Mono-Lingual Search Engine: Combining Keywords with Context for Semantic Search Engine

Moulay Abdellah Kassimi, Harif Abdellatif, and Abdessalam Essayad

Abstract The success of pre-trained models relies on their capacity to understand human language and the success of word embeddings and transformers that allow access to large pre-trained NLP models. However, finding suitable architectures and techniques for monolingual semantic search is a challenge as this research focuses deeply on English. Our approach focuses on combining keyword-based matching and semantic search using different search methods to increase precision and recall. In this paper, we present an analysis of the different approaches adopted by our semantic search engine and the comparison between them. Our experiment on datasets from different domains has shown the efficiency and effectiveness of the adopted method. Indeed, our method achieved an accuracy of 0.89 which improves by up to 4% when combining keywords with semantics in search engines.

Keywords Natural language processing · Context · Semantics · Search engine · Embeddings transformers · Deep learning

1 Introduction

Information Retrieval problems have been widely studied in many real applications over the last few decades. Especially, with recent advances in Natural Language Processing (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 is an artificial intelligence technology that helps computers understand human language and extract relevant information. NLP gives computers the ability to understand human language [1]. Recently, pre-trained models and transformers have revolutionized the field of

M. A. Kassimi (✉) · H. Abdellatif · A. Essayad
National School of Applied Sciences, Ibn Zohr University, Agadir, Morocco
e-mail: m.kassimi@uiz.ac.ma

A. Essayad
e-mail: a.essayad@edu.uiz.ac.ma

N. Gherabi et al. (eds.), *Advances in Intelligent System and Smart Technologies*,
Lecture Notes in Networks and Systems 826,
https://doi.org/10.1007/978-3-031-47672-3_34

NLP. The pre-trained models such as BERT (Bidirectional Encoder Representations from Transformers) [2] and GPT (Generative Pre-trained Transformer) [3], 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 [4]. Therefore, Transformers can consider the context of a word and then use it to understand the intent behind the search query. Applying BERT models to the Arabic language remains a challenging task with complicated orthography and morphology compared to other languages. Particularly, in the search process users have abandoned these diacritics, and search engine systems are expected to understand the lost diacritics [5]. 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. This paper focuses on combining keyword-based matching and semantic search using different search methods to increase precision and recall. In fact, the goal is to perform lexical and semantic searches taking advantage of the Elasticsearch engine and the contextual embeddings to index our Arabic corpus. Therefore, to extend the power of great models like BERT to the Arabic language.

2 Related Works

Search Engine is the practical application of information retrieval techniques in which the main objective is to retrieve a relevant document from a corpus using an appropriate model such as the Boolean Model [6] or Vector Space Model [7]. The Boolean model creates queries in a Boolean expression terms structure which are combined with the operators AND, OR, and NOT [8]. 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 queries in the vector-space model are represented by a vector of index terms with related weights. These weights represent the importance of the index [9]. Keyword-based search engines used to be built around BM25 and TF-IDF-based vector space models as are easy to use and are effective if the researcher knows exactly what they are looking for. BM25, for Best Matching, is a ranking function used by search engines to rank matching documents according to their relevance to a given search query [10]. While TF-IDF, for term frequency-inverse document frequency, gives more weight to words that are unique to a document than to words that are frequent. In [11] 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. Authors in [12] developed a system that has 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 queries as bags of words (BOW), a statistical model for information retrieval based on Markov random fields [13]. The such 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 are still in use and are the simplest IR System but 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 [14]. The number of ontology-based semantic search engines has been continuously growing in the last decade. In our previous work [15], 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 are not sufficient and do not help much in semantic information retrieval, especially for searching text.

Recently, to tackle this problem, many word embedding techniques have been proposed, nevertheless, Word2Vec [16], Glove [17] and Fasttext [18] are the most popular semantic embedding techniques. The authors in [19], present a comparison between Word2Vec and Glove. Word2Vec predicts words based on their context by using one of two distinct neural models CBOW (Continuous bag of words) and Skip-gram. CBOW uses embeddings to train a neural network where the context is represented by multiple words for given target words [20]. Skip-gram is used to predict the context word for a given target word. As discussed in [21], the performance of each combination is very close for the Arabic language but word2vec gives the best vector representation on its own. The N-gram model was used in the search engine to build an auto-completion program or search suggestions. It is important for many Natural Language Processing applications, and it is used to predict a word based on its previous and next words sequence in the works. Ismail and Rahman [21] construct the clusters using semantic and contextual similarity. The same cluster contains words that have a similar meaning and are used in a similar context in a document. WThe ord2vec model was followed by a Fasttext model [18], 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. Fasttext model is a relatively new model in which the library contains hundreds of millions of words that have been trained on Wikipedia and Common Crawl data. It is strong [20] and shows better performance than word2vec and the best is that encourages the Arabic language. AraVec [22] embeddings, also pre-trained Arabic embeddings are word2vec vectors trained on Tweets and Wikipedia Arabic articles and have wide coverage of dialect words. AraVec pre-trained word embeddings were used by Heikal et al. [23] 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 Fasttext aimed at representing words by capturing their semantic properties. Unfortunately, Words have different meanings in different contexts and these models did not incorporate the context. To address this issue, new models generate contextualized representations like in Peters et al. [24]. Authors used ELMO to create vectors derived from

a bidirectional LSTM that is trained on a large text corpus. The new approaches are 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 the search query. The performance on different tasks improved, leading to larger structures that had sentence representations such as BERT [2] and T5 [25], which offered improved performance by exploring different pre-training techniques and huge training data. BERT is 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 has been a focus on fine-tuning BERT with small data.

For Arabic language, Google produced a multilingual BERT [2] 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 [26–28]. Unfortunately, these models are tested for Question Answering applications. Moreover, the best results are currently achieved by fine-tuning AraBERT [26]. In [20] the authors emphasize that we can get better results by combining AraBERT and AraVec. Recently, Many Arabic models have been pre-trained such as MARBERT [29], CAMeLBERT [28], and QARiB [30]. Therefore, we use these and other pre-trained models to compare their abilities to create a semantic search engine for the Arabic language. Thus, this paper aimed to investigate 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.

3 Data Collection

3.1 Datasets

To evaluate different techniques, we need a large dataset. Unfortunately, there are not many Arabic datasets freely available, 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 [29] which is the first Arabic freely-available multiple information retrieval tasks including ad-hoc search. It contains a crawl of 355M Arabic tweets encoded in UTF-8 which about 62K tweets were judged. The second dataset [31] concerns a benchmark designed for mono-lingual retrieval containing 2,106,586 tweets. One very interesting thing about these collections is to contain documents, queries, and relevant judgments. This can be very helpful to evaluate search engine approaches using mean average precision (MAP) metric. Indeed, to evaluate system performance we used MAP and precision at a different rank (P@n) averaged over all topics. We first preprocessed the dataset

to be used for Arabic information retrieval purposes and other Arabic NLP tasks. Therefore, we introduce methods for cleaning 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 (Mean Average Precision) to rank each relevant document.

4 Implementation and Results

4.1 Metrics

In this paper, 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 [32]. 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 rank k of the document (d) in ranked list (L) returned for a given query (q):

$$\begin{aligned} L &= (d_1 \ldots d_k) \\ \mathrm{p(k)} &= |\mathrm{L} \cap \mathrm{rel}|/\mathrm{k} \end{aligned} \tag{1}$$

Then, and If n is a length of ranked list *L,* we can define **average precision (ap)** as [32]:

$$ap(L) = \frac{1}{|rel|} \sum_{i=1}^{n} p(i) \tag{2}$$

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

$$MAP(\mathcal{L}) = \frac{1}{m} \sum_{i=1}^{m} ap(\mathcal{L}_i) \tag{3}$$

The value of MAP provides a succinct summary of the ranking technique effectiveness [32]. 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 of each relevant document.

Table 1 Comparing the vector-based end embedding-based methods

Models	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50	mAP@60
tfidf	0.827	0.827	0.814	0.8019	0.7949	0.7885	0.7817
bm25	0.880	0.885	0.882	0.8797	0.8720	0.8574	0.8517
Fasttext	0.000	0.011	0.013	0.0241	0.0269	0.0283	0.0283
doc2bow	0.734	0.750	0.758	0.7505	0.7468	0.7419	0.7354
word2vec	0.757	0.772	0.777	0.7907	0.7968	0.7947	0.7934

4.2 First Experiment

For our first experiment we compared the vector-based end embedding-based methods for extracting features from the text, mainly: BM25, TF-IDF, Fasttext, word2vec, doc2bow. To be able to compare the efficacity of each method, we first transformed the documents into vectors by using a suitable function according to model. Next, we searched with a vector of the query in the transformed documents. In this experiment, we don't feed transformed documents into indexer as we aim at comparing efficacity of each method. We also measured MAP for every method on different thresholds to compare our results properly (Table 1).

It can be observed from Table 2 that most of the models perform in a similar manner except for Fasttext. The highest MAP achieved by BM25 and TF-IDF were 0.8850 and 0.8272 respectively on the top 10 returned documents. word2vec, 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 notice that the performance of Fasttext was the worst in terms of MAP. This could mainly due to the fact that it is BM25 prioritizes keywords in the query over other words, while Fasttext adds other semantic terms using Query Expansion. Indeed, in our search engine, Fasttext 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 corpora and datasets that require enough computational resources.

4.3 Second Experiment

For our second experiment, we try to implement Arabic BERT and multilingual BERT that will rank the relevance of the documents with respect to a given query. Many pre-trained models for Semantic Textual Similarity are available in the Sentence Transformers library. Our first selection was multilingual BERT models, and we decided to compare three different models: paraphrase-multilingual-MiniLM-L12, multi-qa-mpnet-base-cos, and distills-base-multilingual-cased 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

Table 2 Comparing different BERT models

Models	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50	mAP@60
Paraphrase-multilingual-MiniLM-L12-v2	0.6863	0.6323	0.6001	0.5916	0.5781	0.5740	0.5671
Paraphrase-multilingual-mpnet-base-v2	0.7108	0.6585	0.6200	0.5975	0.5839	0.5821	0.5782
Multi-qa-MiniLM-L6-cos-v1	0.4843	0.4568	0.4265	0.4043	0.3799	0.3541	0.3421
Multi-qa-mpnet-base-cos-v1	0.4838	0.4679	0.4391	0.4201	0.4143	0.4063	0.3959
Distiluse-base-multilingual-cased-v1	0.6576	0.6205	0.5911	0.5688	0.5538	0.5498	0.5424
Distiluse-base-multilingual-cased-v2	0.6706	0.6313	0.5938	0.5706	0.5566	0.5500	0.5400

Table 3 Comparing different Arabic BERT models

Models	mAP@5	mAP@10	mAP@20
AraBERT	0.675833	0.645661	0.612475
MARBERT	0.584222	0.557474	0.521709
ARBERT	0.636001	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

Table 4 Retrieved results scored by a cross-encoder re-ranker

Indexer	Type_search	MAP@10
Elasticsearch	Match	0.883046
Rank-BM25	Match	0.886862
Elasticsearch	Dense	0.876055
pytorh	Dense	0.852426
Elasticsearch	Re-rank	0.876055
pytorh	Re-rank	0.791110

embeddings. Next, we searched in the converted documents with an embedding of the query.

It can be observed from Table 3 that paraphrase-multilingual and distills-base-multilingual pre-trained word-embedding models perform in a similar manner. 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 Arabic-language documents in this study.

Our second selection was Arabic BERT models, and we decided to compare all available models.

Similarly, on the Arabic BERT, it is shown in Table 4 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.

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, paraphrase-multilingual-MiniLM-L12 and QARiB, and feed them into various indexers to compare the efficiency of indexers and methods selected in previous experiments.

4.4 Third 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 run a standard BM25-based method on the same corpus and rescore the top 200 results using Elasticsearch indexer for multilingual BERT. Therefore, we implemented functions in Elasticsearch indexer to return three kinds of results: standard BM25-based result, dense-vector result, and re-ranked result. This task is performed using multilingual BERT. Indeed, Elasticsearch provides, in addition to standard BM25 search, techniques to index embeddings and scoring based on dense vectors, then to compare documents. In addition, a dense vector saves the text embeddings and uses different metrics such as cosine-similarity and Euclidean distance then to calculate the similarity between the query vector and the indexed content Vector. Finally, Rescoring function is used in Elasticsearch 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 run a standard BM25-based method, but in this case Rank-BM25 library was used to create our own function to run and then to search using the keyword-based search engine. Next, the same query was entered into the embedding-based search engine and the top 200 retrieved results are scored by a Cross-Encoder re-ranker and the 10 results with the highest score are evaluated. We use Arabic BERT (QARiB) as input in Cross-Encoder and the results are shown in the Table 4.

Unfortunately, no outstanding difference between the two tasks for match-based research. In addition, BERT (monolingual and multilingual) 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 finetune BERT for our specific task.

5 Conclusion

In this paper, we performed a comparison between different approaches used in Information Retrieval such as Bags of words (BOW), TF-IDF, Word embedding, and BERT-based search engines to find suitable mono-lingual architectures and techniques for semantic search engines. EveTAR data sets of 335M Arabic tweets were used to verify our experiment. Among these different approaches, the embedding-based search engine with standard BM25-based methods had the highest accuracy when the retrieved results were scored by a Cross-Encoder re-ranker. We also explored using monolingual BERT, which did not significantly provide remarkable results, especially for re-ranking. The results can be further improved with the use of a domain-specific task, which will be our future work.

References

1. Fukushima, K., Miyake, S.: Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position. Pattern Recogn.Recogn. **15**(6), 455–469 (1982)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding (2018). arXiv:1810.04805
3. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
4. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Polosukhin, I.,, et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
5. Farghaly, A., Shaalan, K.: Arabic natural language processing: challenges and solutions. ACM Trans. Asian Lang. Inf. Process. **8**(4), 14 (2009)
6. 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 (2009)
7. 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)
8. Raghavan, P., Manning, C.D., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
9. Joho, H., Sanderson, M.: Document frequency and term specificity. In: The Recherche d'Information Assiste par Ordinateur Conference (RIAO) (2007)
10. He, B., Ounis, I.: Term frequency normalisation tuning for BM25 and DFR models. In: European Conference on Information Retrieval, pp. 200–214 (2005)
11. 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)
12. Abu-Salih, B.: Applying Vector Space Model (VSM) Techniques in Information Retrieval for Arabic Language (2018)
13. Metzler Jr, D.A.: Beyond bags of words: effectively modeling dependence and features in information retrieval. University of Massachusetts Amherst (2007)
14. 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)
15. Kassimi, M.A., El Beqqali, O.: 3D model classification and retrieval based on semantic and ontology. Int. J. Comput. Sci. Issues (IJCSI) **8**(5), 108 (2011)
16. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of Workshop at ICLR, pp. 1301–3781 (2013)
17. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
18. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017)
19. Bhoir, S., Ghorpade, T., Mane, V.: Comparative analysis of different word embedding models. In: 2017 International Conference on Advances in Computing, Communication and Control (ICAC3), pp. 1–4 (2017). https://doi.org/10.1109/ICAC3.2017.8318770
20. Alghanmi, I., Espinosa-Anke, L., Schockaert, S.: Combining BERT with static word embeddings for categorizing social media (2020)
21. Ismail, S., Rahman, M.S.: Bangla word clustering based on N-gram language model. In: 2014 International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT), pp. 1–5. IEEE (2014)
22. Soliman, A.B., Eisa, K., El-Beltagy, S.R.: 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), Dubai, UAE (2017)
23. Heikal, M., Torki, M., El-Makky, N.: Sentiment analysis of Arabic tweets using deep learning. Proc. Comput. Sci. **142**, 114–122 (2018). https://doi.org/10.1016/j.procs.2018.10.466

24. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Proceedings of NAACLHLT, pp. 2227–2237 (2018)
25. 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 (2021). arXiv:2108.08877
26. Antoun, W., Baly, F., Hajj, H.: Arabert: transformer-based model for Arabic language understanding (2020). arXiv:2003.00104
27. Abdelali, A., Hassan, S., Mubarak, H., Darwish, K., Samih, Y.: Pre-training Bert on Arabic tweets: practical considerations (2021). arXiv:2102.10684
28. Inoue, G., Alhafni, B., Baimukan, N., Bouamor, H., Habash, N.: The interplay of variant, size, and task type in Arabic pre-trained language models (2021). arXiv:2103.06678
29. Hasanain, M., Suwaileh, R., Elsayed, T., Kutlu, M., Almerekhi, H.: EveTAR: building a large-scale multi-task test collection over Arabic tweets. Inf. Retrieval J. (2017). https://doi.org/10.1007/s10791-017-9325-7
30. Abuzayed, A., Al-Khalifa, H.: BERT for Arabic topic modeling: an experimental study on BERTopic technique. Proc. Comput. Sci. **189**, 191–194 (2021)
31. Zhang, X., Ma, X., Shi, P., Lin, J.: Mr. TyDi: a multi-lingual benchmark for dense retrieval (2021). arXiv:2108.08787
32. Zhai, C., Massung, S.: Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining. ACM and Morgan & Claypool (2016)