DEVELOPMENT AND EXPERIMENT: BM25 + BERTOPIC

1. Introduction and research problem

We started from the vocabulary mismatch problem ("*vocabulary mismatch*") in which users use different query terms from those used in relevant documents. This is one of the central challenges in information retrieval (*Information Retrieval*) (Nogueira *et* al, 2019, p.1).

Our goal is to improve the information retrieval (IR) of a search engine for a specific situation. We start from the hypothesis that, if a user is interested in a specific document from a group, he is also interested in returning other documents from the same group (Moura, 2009, p.17 *apud* Chakrabarti, 2003; Kobayashi and Aono, 2004).

Our proposal was to use a document expansion technique with terms that are representative of the content of the documents, composing a new field in the search system with the extracted terms. We created this field by modeling topics from our documents enhanced by pre-trained embeddings (BERT) models.

For this, we created an experimental environment that uses Elasticsearch for information retrieval. Elasticsearch is based on the BM25 algorithm (a classic TF-IDF-based IR algorithm) (Beiske, 2013). In this sense, we get a *framework* that is composed of (BERT + topic template) + BM25.

2. Methodology

From this experimental environment, we can compare the IR results between:

(a) Our baseline, composed of the original documents;

(b) Documents enriched with terms extracted from a topic model, composing a field in the search system with the new terms;

We opted for a variation of the topic model that also incorporates pre-trained embedding models based on BERT, in order to obtain topics with better semantic content. For this, the Python library called BERTopic (Grootendorst, 2022) and the pre-trained model "*distiluse-base-multilingual-cased-v1*" are used.

The entropy metric will be used to compare the results of (a) and (b).

2.1. Ranking Function - BM25

We need to find the relevance of query terms and documents to constitute a search engine (RI). “Word count” is not a sufficient metric to find the relevance of terms. Generally, the terms that appear the most are irrelevant, such as conjunctions (a, o, da, do etc) (Jimenez *et* al, 2018, p.2888). One way around this and finding the most relevant terms is through algorithms like TF-IDF (*term frequency–inverse document frequency*). Thus, we were able to filter the terms that appear a lot in all documents, because we understand that they are irrelevant; as a consequence, we give greater weight to the terms that appear a lot in some documents and we can see their distribution throughout the corpus of documents.

$$\text{TF-IDF} = \ tf(w,d)\ x\ idf(w,D)$$

While TF-IDF as a vector model favors the frequency of terms and penalizes the frequency of documents, it also disregards the size of the documents and the saturation of the frequency of the terms (Seitz, 2022). Another model, called BM25, is a proposal to solve this limitation using a probabilistic model. Thus, while the term frequency is controlled by a saturation function to prevent its linear growth, the parameter used to calculate the average size of documents penalizes long documents with a high frequency of search terms (Jimenez *et* al, 2018, p. 2888).

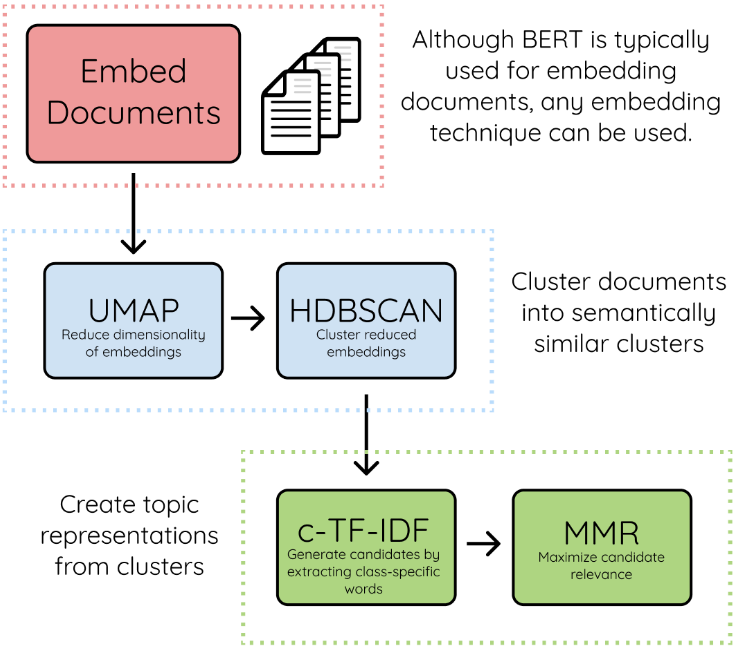
$$\text{BM25 :} \qquad \sum_{w \epsilon Q} idf(w)\ x\ \frac{(k1+1)\ x\ tf(w,d)}{k1\ x\ K(d)+tf(w,d)}\ x\ \frac{(k3+1)\ x\ tf(w,q)}{k3+tf\ (w,q)}$$

BM25 is the most used algorithm in search engines due to its *tradeoff* of accuracy and computational performance, incorporated in systems based on Lucene (such as Elasticsearch and Solr).

2.2. Topic Generation

Our baseline (a) uses the original documents without semantic enrichment. To enrich them, we generate topics (set of hierarchical words) that will compose a new field in the search system with the terms extracted from the topics.

For this, we use the *BERTopic* library framework (Grootendorst, 2022) to create our topic model. It goes through a few steps to produce our topics.


Embed Documents
Although BERT is typically used for embedding documents, any embedding technique can be used.
UMAP
Reduce dimensionality of embeddings
HDBSCAN
Cluster reduced embeddings
Cluster documents into semantically similar clusters
Create topic representations from clusters
c-TF-IDF
Generate candidates by extracting class-specific words
MMR
Maximize candidate relevance

Source: (Grootendorst, 2022)

At first, we took advantage of a *embeddings* to project our documents in this vector space and extract their vectors using the *sentence-bert*. The pre-trained model is used to improve the quality of our vectors than would be the case if we only used our document set to constitute our vector space (Nogueira *et al*, 2019).

In a second moment, the UMAP algorithm is applied to reduce the dimensionality of our vector space. Then, the HDBSCAN algorithm is applied to cluster our documents (Grootendorst, 2022).

Finally, we apply the CTF-ICF formula (a variation of the TF-IDF formula) to the clusters developed in the previous phase. Thus, each cluster is converted into a single document rather than a set of documents. From each cluster, we extract the frequency of the word x in cluster c, where c refers to the cluster we created earlier. This results in our cluster-based TF representation. As in the classic TF-IDF, we multiply TF by IDF to obtain the per-word importance score in each class (Grootendorst, 2022).

$$W_{x,c} = \mathbf{tf}_{x,c} \times \log\left(1 + \frac{A}{f_x}\right)$$

c-TF-ICF

Term x within class c

$\mathbf{tf}_{x,c}$ = frequency of word **x** in class **c**

$\mathbf{f}_x$ = frequency of word **x** across all classes

A = average number of words per class

Source: (Grootendorst, 2022)

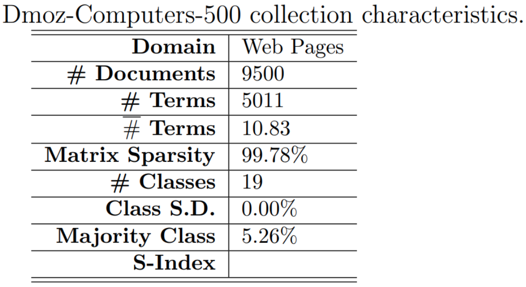
The TF-IDF is used to compare the importance of words across all documents in our corpus. Instead, we only deal with the documents in each cluster and then apply the TF-IDF respectively to each cluster as if it were a document (Grootendorst, 2022). The result would be the measure of importance for words within a cluster. The more important words that are within a cluster, the more representative it is for that topic. In other words, if we extract the most important words by cluster, we get topic descriptions. This model is called cluster-based TF-IDF (CTF-ICF) (Grootendorst, 2022).

Once we have our topic model, we can infer which topic is most important for each document. Thus, we expand our documents, taking advantage of the most important terms of their topic relative to the document, to compose a new field in the search system that can improve our information retrieval.

3. Assessment Criteria

3.1. Datasets

In 2013, the Institute of Mathematics and Computer Sciences of the University of São Paulo (ICMC-USP) made available sets of documents for the evaluation of computational experiments (Rossi, Marcacini, Rezende, 2013). We chose the computing dataset extracted from the *Open Directory Project* (*Dmoz-Computers-500 Collection*) (Netscape *apud* Rossi, Marcacini, Rezende, 2013). Its distribution has the following characteristics:

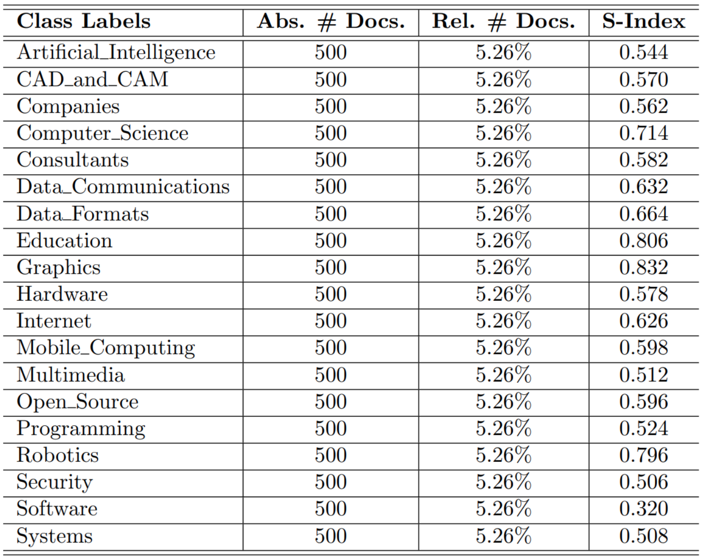


Dmoz-Computers-500 collection characteristics.

Domain	Web Pages
# Documents	9500
# Terms	5011
$\overline{\#}$ Terms	10.83
Matrix Sparsity	99.78%
# Classes	19
Class S.D.	0.00%
Majority Class	5.26%
S-Index	

Source: (Rossi, Marcacini, Rezende, 2013)

Each document has its classification. 500 documents were chosen, stratified into 19 categories. It is these ratings that will be used to compute the quality of our information retrieval.



Class Labels	Abs. # Docs.	Rel. # Docs.	S-Index
Artificial_Intelligence	500	5.26%	0.544
CAD_and_CAM	500	5.26%	0.570
Companies	500	5.26%	0.562
Computer_Science	500	5.26%	0.714
Consultants	500	5.26%	0.582
Data_Communications	500	5.26%	0.632
Data_Formats	500	5.26%	0.664
Education	500	5.26%	0.806
Graphics	500	5.26%	0.832
Hardware	500	5.26%	0.578
Internet	500	5.26%	0.626
Mobile_Computing	500	5.26%	0.598
Multimedia	500	5.26%	0.512
Open_Source	500	5.26%	0.596
Programming	500	5.26%	0.524
Robotics	500	5.26%	0.796
Security	500	5.26%	0.506
Software	500	5.26%	0.320
Systems	500	5.26%	0.508

Source: (Rossi, Marcacini, Rezende, 2013)

3.2. Metrics

In information retrieval applications it is necessary to have a reliable measure to test the results achieved. Entropy (uncertainty) provides a valuable measure in the verification of information systems, which is calculated through the set of the K highest ranked documents for a query (Grivola *et* al., 2005). Good retrieval results are expected to provide a more homogeneous set of documents. Therefore, the entropy of the document set should be lower when the performance obtained for a given query is good.

If the entropy of the set is high, the linguistic structure of the documents is highly variable. Documents with high linguistic variability pose a greater risk that some retrieved documents will be irrelevant. The probability of retrieving irrelevant documents increases with K (Grivola *et* al., 2005). As a single long, non-relevant document can cause a significant increase in entropy, it is advisable to keep K small (Grivola *et* al., 2005). Considering these factors, we use the following entropy formula:

$$H \;=\; -\sum_{c}^{K} nf(c)\ x\ \frac{log(nf(c))}{log(K)}$$

In our entropy calculation formula *H*, *nf(c)* represents the normalized frequency of each retrieved class and *K* is the number of classes found in the search.

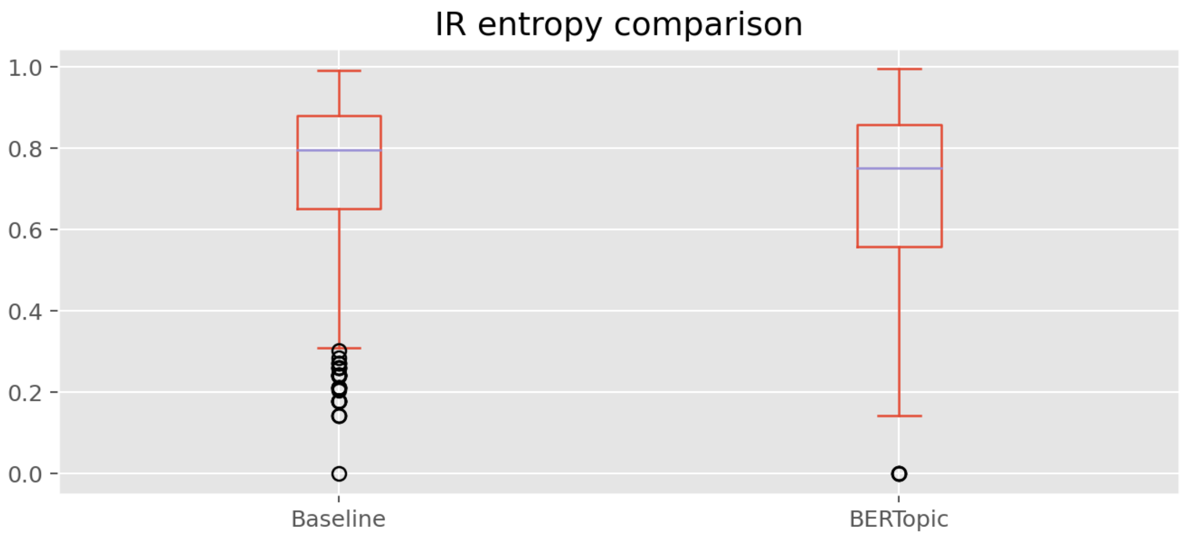
4. Preliminary Experiments

We conducted our experiment in a Jupyter Notebook using the Python language. We started by configuring an Elasticsearch server, creating a BM25 environment to carry out our empirical tests. We feed the system our textual data from the *Dmoz-Computers-500*.

First, we did our experiment with our baseline model, that is, our original documents without semantic enrichment. We tested just one query ('*neural networks for linux systems*') in our search engine and calculated entropy from the distribution of classes in the information retrieval result. Next, we find the most important bigrams from our set of documents to use each of them as a new query, reserving the results obtained and the entropy calculation for each one.

In the second part of our study, we transformed our data into a topic model. We start by eliminating *stopwords*, then vectorize our documents. We then reduce the dimensionality of our vector space with the UMAP algorithm and cluster the vectors of our documents with the HDBSCAN algorithm. Finally, we created a CTF-ICF topic template. Once we had our topic model, we enriched the documents with the most relevant words from the most relevant topic in each document, creating a new field in the search system with the topic terms. We repeat the process applied to the baseline, but now with the new field, that is, we create bigrams to use as queries and reserve the results obtained with our enriched model to calculate the entropy in each of the queries.

Once we had our baseline and our hypothesis we were able to compare them.



As we can see in our *boxplot*, the mean entropy of the enriched documents was lower than our baseline (0.68 and 0.74 respectively), remembering that the lower the entropy, the better our result. In addition, the third quartile had a lower score in enriched documents. We can therefore conclude that the hypothesis model, (BERT + topic model) + BM25, improved the search results.

In the future, we will conduct new experiments looking for other parameters that may generate even better results. Further, we will also conduct experiments on more datasets. Thus, we will do a statistical analysis of the results obtained using Friedman Test, to check if the improvement is relevant.

BIBLIOGRAPHIC REFERENCES

Beiske, K. (2013) “Similarity in elasticsearch,” *Elastic Blog*. Elastic, 26 November. Available at: https://www.elastic.co/en/blog/found-similarity-in-elasticsearch (Accessed: July 3, 2022).

Grivolla, J., Jourlin, P. and De Mori, R. (no date) *Automatic classification of queries by expected retrieval performance*, *Grivolla.net*. Available at: http://www.grivolla.net/articles/sigir2005-qp.pdf (Accessed: July 3, 2022).

Grootendorst, MP (no date) *The algorithm*, *Github.io*. Available at: https://maartengr.github.io/BERTopic/algorithm/algorithm.html (Accessed: July 3, 2022).

Jimenez, S. *et* al. 'BM25-CTF: Improving TF and IDF Factors in BM25 by Using Collection Term Frequencies'. 1 Jan. 2018 : 2887 – 2899. Available at: https://www.researchgate.net/profile/Sergio-Jimenez-7/publication/325231406_BM25-CTF_Improving_TF_and_IDF_factors_in_BM25_by_using_collection_term_frequencies/links/5b0d8349aca2725783f140e5/BM25-CTF-Improving-TF-and-in-using-factor-by-IDF-25 collection-term-frequencies.pdf (Accessed: July 3, 2022).

Kamal, A. (2021) *Building your favorite TV series search engine - Information Retrieval Using BM25 Ranking*, *Medium*. Available at: https://abishek21.medium.com/building-your-favourite-tv-series-search-engine-information-retrieval-using-bm25-ranking-8e8c54bcdb38 (Accessed: July 3, 2022).

Manning, Christopher D., et al. (2008) *Introduction to information retrieval*. Cambridge University Press. Available at: https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf (Accessed: July 3, 2022).

Moura, MF (2009). 'Contributions to the construction of topic taxonomies in restricted domains using statistical learning'. Doctoral thesis. Institute of Mathematics and Computer Sciences of the University of São Paulo (ICMC-USP), São Carlos. Available at: https://teses.usp.br/teses/disponiveis/55/55134/tde-05042010-162834/publico/MFM_Tese_5318963.pdf (Accessed: July 3, 2022).

Nogueira, R. *et al.* (2019) “Document expansion by query prediction,” *arXiv [cs.IR]*. Available at: http://arxiv.org/abs/1904.08375 (Accessed: July 3, 2022).

Open directory project.org: ODP web directory built with the DMOZ RDF database (no date) *Odp.org*. Available at: http://www.odp.org/homepage.php (Accessed: July 3, 2022).

Seitz, R. (no date) *Understanding TF-IDF and BM-25*, *KMW Technology*. Available at: https://kmwllc.com/index.php/2020/03/20/understanding-tf-idf-and-bm-25/ (Accessed: July 3, 2022).

Yates, A., Nogueira, R., & Lin, J. (2021). Pretrained transformers for text ranking: BERT and beyond. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Available at: https://arxiv.org/pdf/2010.06467v1.pdf (Accessed: July 3, 2022).