenized and used to train the model over 20,000 files. The model uses the Skip-Gram model with a window size of 40 and embedding size of 60. The minimum frequency is set to 5 with down sampling at 1e-2.

B. MODULE 2

In this module, the user query will be given as input and the document will be retrieved as output. The given user query will be converted to a formulated query which is formed using original query, similarity query and medical term query. From the formulated query, the document IDs will be retrieved.

ENTITY LINKING

The Entity Linking of medical terms done in this module which will be used in indexing. From scispaCy, 5 models are imported:

- Unified Medical Language System
- Medical Subject Headings
- RxNorm
- Gene Ontology
- Human Phenotype Ontology

Using these 5 models, CUI (Concept Unique Identifier) and TUI (Term Unique Identifier) will be retrieved for each term. Umls Entity Linker will give generic CUI and TUI. MeSH will give generic CUI. RxNorm, Gene Ontology, Human Phenotype Ontology will give the CUI and TUI related to the field of drugs, genetics and phenotypic abnormalities in the human diseases respectively. The common terms will be combined to reduce the space.

GROUPING AND INDEXING

The trained *fast*Text Model is loaded to get the word vector. The trained data for POS and NER tags and Entity Linking will be in 2 different files. These two files have to be grouped together for indexing. Pickle files are used to store the grouped data where the data is stored in the list format of WORD, ID, POS, NER, CUI, TUI, WORD VECTOR. From the pickle files, all the data will be loaded and stored into a CSV file.

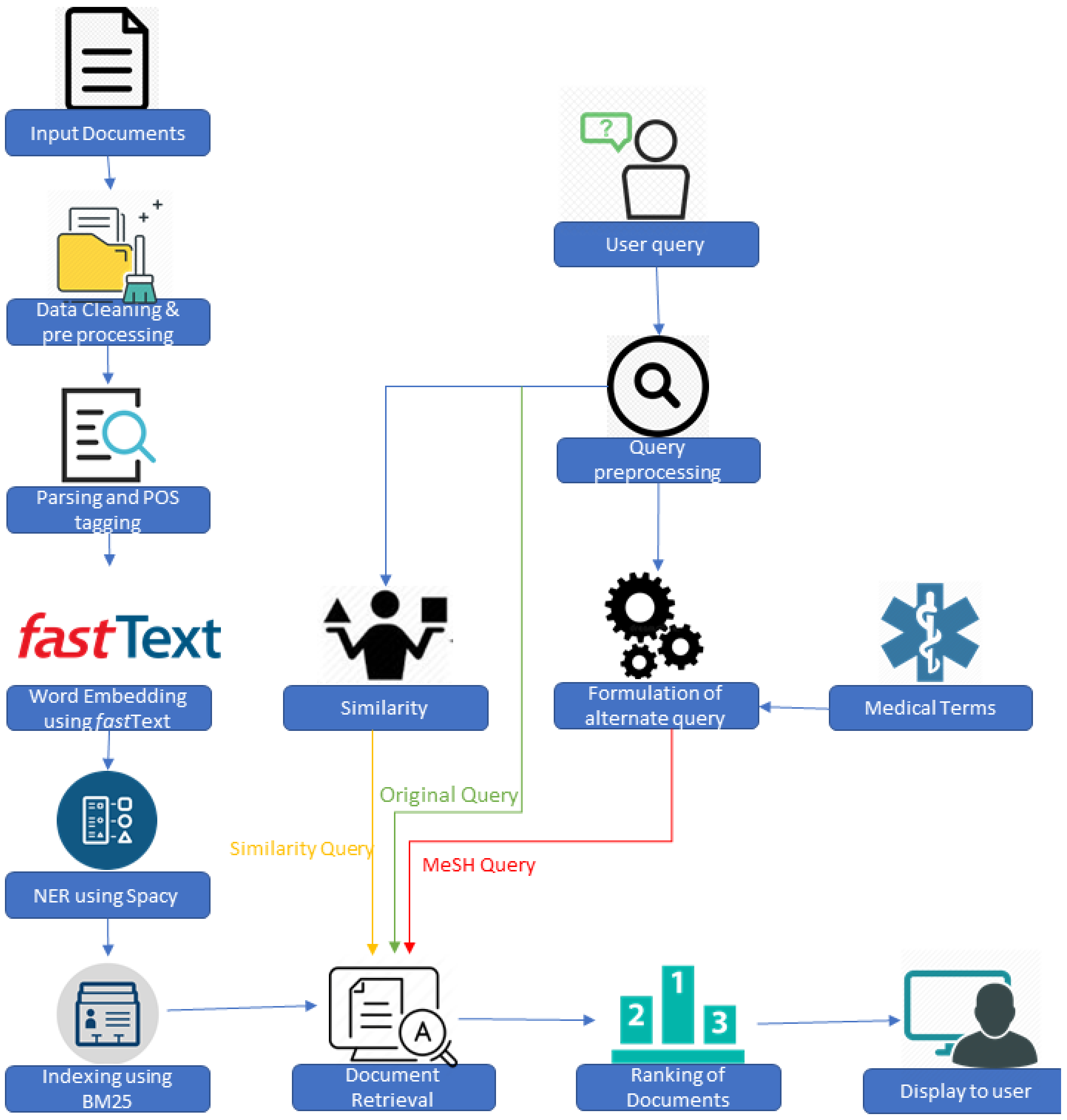

Fig. 1. ARCHITECTURE DIAGRAM

QUERY FORMULATION

The 5 medical knowledge bases discussed in the ENTITY LINKING phase and the 4 spaCy models used in MODULE 1 to generate POS and NER tags will be loaded in this phase. For the given user query, UMLS links, MeSH links, RxNorm links, GO links and HPO links containing CUI and TUI tags and NER and POS tags will be generated. These are grouped together to form the MEDICAL TERM QUERY. The given query will be processed with the help of NLTK packages to form the ORIGINAL QUERY. Using the trained *fast*Text model, the similar terms for each term in the query will be retrieved and grouped together to form SIMILARITY query. Once these three alternate queries have been formulated they will be used to retrieve relevant documents from the index.

DOCUMENT RETRIEVAL

The indexed files will be fetched and stored in a pandas data frame. Each fetched column will be changed to their respective format using applymap() function. The new column will also be created to store the similarity score. Based on the terms in the original query and similarity query, all IDs will be retrieved based on each term in the query. From the CUI and TUI terms formed in the medical term query, respective IDs will be retrieved. Cosine similarity score is generated using indexed word vectors and the word vector found for each term in the query from which IDs will be retrieved. The IDs will be grouped together and given to the next module.

Algorithm 1 Algorithm to extract medical term query

```
Input : ($Preprocessed\_query$)
Output : ($Medical\ term\ query$)
1: function EXTRACT_MEDICAL_QUERY($query$)
2:     Load knowledge bases ($UMLS$, $MeSH$, $RxNorm$,
3:     $GO$, $HPO$)
4:     Create Linkers for each knowledge base
5:     Attach linkers to NLP pipes with NER models
6:     $Medical\_terms$=[]
7:     for ($i \in Preprocessed\_query$ ) do
8:         Retrieve CUIs and TUIs for each term from
9:         knowledge base
10:        Combine and remove duplicates
11:        $Medical\_terms$.extend(combined CUIs and
12:        TUIs)
13:    end for
14:    Return $Medical\ term\ query$
15: end function
```

Algorithm 2 Algorithm to extract similar query

```
Input : ($Preprocessed\_query$)
Output : ($Similarity\ based\ query$)
1: function EXTRACT_SIMILAR_QUERY($query$)
2:     Load fastText models
3:     Fetch the keywords word vectors which is built by
4:     FASTTEXT
5:     $Similar\_query$ = []
6:     for ($i \in Preprocessed\_query$ ) do
7:         The fastText query and fastText keyword vectors
8:         using cosine similarity
9:         Extract the top ranked terms
10:        $Similar\_query$.extend(top ranked terms)
11:    end for
12:    Return $Similar\_query$
13: end function
```

Algorithm 3 Algorithm for query formulation

```
Input : ($Preprocessed\_query$)
Output : ($Formulated\ query$)
1: function FORMULATE_QUERY($Preprocessed\_query$)
2:     $Medical\ query$
3:             =Extract_medical_query($Preprocessed\_query$)
4:     $Similar\_query$
5:             =Extract_similar_query($Preprocessed\_query$)
6:     Return $Preprocessed\_query$, $Medical\ query$,
7:     $Similar\_query$;
8: end function
```

C. MODULE 3

RANKING OF DOCUMENTS

The input to this module is the list of documents. For the three different queries, their respective documents are fetched. Each query is given a respective weight. The high priority is given to the original query and next priority is given to similar term query and least priority is given to medical term query. The frequency of each term occurring in a document is calculated and then multiplied with the respective weight and stored. These stored values will be ranked and 1% of the document is displayed. The output of this module is the ranked list of documents.

Algorithm 4 Algorithm for Document Retrieval and Ranking

```
Input : ($preprocessed\ query$,$w1$,$w2$,$w3$)
Output : ($List\ of\ Documents$)
1: function RETRIEVAL_RANKING_QUERY($query$)
2:     score={};
3:     Preprocess the query;
4:     original_query,similar_query,medical_query
5:             =Formation_Query($preprocessed_query$)
6:     for ($i \in preprocessed\ query$ ) do
7:         doc1 = Retrieve all doc ID with term i;
8:         for ($j \in doc1$ do
9:             score[j]+=w1;
10:        end for
11:    end for
12:    for ($i \in similar\ query$ ) do
13:        doc2 = Retrieve all doc ID with term i;
14:        for ($j \in doc2$ do
15:            score[j]+=w2;
16:        end for
17:    end for
18:    for ($i \in medical\ query$ ) do
19:        doc3 = Retrieve all doc ID with term i;
20:        for ($j \in doc3$ do
21:            score[j]+=w2;
22:        end for
23:    end for
24:    DOC_ID list based on keys in decreasing order of
25:    values;
26:    Return 1% of DOC_ID
27: end function
```

FRONT END

The front end created using Flask and the ng rok library will be used to expose the local port of the colab server to make it accessible to the user and also allow multiple users to access the server at the same time. The Jinja templating language that comes prepackaged with Flask will also be used to create predefined templates and rules. From the server, the top ranked documents will be retrieved and displayed properly.

TABLE I
EVALUATION METRICS

Query	F1 Score	Precision
Branycardia with sinus arrhythmia	96.37%	92.99%
posterolateral hypothalamic hypocretin neurons	95.78%	91.90%
Benzodiazepine receptor agonist	98.50%	97.03%
Glasgow coma score and unconscious	89.89%	81.64%
Involuntary trembling or quivering	75.41%	60.53%
Thrombocytopenia	98.95%	97.92%
Tachycardia with shortness of breath	93.37%	87.56%
Arrhythmia with normal heart rate	98.12%	96.32%

SQLAlchemy will be used for maintaining a database for keeping track of the searches made and displaying the history page.

V. IMPLEMENTATION

In this design, the datasets are taken from BioASQ. BioASQ organizes challenges on biomedical semantic indexing and question answering. The challenges include tasks relevant to information retrieval. The development dataset from BioASQ consists of biomedical questions in English, along with their gold concepts, articles, snippets, RDF triples, ”exact” answers, and ”ideal” answers, MeSH terms assigned to the articles by the human curators in PubMed, biomedical articles published in PubMed and corresponding full text from PMC. This dataset from BioASQ contains articles in the format of XML files. Journal title, abstract, body and article title are the field areas in the dataset which contains information of the document and helps in training and to rank documents for retrieval. The dataset for task 5C from BioASQ is chosen here and 37,815 files in total have been collected for this project from BioASQ.

The design discussed is this paper is implemented using python language. For the development of this project Google Colab, a Python development environment is used. The Data Set is pre-processed using Beautiful Soap which helps to remove the XML tags. The *fastText* word embedding is used to find the simialar term for a term. For training the word embedding, a total of $20,000$ documents is used with the number of epoch as 5. After training, the model is stored. For the extraction of medical query, *UMLS*, *MeSH*, *RxNorm*, *GO*, *HPO* models are used. Each model helps is extracting in the mediacal term query.

All the words in the document and the medical term for each word will be indexed. When a query is given, the query will be pre-processed by the user and the pre-processed query will be used to extract the similar term and the medical term. These 3 queries will be used to retrieve the document. After retrieving the document, each document ID will be given a score based on the frequency of the document occuring and based on the weight given for the respective type of query. By using the score, the documents will be ranked. 1% of retrieved document will be displayed to the user.

VI. RESULTS

Looking through current existing models of medical domain information retrieval system we would see some famous medical sites such as PubMed, MEDLINE, PubMed Central etc. but each of these sites only works on a basic term match mechanism and provides us with a bulk of matched terms of documents which would be both relatable and unrelatable to our search query. If we look through PubMed it works by searching the query term with the MeSH database and picks documents from the articles they have with the related terms and displays those documents.

The MEDLINE database is is a subset of the PubMed database which obtains data from NLM's Data Distribution program. Comprehensive journal selection process (NLM) controlled vocabulary, Medical Subject Headings (MeSH), to index citations makes MEDLINE differ from PubMed. PubMed Central (PMC) acts as a counterpart to NLM journal collection and there are identifiable links between the full text in PMC and corresponding citations in PubMed. This shows how these major sites are being interconnected.

The approach in this paper is mainly introduced to increase the precision of relevant documents retrieved. Traditional search engine searches only by the keyword. Model designed in this paper considers 3 different queries which gives high number of relevant documents. F1 score is measured to show the improvement of our model. F1 score is measure calculated using precision and recall. Precision is the number of relevant documents in the retrieved documents. Recall is the number of relevant documents retrieved from the total relevant documents which is theoretically 1. F1 score is found using formula,

$$F1 = \frac{(2*precision*recall)}{(precision+recall)}$$

The precision is found using formula,

$$precision = \frac{relevant\ documents}{retrieved\ documents}$$

The precision and F1 score obtained in our model show our search engine is more effective and accurate than the traditional search engine.

VII. CONCLUSION AND FUTURE WORK

We have developed a system that efficiently retrieves documents than these present models which works by term matching and retrieves a sack of documents that may be related or unrelated to search query. We conclude that we have constructed a document retrieval system that works in a better way having a considerably good accuracy, precision, and F-score. We have these metrics by comparing only our specific data-set manually which showed up a static increase in the percentage of accuracy and F-score higher than the percentage mentioned in the BioASQ site where we incorporated our data set.

In future works, we have planned to work to use ELMo instead of fastText. It is a state-of-the-art NLP framework developed by AllenNLP and is used to represent words in vectors or embedding in a novel way. The ELMo vector assigned to a token or word is a function of the entire sentence containing that word Unlike traditional word embeddings such as word2vec,fastText, and GLoVe. Therefore, the same word can have different word vectors under different contexts. We have also planned to bring changes in our ranking methodology where we are to introduce an LTR ranking algorithm, Since we have only used F-score-based ranking in the current model. So this would increase the accuracy of the documents retrieved and might also contribute to the time reduction in document retrieval.

REFERENCES

[1] Bouziri, A., Latiri, C. and Gaussier, E. (2020). LTR-expand: query expansion model based on learning to rank association rules. Journal of Intelligent Information Systems, 55(2), pp.261–286

[2] C. Kohlschein, D. Klischies, A. Paulus, A. Burgdorf, T. Meisen and M. Kipp, (2018) An Extensible Semantic Search Engine for Biomedical Publications, IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 1-6

[3] D. H. Fudholi and L. Mutawalli, (2018) A Lightweight SemanticBased Medical Document Retrieval, 6th International Conference on Information and Communication Technology (ICoICT), pp. 147-151

[4] Diaz, F., Mitra, B. and Craswell, N. (2016). Query Expansion with Locally-Trained Word Embeddings. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers, pp. 367–377

[5] Mohamed, E.H. and Shokry, E.M. (2020). QSST: A Quranic Semantic Search Tool based on word embedding. Journal of King Saud University - Computer and Information Sciences

[6] Nawab, R.M.A., Stevenson, M. and Clough, P. (2017). An IR-Based Approach Utilizing Query Expansion for Plagiarism Detection in MEDLINE. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 14(4), pp.796–804

[7] Neveol, H. Dalianis, S.Velupillai, G.Savova, and P.Zweigenbaum, (2018) Clinical natural language processing in languages other than English:opportunities and challenges, Journal of biomedical semantics, vol. 9,no. 1, pp. 12

[8] Pathak, B. and Lal, N. (2017). Information retrieval from heterogeneous data sets using moderated IDF-cosine similarity in vector space model.International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), pp. 3793-3799

[9] Prabhu, L. A. J., Sengan, S., Kamalam, G. K., Vellingiri, J., Gopal, J., Velayutham, P., Subramaniyaswamy, V. (2020). Medical Information Retrieval Systems for e-Health Care Records using Fuzzy Based Machine Learning Model. Microprocessors and Microsystems

[10] Sankhavara J. (2018). Biomedical document retrieval for clinical decision support system. In: Proceedings of ACL 2018, student research workshop. pp. 84–90

[11] Sankhavara, J. (2020). Feature Weighting in Finding Feedback Documents for Query Expansion in Biomedical Document Retrieval. SN Computer Science, 1

[12] Sarrouti, M. and Ouatik El Alaoui, S. (2017). A Biomedical Question Answering System in BioASQ 2017.Association for Computational Linguistics, pp 296–301

[13] Sarrouti, M. and Ouatik El Alaoui, S. (2017). A passage retrieval method based on probabilistic information retrieval model and UMLS concepts in biomedical question answering. Journal of Biomedical Informatics, 68, pp.96–103

[14] Sarrouti, M. and Ouatik El Alaoui, S. (2020). SemBioNLQA: A semantic biomedical question answering system for retrieving exact and ideal answers to natural language questions. Artificial Intelligence in Medicine, 102, p.101767

[15] Sarrouti, M., El Alaoui, S. O. (2016). A Generic Document Retrieval Framework Based on UMLS Similarity for Biomedical Question Answering System. Smart Innovation, Systems and Technologies, pp. 207–216

[16] Schulze, F., Schuler, R., Draeger, T., Dummer, D., Ernst, A., Flemming, ¨ P., Perscheid, C. and Neves, M. (2016). HPI Question Answering System in BioASQ 2016. In: Proceedings of the Fourth BioASQ workshop at the Conference of ACL, 2016, pp. 38–44

[17] Sheetrit, E., Shtok, A. and Kurland, O. (2020). A passage-based approach to learning to rank documents. Information Retrieval Journal, 23(2), pp.159–186

[18] Silvestri, S., Gargiulo, F. and Ciampi, M. (2019). Improving Biomedical Information Extraction with Word Embeddings Trained on ClosedDomain Corpora.2019 IEEE Symposium on Computers and Communications (ISCC), pp. 1129-1134

[19] Tran, T. and Kavuluru, R. (2019). Distant supervision for treatment relation extraction by leveraging MeSH subheadings. Artificial Intelligence in Medicine, 98, pp.18–26

[20] Xu, B., Lin, H. and Lin, Y. (2019). Learning to Refine Expansion Terms for Biomedical Information Retrieval Using Semantic Resources. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 16(3), pp.954–966

[21] Xu, B., Lin, H., Lin, Y., Ma, Y., Yang, L., Wang, J. and Yang, Z. (2018). Improve Biomedical Information Retrieval Using Modified Learning to Rank Methods. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 15(6), pp.1797–1809