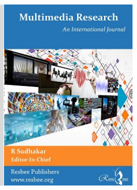
Resbee Publishers

Multimedia Research (MR)

Received 06 April, Revised 19 April, Accepted 25 May



BERT Representation for Arabic Information Retrieval: A Comparative Study

Moulay Abdellah KASSIMI
National School of Applied Sciences,
Ibn Zohr University, Agadir, Morocco
moulayabdellah.kassimi@edu.uiz.ac.ma

Abdessalam ESSAYAD
National School of Applied Sciences,
Ibn Zohr University, Agadir, Morocco
a.essayad@edu.uiz.ac.ma

Abstract: Information is rapidly growing in online documents and social media in all languages. Retrieval of information from a language is a high-level task. However, Information Retrieval has become more important in research and commercial development. Presently only a few tools were available in the market for retrieval. Each language has its unique way of pronunciation and language structure. Arabic has a complex morphology. This made it difficult in the advancement of this field. A typical IR model is required to understand similar words in the matching process. In this paper, we presented a comparative study on recent approaches in Arabic Information Retrieval. We implemented and compared all existing approaches for Arabic IR with Arabic datasets. The information retrieval used an Arabic dataset. We also introduced a dictionary, an Arabic Lemmatizer.It contains Arabic words collected from several Arabic books and websites. We compare the performance of different lemmatization techniques. Then we conduct a series of experiments to compare different approaches to Arabic IR. Furthermore, Arabic BERT examined the superior performance with the existing approach's performance. The experimental result showed BM25 and multilingual BERT ranked most for tasks. The Large Arabic Dataset scored an accuracy of 89% in information retrieval.

Keywords: Information Retrieval; NLP; Context; Semantics; Search Engine; Word embeddings; Transformers; Deep learning; BERT; BM25; Re-ranking.

Nomenclature

Acronym	Description
NLP	Natural Language Processing
BERT	Bidirectional Encoder Representations from Transformers
GPT	Generative Pre-trained Transformer
MAP	Mean Average Precision
CBOW	Continuous Bag-of-Words
ANN	Approximate Nearest Neighbour
RegEx	Regular expressions
BOW	Bags Of Words
DFR	Divergence From Randomness
VSM	Vector Space Model

1. Introduction

With recent advances in NLP, many researchers are interested in developing business web applications that leverage retrieval information techniques such as search engines, recommender systems, question-answering, and sentiment analysis. NLP gives computers the ability to understand human language [13] and recently pre-trained models and transformers have revolutionized this field. The pre-trained models such as BERT [9] and GPT [32] take context into account and can represent a whole sentence using sophisticated embeddings for documents.

These models were trained with large language datasets and we can create a high-quality model with minimal effort that NLP can fine-tune for specific tasks. The Transformer aims to solve sequence-to-sequence tasks relying on self-attention to compute representations of its input and output [34]. Therefore, Transformers can consider the context of a word and then use it to understand the intent behind the search query. Furthermore, Transformers can take learnings from one language and apply them to others. So, we can take models that are performed in English and apply them to the Arabic language.

Currently, applying BERT models to the Arabic language remains a challenging task that has complicated orthography and morphology compared to other languages. Indeed, the Arabic language contains diacritics that are placed above or below letters; thus, different diacritics can result in different meanings and ambiguity in Arabic documents. In particular, in the search process, users have abandoned these diacritics, and search engine systems are expected to understand the lost diacritics [11]. Furthermore, users are not always sure of the best way to formulate a request and they may not know what words to use or how to spell them. These complexities affect the searching task which cannot reach a high performance compared to the IR in English and other languages. Moreover, for Keyword-based search engines, the complex morphology of Arabic makes Tokenization in the pre-processing step more difficult as it is mainly based on the identification of word roots. However, this paper aims at investigating the impact of different lemmatizing techniques on result performance and to compare the performances of Arabic pre-trained models. This will be done through a search engine focused on models' comparison to identify the best overall recovery approach available for the Arabic language. The Arabic morphology makes lemmatizing and stemming difficult to perform. To overcome these difficulties, we proposed a study on Arabic IR with the objective,

- To evaluate our approach to Arabic language lemmatizing and to test the impact of different lemmatizing techniques on performance.
- To create an efficient implementation of Arabic Information Retrieval using BERT, and to compare it with other IR to fetch the best results quickly.

In this paper, we include our lexical database for the Arabic language, aiming to overcome the challenge of Arabic language lemmatizing and stemming. We focus on combining keyword-based matching and semantic search using different search methods to increase accuracy. The goal is to perform a lexical and semantic search, taking advantage of the Lucene engine and the contextual embeddings to index our Arabic corpus. Therefore, to extend the power of great models like BERT to the Arabic language. From this method, BM25 and multilingual BERT reranking reached an 89% accuracy score on a large dataset collection.

The organization of this paper is in this order: Section 2 presents the related works; Section3 portrays the text pre-processing and ARABIC language. The method is illustrated in Section4; Section 5 covers the data collection; Section 6 provides the datasets;Section 7 covers the evaluation measures. The implementation and results were covered in Section 8; Section 9 provides the estimation of similarity measures; Section 10 covers improving search engine efficiency. Advantages and Disadvantages were covered in section 11 and finally, Section 12 concludes the manuscript.

2. Related works

The main objective of information retrieval wasto retrieve relevant documents from a corpus using an appropriate model such as Boolean Model [23] or Vector Space Model [30]. The Boolean model creates a query in a Boolean expression terms structure which was combined with the operators AND, OR, and NOT [31].It represents a request to determine what documents contain a given set of keywords and then returns documents that exactly satisfy the Boolean query. In contrast, documents and queriesin the vector-space model are represented by a vector of index terms with related weights. These weights represent the importance of the index [20]. Keyword-based search engines used to be built around BM25 and TF-IDF-based vector space models that are easy to use and are effective, if the researcher knows exactly what he is looking for. BM25, for Best Matching, wasa ranking function used by search engines to rank matching documents according to their relevance to a given search query [15]. While TF-IDF, for term frequency-inverse document frequency, gives more weight to words that wereunique to a document than to frequent words. In [21] BM25 and TF-IDF were compared in terms of feature extraction for classification and its experiment improves the performance of TF-IDF according to the value of the F1-measure.

RegEx can be applied to search enginesto overcome the limitations that keyword-based search systems can present. Indeed, with RegEx, we can use pattern matching to search for particular strings and then refine the search. In [12], RegEx has been used for data processing tasks such as extracting information as well as refining the search to reduce the number of search results displayed to the user. The author [27] proposed the combination method of keyword-based and RegEx searches. To perform a search, authors embed RegEx patterns as part of the search engine's algorithm in their application.

Implementing a search engine application was a complex and challenging task, especially for the Arabic language. The authors [4] developed a system that had been validated through an Arabic Information retrieval website. The authors in their experiments show the effectiveness of the vector space model approach to retrieving relevant information. Most search engines treat documents and queriesas BOW, a statistical model for information retrieval based on Markov random fields [24]. Such a system usually uses words as features to decide which documents are most likely to be relevant. Given a set of documents and a text query, the BOW-based search systems werestill inuse and are the simplest IR Systemsbut very inefficient and cannot represent terms by meaning.

On the other hand, Semantic Search techniques enable search engines to understand the meaning and intent behind the user's query and provide more accurate search results [17]. The number of ontology-based semantic search engines had been continuously growing in the last decade. In our previous work [22], we used an ontology to build a semantic search engine and showed improved results for search and retrieval in a 3D database. However, the semantics supported by ontology wasnot sufficient and does not help much in semantic information retrieval, especially for searching text.

Recently to tackle this problem, many word embedding techniques hadbeen proposed. Nevertheless, Word2Vec [25], Glove [28] and Fasttext [7] were the most popular semantic embedding techniques. The authors [6] presented a comparison between Word2Vec and Glove. Word2Vec predicts words based on their context by using one of two distinct neural models CBOW and Skip-gram. CBOW uses embeddings to train a neural network where the context was represented by multiple words for a given target word [37]. Skip-gram wasused to predict the context word for a given target word. As discussed [19], the performance of each combination wasvery close to the Arabic language but word2vec gives the best vector representation on its own. The N-gram model was used in search enginesto build an auto-completion program or search suggestions. It was important for many NLP applications, and it wasused to predict a word based on its previous and next words sequence in the works. Ismail and Rahman [19] constructed the clusters using semantic and contextual similarity. The same cluster contained words that had similar meanings and wereused in similar contexts in a document. The Word2vec model was followed by a Fast text model [7], the library produced by Facebook. It is one of the most powerful and fast libraries for dealing with texts in which word embeddings were constructed from a bag of character n-grams. The fast text model wasa relatively new model in which the library contains hundreds of millions of words that have been trained on Wikipedia and Common Crawl data. It was strong [37] and showed better performance than word2vec and the best was that it encourages the Arabic language. AraVec [3] embeddings, also pre-trained Arabic embeddings are word2vec vectors trained on Tweets and Wikipedia Arabic articles that had wide coverage of dialect words. AraVec pre-trained word embeddings were used [16] to create a word embedding representation for Arabic sentiment analysis. However, it takes a long time to upload.

Word embedding techniques previously mentioned like Word2vec and Fast text aimed at representing words by capturing their semantic properties. Unfortunately, Words had different meanings in different contexts and these models didn't incorporate the context. To address this issue, new models generated contextualized representations [29]. Authors used ELMO to create vectors derived from a bidirectional LSTM that was trained on a large text corpus. The new approaches were Transformer and BERT. Transformers based on the self-attention mechanism consider the full context of a term by looking at the terms that come before and after it to understand the intent behind a search query. The performance on different tasks improved, leadingto larger structures that had sentence representations such as BERT [9] and T5 [26], which offered improved performance by exploring different pre-training techniques and huge training data. BERT was powerful and will help us accurately retrieve results matching the learner's intent and contextual meaning. It provides embeddings for words as well as documents and, recently, there had been a focus on fine-tuning BERT with small data.

For the Arabic language, Google produced a multilingual BERT [9] supporting 150+ languages which showed a better performance for most languages. However, pre-training monolingual BERT for the Arabic language proved to provide better performance than multilingual BERT such as AraBERT [26][38][1]. Unfortunately, these models weretested for Question Answering applications. Moreover, the best results werecurrently achieved by fine-tuning AraBERT [38]. The authors [37] emphasize that we can get better results by combining AraBERT and AraVec. Recently, Many Arabic models hadbeen pre-trained such as MARBERT [14], CAMeLBERT [18], and QARiB [5]. Therefore, we use these and other pre-trained models to compare their abilities to create a semantic search engine for the Arabic language.

2.1 Review

Table 1 portrays the methodology, advantages, and disadvantages of the existing method. We considered ten papers that used a different methodology that used IR at various levels. Each method has certain benefits and shortcomings, that were explained in detail.

***Table 1:** Review Based on Existing Methods*

Author	Methodology	Advantages	Disadvantages
Lashkari, A. H., *et al.* [23]	IR	• Easy to implement.	• Didn't work well for large datasets.
Platzer. C. *et al.,* [30]	VectorSpace Search Engine	• Provided valuable browser repositories.	• May not be suitable for all use cases.
Joho H. and M. Sanderson, [20]	WordNet	• Effective estimation in specific terms, particularly at the very specific levels that cover the majority of vocabulary.	• They did not appear frequently enough in the corpus to provide an accurate measure of document frequency.
Kadhim, A. I., [21]	DFR	• Addresses the collection-dependence problem	• Evaluated on TREC collections only.
Frenz, C. M., [12]	Pubmed	• Regular expressions can be used to identify patterns in data that may not be easily searchable using keyword-based methods. • Used to get extract specific information from large datasets. • Used to refine search results. • Reduced time in manually filtering.	• Regular expressions can be complex and difficult to write correctly, especially for those who are not familiar with programming languages.
Onyenwe, I., *et al.*[27]	RegEx	• Provided more comprehensive search results. • More flexible and powerful search patterns. • Applied beyond web searches, such as information extraction and search refinement.	• Required more computational resources.
Abu Salih., *et al.*[4]	VSM	• Ability to retrieve documents based on their relevancy degree. • Get relevant feedback from users of the system. • Effectiveness of the VSM model and is valid in the Arabic language.	• Affected based on factors such as the quality of the data, the size of the dataset, and the choice of parameters used in the model.
Hu, J., *et al.* [17]	Wikipedia knowledge	• The best human knowledge bases. • More comprehensive and accurate representation of the intent domains. • Achieved much better coverage to classify queries. • It is a very general method and can be easily applied to various intent domains.	• Relied on quality and completeness. • Didn't work well for intent domains that are not well-represented in Wikipedia. • Didn't suitable for real-time applications
Kassimi, *et al.* [22]	Extension of the triliteral algorithm	• It runs faster than other algorithms.	• Didn'tgenerate adjectives and • generate different types of derived nouns.
Mikolov T, *et al.* [25]	Continuous BOW and Skip-gram	• Efficient and faster, requiring less computational resources. • Learn high-quality word vectors from very large datasets with billions of words. • Perform better on a word similarity task compared to other neural network-based methods. • Pre-trained word vectors are publicly available.	• The quality of the word vectors may depend on the quality and size of the training data.

2.2 Challenges

The existing method for information retrieval faces several challenges that need to be addressed through research. One such challenge is data sets [23][25][17][27] a set of keywords that are used for IR. Some methods didn't work well for large datasets and some methods doesn't have enough datasets to proceed.

Additionally, the accuracy and the quality of information are low as highlighted in [25] [22] [4][17]. Hence, there is a need for developing new and more effective techniques to enhance the accuracy and quality of the information.

3. Text Preprocessing and Arabic language

A pre-processing step is crucial for information retrieval applications. It aims to identify the optimal form of the term to achieve the best retrieval performance. In our system, we first introduce methods for cleaning each sentence in the corpus removing noise and many unnecessary words. Second, we introduce a tokenization method to break a stream of text into tokens. Finally, we introduce our lexical database for the Arabic language aiming to overcome the challenge of Arabic language lemmatizing and stemming.

Stemmers attempt to reduce morphological variations of words to a common stem, while Lemmatization involves making an appropriate reduction to the dictionary form, called a lemma.

The Arabic language is characterized as anetymological language or phenomenon of derivation, meaning that the word is taken from its root. Words share their root letters, origins, and arrangement. Therefore, most Arabic words are generated from a common root. This common root leads to a common meaning between words, whose type is determined by the root form. In the Arabic language, there are also morphological patterns, which is the criterion on which the conjugation of verbs depends. It is a standard developed by Arab scholars to know the conditions of word structure, and it is one of the best standards known for controlling languages. Furthermore, in all languages, some words have multiple meanings determined according to their position in the sentence or context. The problem of confusion in meaning is one of the biggest challenges facing Arabic NLP and its applications, especially information retrieval, in addition to the problem of pronouns that may refer to the person or thing. There is another ambiguity due to the structure of the sentence, where the sentence means more than one meaning or can be interpreted in more than one way, such as "The father hit his son because HE was drunk" (“ ضَرَبَ الأَبُ ابْنَــهُ لِأَنَّــهُ كانَ مَخْمورًا”), where the personal pronoun can refer to the son or the father. In fact, in the pronoun, it is permissible to refer to the father and the son, and this is only determined by the context.

Fortunately, Lemmatization and stemming allow us to reduce this meaning ambiguity by grouping related words and removing prefixes and suffixes according to the morphological rules. Furthermore, lemmatization tries to group words by their meaning.This, however, is more complicated and needs information about the language to split the sentence in a way that makes sense.

There are two major classes of lemmatization available in the literature: Algorithmic lemmatization and Dictionary lemmatization. Algorithmic lemmas apply rules to a word to reduce it to its root form. They work well for regular words, they are fast and use little memory. Unfortunately, Arabic is a language with poor algorithmic lemmas. Dictionary lemmatizes words to their dictionary forms rather than applying rules to each one. Therefore, A dictionary lemmatize is only as good as its dictionary. In our experiments, proposed dictionary lemmatizes are fed into the existing Arabic dictionary by taking advantage of the Lucene engine where the size of the dictionary does not have an impact on research results.

In the preprocessing step, not much semantic meaning is captured by the keyword-based model. Word embedding is a new technique in which each word or phrase is mapped to an N-dimensional vector. Therefore, a simple tokenizer corpus works better compared to more complex preprocessing techniques such as lemmatization or stemming [8]. First, we are losing valuable information that would help deep learning. Second, most embeddings have preprocessed their corpus in their way. However, it is better to improve our data quality by cleaning up the corpus to remove some punctuation marks to take into account the context of the word.

4. Methods

Semantic search engines compare user queries with each text from a set of documents, calculate a similarity score for each comparison, and get the n-top results based on the similarity scores. To provide an effective search engine over a large corpus, we incorporate an indexer into our system. We have tried various approaches to make the best decision possible, particularly in light of the problem of the inaccessibility of Arabic information.

Initially, we compared in the first experiment our dictionary lemmatizer with three other lemmatizing methods. These methods are among the most widely used for Arabic lemmatization.Afterthat, we tried different techniques in both approaches: keyword-based systems and semantic-based systems. First, we started by testing a few methods, including BM25 and TF-IDF in the keyword-based system, and then in different indexers. Then we combined classical vectorization with word embedding by integrating the best techniques. Second, we tested the new approach, which only uses BERT to take into account the context of words. In this step, we have compared different multilingual transformers and Arabic pre-trained models using BERT, then the best one with the previous approaches. To identify similar sentences, we need to compare some techniques, such as Euclidean distance and Cosine Similarity.

In the finalexperiment, which was performed to answer the efficiency objective of this study, we have compared the taken time of models developed using a keyword-based system and a semantic-based system.

5. Data Collection

5.1 Lexical database

Our Dictionaryis a collection from several Arabic books and websites that contain approximately 3,00,000 Arabic words covering various domains such as Religion, Literature, Economics, Law, and Politics. For the Religion domain, words are taken from the Holy Quran, the Hadith narratives uploaded onto the GitHub repository [10], and from Maktabah Shamilah, which contains hundreds of Arabic books in various Islamic domains.

This corpus is available in text form, providing various tags (Figure 1): word, lemma, pos, root, pattern, and affix.

suffix	pattern	root	prefix	pos	lemma	word
	أَفْعَل	حمد		مصرفة منتظمة	أَحْمَد	أَحْمَدُ
يّ	فَعْل	شوق		مصرفة منتظمة	شَوْقِيّ	شَوْقِيٌّ
	فَعِيل	أمر		مصرفة منتظمة	أَمِير	أَمِيرُ
	فُعَلَاء	شعر	ال	مصرفة منتظمة	شَاعِر	اَلشُّعَرَاءِ
	أَوَّل	أول		استثناء صرفي	أَوَّل	أَوَّلُ
	مَنْ	اَلَّذِي		جامدة	مَنْ	مَنْ
	اِفْتَعَلَ	بكر		مصرفة منتظمة	ابْتَكَرَ	اِبْتَكَرَ
	فِعْل	شعر	ال	مصرفة منتظمة	شِعْر	اَلشِّعْرَ
يّ	مَفْعَل	سرح	ال	مصرفة منتظمة	مَسْرَحِيّ	اَلْمَسْرَحِيَّ
	هُوَ	هُوَ	فَ	جامدة	هُوَ	فَهُوَ
ه	فَاعِل	رود		مصرفة منتظمة	رَائِد	رَائِدُهُ
	أَوَّل	أول	ال	استثناء صرفي	أَوَّل	اَلْأَوَّلُ

Fig 1: *Various tags from our corpus*

In the Lucene engine, Stemmer token filters handle stemming based on how they stem words: based on a set of rules or by looking them up in a dictionary. Stemmer token filter via ArabicStemmer Class and Snowball-based stemming rules are used for the Arabic language.Hunspell, the spellchecker's OpenOffice.org project, is a token filter used to perform dictionary stemming. An interesting property of the hunspell stemmer is that understanding the format will allow us to write our custom dictionaries. Lucene is provided with the Arabic spelling dictionary and thesaurus for spellchecker based on Hunspell. To use this dictionary, we have enriched it with our own by taking the four steps:

- Collecting data from Arabic books and scrap websites with Python language and Beautiful Soup library.
- Extracting words and tags using Lemmatizer algorithms such as Farasa and Qalsadi Lemmatizers.
- Installing Arabic dictionaries in Lucene and defining a Hunspell token filter that uses them.
- Adding our custom dictionaries using the Python language and the Hunspell package.

As a next step, we try different Lemmatizeres using Lucene to index our documents and to get suitable comparison values for the same collection and queries.

6. Datasets

To evaluate different techniques and Arabic word lemmatizing, we need a large dataset. Unfortunately, there are not many free available Arabic datasets, especially for information retrieval tasks. Indeed, datasets for search engine evaluation require documents, queries, and relevance judgments to address various tasks. In this paper, the experiments were carried out with two Arabic datasets: The first one is EveTAR test collection [14] is the first Arabic-language freely-available multiple information retrieval tasks including ad-hoc search. It contains a crawl of 355M Arabic tweets encoded in UTF-8, from which about 62K tweets were judged. The second dataset [36] concerns a benchmark designed for mono-lingual retrieval containing 2,106,586 tweets. One very interesting thing about these collections is that they contain documents, queries, and relevance judgments. This can be very helpful to evaluate search engine approaches using MAP metrics. Indeed, to evaluate system performance, we used MAP and precision at different ranks (P@n) averaged over all topics. We first preprocessed a dataset to be used for Arabic information retrieval purposes and other Arabic NLP tasks. Therefore, we introduce methods to clean each sentence in the corpus and the search queries, removing noise and many unnecessary words. After initializing the models, we converted the corpus and the search queries into sentence embeddings to compare semantic approaches and enhance the efficiency of the selected models. In this kind of database, the useful evaluation metric is MAP to rank each relevant document.

7. Evaluation Measures

As mentioned above, our retrieval system is evaluated by using average precision on a ranked list. Indeed, we need to evaluate the quality of a ranked list rather than whether a relevant document was returned anywhere in the results[35]. This is because our evaluation database contains binary relevance, where documents are judged as being relevant (rel) or non-relevant. First, we calculate the Precision at the rank k of the document (d) in the ranked list (L) returned for a given query (q):

$$L = \left(d\_1, \dots d\_k\right) \tag{1}$$

$$p(k) = \frac{|L \cap \text{rel}|}{k} \tag{2}$$

Then, If n is the length of the ranked list L, we can define average precision (ap) as [35]:

$$ap(L) = \frac{1}{|rel|} \sum_{i=1}^{n} p(i) \tag{3}$$

The value (ap) is calculated for just one query (q). However, to compare our results properly, we experiment with different queries. The arithmetic mean of ap for all these queries is called MAP and can be defined as:

$$MAP(\mathcal{L}) = \frac{1}{m} \sum_{i=1}^{m} ap(\mathcal{L}_i) \tag{4}$$

The value of MAP provides a succinct summary of the ranking technique's effectiveness [35]. It can be also used to evaluate a recommender system as we are treating it like a ranking task. Moreover, it is the most useful trec_eval ranking, a tool used to evaluate the rankings, of each relevant document.

8. Implementation and results

8.1 First experiment

To achieve the best retrieval performance for the Keyword-based retrieval stage, we need to identify the optimal form of the term to be indexed. The aim of Lemmatization and root stemming is to return a word to its root using rules or a dictionary. Thus, in the first experiment, we mapped a word to its roots using our dictionary through the Lucene indexer. The Elastic search options, based on the Java Lucene libraries, allow us to compare different text analyzers such as Snowball Stemmer and Arabic Stemmer. It is also provided with the Arabic spelling dictionary and thesaurus for spellchecker based on Hunspell. In this experiment, we enriched it with our dictionary to compare different lemmatizers and their impact on retrieval performance. We first gathered some relevant resources through the Shamila library and then scraped some websites using the Beautiful Soup library. Extraction words and tags are applied by Farasa and Qalsadi Lemmatizers. Secondly, we pre-processed the dataset, introducing methods to clean each sentence in the dataset and the search queries, removing noise and many unnecessary words. To have the same comparison, we also need to apply the same pre-processing steps to the documents and query.

After lemmatizing tasks, the following step is to evaluate different types of lemmatizers. MAP is an effective measure based on average precision (AP) that is used for evaluating retrieval performance. Table 2 represents the MAP measures for Arabic Lemmatizers of ISRI Stemmer, Qalsadi, Farasa, and Dictionary.

Table 2:*MAP measures for Arabic Lemmatizers*

Lemmatizers	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
ISRIStem-mer	0.8920	0.8931	0.8802	0.8752	0.8733	0.8690
Qalsadi	0.8924	0.8784	0.8683	0.8631	0.8599	0.8575
Farasa	0.8915	0.8867	0.8802	0.8705	0.8640	0.8586
Dictionary	0.8837	0.8830	0.8724	0.8572	0.8477	0.8395

Looking at the MAP measure from Table 2,we can see that both the FarasaLemmatizers and ISRIStemmer perform better than the Dictionary and QalsadiLemmatizers on most of the thresholds. However, the Dictionary performs slightly better than QalsadiLemmatizers on thresholds 10 and 20. This comes from the fact that our Lemmatizer uses not only our lexical database but also Qalsadi Dictionary. Indeed, the QalsadiLemmatizerswere implemented based on Hunspell stemmer and Elasticsearch allows us to add our custom dictionaries then just about 2000 new tags are added. The ISRIStemmer gives the best MAP score of 0.8931 on threshold 10 and it will be used in all next experiments.

8.2 Second experiment

For our second experiment, In Table 3, we have compared the vector-based and embedding-based methods to extract features from the text, mainly: BM25, TF-IDF, Fast text, word2vec, and doc2bow. To be able to compare the efficacy of each method, we first transformed the documents into vectors by using a suitable function according to the model. Next, we searched with a vector of the query in the transformed documents. In this experiment, we don't feed transformed documents into the indexer as we aim to compare the efficacy of each method. We also measured MAP for every method at different thresholds to compare our results properly.

Table 3:*Vector-based and embedding-based comparison*

Models	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
TF-IDF	0.8279	0.8272	0.8149	0.8019	0.7949	0.7885
BM25	0.8803	0.8850	0.8827	0.8797	0.8720	0.8574
Fast text	0.0000	0.0113	0.0133	0.0241	0.0269	0.0283
doc2bow	0.7346	0.7502	0.7583	0.7505	0.7468	0.7419
word2vec	0.7578	0.7724	0.7771	0.7907	0.7968	0.7947

From Table 3, we can observe that most of the models perform similarly except for Fast text. The highest MAP achieved by BM25 and TF-IDF, were 0.8850 and 0.8272, respectively on the top 10 returned documents. word2vec and doc2bow also achieved the second-best MAP at 0.7724 and 0.7502 respectively on the top 10 returned documents. The highest MAP was achieved by BM25 on most of the thresholds. Unfortunately, we noticed that the performance of Fast Text was the worst in terms of MAP. The reason is that BM25 prioritizes keywords in the query over other words, while Fast text adds other semantic terms using Query Expansion. Indeed, in our search engine, Fast text was used to add more words of the same meaning to the query for better retrieval in the BM25-based search engine. However, we believe that this can be solved using complicated queries in big corpus and datasets, which require enough computational resources. BM25 looks like a useful method when combined with more advanced text processing techniques such as lemmatization. However, when queries get complicated, the BM25 method fails.

8.3 Third experiment

Another experiment we implement using Arabic BERT and multilingual BERT will rank the relevance of the documents for a given query. Many pre-trained models for semantic textual similarity are available in the Sentence Trans-formers library. Our first selection was multilingual BERT models, and we decided to compare three different models: MiniLM-L12-v2 "paraphrase-multilingual-MiniLM-L12-v2", mpnet-v2 "paraphrase-multilingual-mpnet-base-v2", MiniLM-L6-cos-v1 "multi-qa-MiniLM-L6-cos-v1", mpnet-cos-v1 "multi-qa-mpnet-base-cos-v1", distiluse-v1 "distiluse-base-multilingual-cased-v1", distiluse-v2 "distiluse-base-multilingual-cased-v2" that for each version v1 and v2. Like in the first experiment, we converted the documents and the search queries into sentence embeddings by using a BERT model developed to generate sentence embeddings. Next, we searched in the converted documents with an embedding of the query. Table 4 represents the Multilingual BERT model.

Table 4:*Multilingual BERT models*

Models	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
MiniLM-L12-v2	0.6323	0.6001	0.5916	0.5781	0.5740
mpnet-v2	0.6585	0.6200	0.5975	0.5839	0.5821
MiniLM-L6-cos-v1	0.4568	0.4265	0.4043	0.3799	0.3541
Mpnet-cos-v1	0.4679	0.4391	0.4201	0.4143	0.4063
distiluse-v1	0.6205	0.5911	0.5688	0.5538	0.5498
distiluse-v2	0.6313	0.5938	0.5706	0.5566	0.5500

It can be observed from Table 4 that paraphrase-multilingual and distiluse-base-multilingual pre-trained word embedding models perform similarly. However, the highest MAP was achieved by paraphrase-multilingual-mpnet-base-v2 on all of the thresholds. Therefore, we choose it as the overall best model for the next experiments. In addition, it appears to perform better when searching for Arabic-language documents in this study.

Our second selection was Arabic BERT models, and we decided to compare all available models: MARBERT "UBC-NLP/MARBERT", ARBERT "UBC-NLP/ARBERT", Arabic_BERT "asafaya/bert-base-arabic", QARiB "qarib/bert-base-qarib", AraBERT "aubmindlab/bert-base-arabertv02",CAMeLBERT"CAMeL-Lab/bert-base-camelbert".

Similarly, on the Arabic BERT, it is shown in Table 5 that QARiB had the best MAP on the top 5 and 10 returned documents. AraBERT performed similarly to QARiB and achieved the best MAP on the top 20 returned documents.

Table 5:*Arabic BERT models*

Models	mAP@5	mAP@10	mAP@20
AraBERT	0.67583	0.645661	0.612475
MARBERT	0.58422	0.557474	0.521709
ARBERT	0.63600	0.60611	0.579104
Arabic_BERT	0.601556	0.576883	0.552904
QARiB	0.687252	0.649357	0.601693
CAMeLBERT	0.667197	0.637373	0.609008

As explained above, the MAP was used as a judgment metric as it is more optimal for binary data sets. To get better results we then used the selected models, para-phrase-multilingual-MiniLM-L12 and QARiB, and feed them into various indexers to compare the efficiency of indexers and methods selected in previous experiments.

8.4 Fourth experiment

Since the word embeddings don't seem to be working, such as the BM25 method in the second experiment, we decided to implement a re-ranking on the corpus using models selected in the previous experiment. First, we ran a standard BM25-based method on the same corpus and rescore the top 200 results using the Lucene indexer for multilingual BERT. Therefore, we implemented functions in the Lucene indexer to return three kinds of results: standard BM25-based results, dense-vector results, and re-ranked results. This task is performed using multilingual BERT. Indeed, Lucene provides an addition to standard BM25 search, techniques to index embeddings and score based on dense-vector, and then to compare documents. In addition, a dense vector saves the text embeddings and uses different metrics such as cosine-similarity and Euclidean distance to calculate the similarity between the query vector and the indexed content Vector. Finally, the rescoring function is used in Lucene to achieve re-ranking on the multilingual BERT-based results.

In contrast, the second task is directly performed using Arabic BERT without any indexers. Like in the first task, we ran a standard BM25-based method, but in this case, the Rank-BM25 library was used to create our function to run and then to search using a keyword-based search engine. The same query was then entered into an embedding-based search engine, and the top 200 retrieved results are scored by a Cross-Encoder re-ranker, with the top 10 results being evaluated. We use Arabic BERT (QARiB) as input in Cross-Encoder and the results are shown in the table below. Table 6 covers the re-think result along with the type search for each indexer.

Table 6: *BM25-based, dense-vector, and re-ranked result*

Indexer	Type search	MAP@10
Lucene	match	0.883046
Rank-BM25	match	0.886862
Lucene	dense	0.876055
pytorh	dense	0.852426
Lucene	re-rank	0.876055
pytorh	re-rank	0.791110

Unfortunately, there was no outstanding difference between the two tasks for match-based research. In addition, BERT does not provide remarkable results that are relevant to the query, especially for re-ranking. We can temporarily explain this phenomenon by the lack of implementation of the same information retrieval methods in big datasets or fine-tuning BERT for our specific task.

9. Evaluation of similarity measurement

In all experiments, we compute the cosine similarity score for the query vector and each document, as two terms are similar if their vectors are similar. However, there are other commonly used distance metrics in information retrieval, of which the choice depends on the use case. We take advantage of the opportunity in this experiment to evaluate and compare different distance metrics for word embedding. Fortunately, as can be observed from Table 7, all metrics were performed similarly with a bit of an advantage for cosine similarity in MAP and time consumption.

Table 7:*Different distance metrics for words embedding*

Type of similarity	mAP@10	time taken
Cosine similarity	0.85243	0.04732
city block	0.84944	0.04093
Euclidean	0.84800	0.03915
sqeuclidean	0.84800	0.03965
correlation	0.85290	0.07923

From these experiments, we can combine the best approaches and decide on the final structure of our system. However, there remains an important element in the performance of a research system which is efficiency. Then the next step is to run different indexers and choose the speed that improves the efficiency of the search engine and then display results in real-time.

10. Improving search engine efficiency

For our fifth experiment, we performed a semantic search with cosine similarity in different indexers and we decided on a multilingual BERT model, which was selected in the experiment above.

To get comparable values, we evaluated the search results based on the same queries and sentence embeddings on the same GPU-device this for different indexers or libraries: Annoy, FAISS, hnswlib, Lucene, and directly in GPU device using PyTorch library.

To check this, we first measure the MAP for two thresholds (Table 8), which fortunately is the same with a bit of difference for the FAISS indexer as it is highly dimensional (784 dimensions). In contrast, other indexers use just 384 as the size of the embedding required by dataset content. Moreover, FAISS compares vectors to find the nearest matches using Euclidean or inner-product similarity metrics.

Table 8: *Cosine similarity in different indexers*

Indexers	faiss	hnswlib	annoy	Lucene	pytorch
MAP@5	0.835778	0.856751	0.856751	0.856085	0.856751
MAP@10	0.840412	0.852426	0.852531	0.852389	0.85353

The time taken for retrieval was calculated by averaging the time required to retrieve all queries. The results in Table 9 clearly show that all indexers are capable of rapid dynamic retrieval in a short time compared with the Lucene indexer. Indeed, Lucene compares the query embeddings with all embeddings stored and indexed so it is a slow indexer.

Table 9:*The time taken to retrieving*

Indexers	faiss	hnswlib	annoy	Lucene	cuda
time@5	0.006392	0.005711	0.005925	0.01132	0.00815
time@10	0.00597	0.005775	0.006208	0.01156	0.008188

However, we look for fast results and also for reliable and correct results whereas the results are not necessarily exact using ANN libraries: Annoy, FAISS, and hnswlib [33]. Therefore, this is not the time to decide which indexer to choose for our Arabic search engine.

Similarly, the table 10 shows results from BM25 for Lucene, whoosh, and pyterrier.

Table 10: *BM25 for Lucene, whoosh and pyterrier*

Indexers	MAP@5	MAP@10
Whoosh	0.225718	0.227479
Pyterrier	0.019888	0.01996
Lucene	0.004	0.00286

Whoosh is a full-text indexing and searching library implemented in Python. PyTerrier is also a Python framework that makes it easy to perform information retrieval experiments. Unlike in the first task, the results in Table 10 clearly show that the Lucene indexer is capable of rapid dynamic retrieval in a short time compared with Whooshand PyTerrier indexers.

11. Advantages and Disadvantages

Advantages

- Each sentence was cleaned and processed to remove the unwanted noise and unnecessary words.
- Higher MAP was obtained compared to the existing method.
- This method provides an accurate result of 89% with the use of the largest dataset

Disadvantages

- The main problem with experimenting with BERT representation is that it takes a long time to obtain scores.

12. Conclusions

In this paper, we conducted some experiments comparing information retrieval results on the Arabic dataset beginning with BM25 for stage retrieval to the reranking stage that rescores the list of candidate documents. The EveTAR dataset of 335M Arabic tweets was used to verify our experiment. First, wehave compared most of the existing lemmatizers in terms of IR performance. The results showed that the ISRIStemmer outperformed the other lemmatizers, giving the best MAP score of 89% on threshold 10. Second, we have compared keywords-based with semantics-based approaches for the purpose of Arabic IR. For feature extraction, the highest MAP was achieved by BM25 compared to the vector-based and embedding-based methods. In the second step, we started using multilingual BERT models and experimental results with Arabic information retrieval datasets indicate that the pre-trained paraphrase-multilingual-mpnet-base-v2 model is the most suitable for such tasks. For Arabic BERT models, pre-trained for Arabic on a large Arabic corpus, AraBERT performed similarly to QARiB and achieved the best MAP on the top 20 returned documents.

Among these different approaches, the multilingual BERT-based search engine combined with the standard BM25-based method had the highest accuracy when the retrieved results were combined with the score-based ranking for the search engine. We also explored using monolingual BERT, which did not significantly provide remarkable results, especially for re-ranking. In the last experiment, we compared the taken time of models developed using the combined system, and experimental results indicate that the Lucene indexer is capable of rapid dynamic retrieval in a short time compared with other indexers. The results can be further improved with the use of a domain-specific task, which will be our future work.

Compliance With Ethical Standards

Conflicts of interest: Authors declared that they have no conflict of interest.

Human participants: The conducted research follows ethical standards and the authors ensuredthat they have not conducted any studies with human participants or animals.

References

[1] Abdelali, A., Hassan, S., Mubarak, H., Darwish, K., &Samih, Y. "Pre-training BERT on Arabic tweets: Practical considerations",arXiv preprint arXiv:2102.10684, 2021.

[2] Abdul-Mageed, M., Elmadany, A., &Nagoudi, E. M. B., "ARBERT & MARBERT: deep bidirectional transformers for Arabic", arXiv preprint arXiv:2101.01785, 2020.

[3] Abu Bakr Soliman, Kareem Eisa, and Samhaa R. El-Beltagy, "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP", in proceedings of the 3rd International Conference on Arabic Computational Linguistics (ACLing 2017) vol. 117, pp. 256-265, Dubai, UAE, 2017.

[4] Abu-Salih, Bilal.,"Applying Vector Space Model (VSM) Techniques in Information Retrieval for Arabic Language", 2018.

[5] Abuzayed, A., & Al-Khalifa, H.,"BERT for Arabic Topic Modeling: An Experimental Study on BERTopic Technique", Procedia Computer Science, Vol. 189, pp. 191-194, 2021.

[6] Bhoir S., T. Ghorpade and V. Mane, "Comparative analysis of different word embedding models", International Conference on Advances in Computing, Communication and Control (ICAC3), pp. 1-4, 2017.

[7] Bojanowski, P., Grave, E., Joulin, A., &Mikolov, T.,"Enriching word vectors with subword information. Transactions of the association for computational linguistics", Vol. 5, pp. 135-146, 2017

[8] Camacho-Collados, J., &Pilehvar, M. T.,"On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis", arXiv preprint arXiv:1707.01780, 2017.

[9] Devlin, J., Chang, M. W., Lee, K., &Toutanova, K.,"Bert: Pre-training of deep bidirectional transformers for language understanding", arXiv preprint arXiv:1810.04805, 2018.

[10] Elmaz, O.,"An Explorative Journey Through Hadith Collections: Connecting Early Islamic Arabia with the World", Journal of Arabic and Islamic Studies, Vol. 21, pp. 39-56, 2021.

[11] Farghaly A. and K. Shaalan, "Arabic natural language processing: Challenges and solutions," ACM Trans. Asian Lang. Inf. Process., Vol. 8, no. 4, pp. 14, 2009.

[12] Frenz, C. M.,"Introduction to Searching with Regular Expressions", arXiv preprint arXiv:0810.1732, 2018.

[13] Fukushima, K., & Miyake, S.,"Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position", Pattern recognition, Vol. 15, no. 6, pp. 455-469, 1982.

[14] Hasanain, M., Suwaileh, R., Elsayed, T., Kutlu, M., Almerekhi, H., "EveTAR: Building a Large-Scale Multi-Task Test Collection over Arabic Tweets", Information Retrieval Journal (2017).

[15] He B. and I. Ounis, "Term frequency normalisation tuning for BM25 and DFR models," in European Conference on Information Retrieval, pp. 200-214, 2005.

[16] Heikal, M., Torki, M., & El-Makky, N.,"Sentiment Analysis of Arabic Tweets using Deep Learning", Procedia Computer Science, Vol. 142, pp. 114–122, 2018.

[17] Hu, J., Wang, G., Lochovsky, F., Sun, J. T., & Chen, Z.,"Understanding user's query intent with Wikipedia", In Proceedings of the 18th international conference on World wide web, pp. 471-480, 2009.

[18] Inoue, G., Alhafni, B., Baimukan, N., Bouamor, H., &Habash, N.,"The interplay of variant, size, and task type in Arabic pre-trained language models", arXiv preprint arXiv:2103.06678, 2021.

[19]Ismail S. and M. S. Rahman., "Bangla word clustering based on ngram language model," in Electrical Engineering and Information & Communication Technology (ICEEICT), International Conference on. IEEE, pp. 1–5, 2014.

[20] Joho H. and M. Sanderson, "Document frequency and term specificity", In the Recherche d'InformationAssiste par Ordinateur Conference (RIAO), 2007.

[21] Kadhim, A. I.,"Term weighting for feature extraction on Twitter: A comparison between BM25 and TF-IDF", In 2019 international conference on advanced science and engineering (ICOASE) (pp. 124-128), IEEE, 2019.

[22] Kassimi, M. A., & El Beqqali, O.,"3D model classification and retrieval based on semantic and ontology", International Journal of Computer Science Issues (IJCSI), Vol. 8, no. 5, pp. 108, 2014.

[23] Lashkari, A. H., Mahdavi, F., &Ghomi, V.,"A boolean model in information retrieval for search engines", In 2009 International Conference on Information Management and Engineering (pp. 385-389), IEEE.

[24] Metzler Jr, D. A.,"Beyond bags of words: Effectively modeling dependence and features in information retrieval", University of Massachusetts Amherst, 2007.

[25] Mikolov T, Chen K, Corrado G, Dean J, "Efficient Estimation of Word Representations in Vector Space", In Proceedings of Workshop at ICLR, arXiv; pp. 1301-3781, 2013.

[26] Ni, J., Ábrego, G. H., Constant, N., Ma, J., Hall, K. B., Cer, D., & Yang, Y., "Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models", arXiv preprint arXiv:2108.08877, 2021.

[27] Onyenwe, I., Ogbonna, S., Onyedimma, E., Ikechukwu-Onyenwe, O., &Nwafor, C.,"Developing Smart Web-Search Using RegEx", arXiv preprint arXiv:2110.04767, 2021.

[28] Pennington J, Socher R, Manning CD, "Glove: Global Vectors for Word Representation", In EMNLP, 2014; vol. 14, pp. 1532-1543.

[29] Peters,M. E., Neumann,M., Iyyer,M., Gardner,M., Clark, C., Lee, K., and Zettlemoyer, L.,"Deep contextualized word representations", In Proceedings of NAACLHLT, pp. 2227–2237, 2018.
[30] Platzer, C., &Dustdar, S.,"A vector space search engine for web services", In Third European Conference on Web Services (ECOWS'05) (pp. 9-pp), IEEE, 2005.
[31] Prabhakar Raghavan Christopher D. Manning and Hinrich Schtze,"Introduction to information retrieval", Cambridge University Press, 2008.
[32] Radford, A., Narasimhan, K., Salimans, T., &Sutskever, I.,"Improving language understanding by generative pre-training",2018.
[33] Reimers, N., &Gurevych, I.,"Making monolingual sentence embeddings multilingual using knowledge distillation", arXivprepri, 2020
[34] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... &Polosukhin, I.,"Attention is all you need. Advances in neural information processing systems", Vol. 30, 2017.
[35] Zhai, C., Massung, S.,"Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining", ACM and Morgan & Claypool (2016).
[36] Zhang, X., Ma, X., Shi, P., & Lin, J.,"Mr. TyDi: A multi-lingual benchmark for dense retrieval", arXiv preprint arXiv:2108.08787, 2021
[37] Alghanmi, I., Espinosa-Anke, L., &Schockaert, S., "Combining BERT with static word embeddings for categorizing social media", 2020.
[38] Antoun, W., Baly, F., & Hajj, H., "Arabert: Transformer-based model for arabic language understanding", arXiv preprint arXiv:2003.00104, 2020