Proceedings of the 18th Conference on Computer Science and Intelligence Systems pp. 1271–1276
DOI: 10.15439/2023F8119
ISSN 2300-5963 ACSIS, Vol. 35

# Hybrid retrievers with generative re-rankers

Marek Kozlowski
National Information Processing Institute
Warsaw, Poland
Email: marek.kozlowski@opi.org.pl

***Abstract*—The passage retrieval task was announced during PolEval 2022 (SemEval-inspired evaluation campaign for natural language processing tools for Polish). Passage retrieval is a crucial part of modern open-domain question answering systems that rely on precise and efficient retrieval components to identify passages that contain correct answers. Our solution to this task is a multi-stage neural information retrieval system. The first stage consists of a candidate passage retrieval step in which passages are retrieved using federated search over sparse (BM25) and dense indexes (two FAISS indexes built using bi-encoder type retrievers based on Polish RoBERTa models). The second stage consists of a re-ranking step of the previously selected passages with a neural model, mt5-13b-mmarco. The model scores each passage by its relevance to a given query. The highest-scoring passages are then retained as the final result. Our system achieved second place in the competition.**

## I. Introduction

PASSAGE retrieval is a crucial part of modern information retrieval systems that rely on highly efficient retrieval components to identify passages (mostly represented as paragraph(s)) that contain correct answers.

Information retrieval is a popular research domain that focuses on obtaining relevant information from a collection of diverse data resources (mainly textual ones). When working with information retrieval tasks, one can rely on using bag-of-words (BOW) systems (such as the BM25) or different approaches supported by deep learning models (such as dense retrievers or re-ranking modules).

Recently, neural information retrieval has surpassed the lexical methods based on BOW (such as TF-IDF + cosine similarity or BM25) by fine-tuning pre-trained language models, including generative ones such as BART, T5, and representation ones such as BERT, RoBERTa. Although they beat the classical methods in terms of quality efficiency, they are not free of drawbacks, such as the need for a relevant training set. They solve many problems of lexical methods, including poor semantic capabilities, but at the cost of an expensive training process that uses a relevant number of labeled examples. There is also evidence that neural information retrieval systems are characterized by poor generalizability to other domains. This means that in a zero-shot or few-shot setup (i.e. no or little training data), lexical methods remain competitive with, or even better than, neural models.

The lack of Polish-language datasets, relevant evaluations, and benchmarks encouraged the Polish AI community to establish the PolEval initiative, a SemEval-inspired evaluation campaign for natural language processing tools for Polish. Submitted tools compete against each other using available data in tasks selected by the organizers, and are evaluated according to pre-established procedures. In 2022, the following tasks were announced: punctuation prediction from conversational language, abbreviation disambiguation, and passage retrieval.

The goal of the passage retrieval task was to develop a system for cross-domain question answering retrieval in the Polish language.

The participants were given a training set that comprised question–passage pairs from the trivia domain—the type of general-knowledge questions that are typical on popular television quiz shows. For each test question, the systems were tasked with retrieving ordered lists of the ten most relevant passages (i.e. those that contain the answer) from the provided corpus. The systems were scored based on their performance on all three test sets.

Using the PolEval data and our own (the translated MS MARCO dataset), we evaluated various approaches toward passage retrieval, where the sentence-level queries are given and the corpus is counted in millions of passages. We present the best of our submitted approaches.

This article is structured as follows: In Section 2, we discuss related work. Section 3 is devoted to presenting the datasets. In Section 4, we present our approach and the results of our evaluations. Section 6 outlines our conclusions.

## II. Related Work

We commence this section from the standard formulation of the passage retrieval task. From a finite, but arbitrarily large collection of passages $P = \{p_1, p_2...\}$, the system's task, given a query $q$, is to return a top-N ranking of the passages that maximizes a metric of quality, such as normalized discounted cumulative gain (NDCG) or average precision. NDCG is a measure of ranking quality. Highly relevant documents are more useful than moderately relevant documents, which are, in turn, more useful than irrelevant documents. In the subsections below, we explain sparse and dense retrieval, as well as dense re-ranking.

### A. Sparse retrieval

Traditionally, retrieval has been dominated by lexical approaches like TF-IDF + cosine similarity, and BM25.

BM25 is a BOW retrieval function that ranks sets of documents based on the query terms that appear in them, regardless

This work corresponds to the Poleval 2022—Passage Retrieval competition.

of their proximity [1]. BM25 is a retrieval model based on the probabilistic retrieval framework. BM25 often achieves better performance compared to TF-IDF, which rewards term frequency and penalizes document frequency. BM25 goes beyond this to consider document length and term frequency saturation.

The results [2] demonstrate that BM25 is a robust baseline. However, these approaches suffer from a lexical gap and are able to retrieve only documents that contain keywords present within the query. In addition, lexical approaches treat queries and documents as BOW by not considering word order.

To overcome this lexical gap, techniques to improve lexical retrieval systems using neural networks have been proposed. Sparse methods, such as docT5query [3], identified document expansion methods that use sequence-to-sequence models that generate possible queries for which a given document would be relevant. At base, it involves training a model that predicts questions for which the input document might contain answers. The generated questions are then appended to the original documents, which are indexed. The docT5query model takes its name from the generative model T5. The primary advantage of this approach is that expensive neural inference is pushed to indexing time.

### B. Dense retrieval

Recently, dense retrieval approaches have also been proposed. They are capable of capturing semantic matches, and attempt to overcome the (potential) lexical gap. Dense retrievers map queries and documents in a single, common dense vector space. A bi-encoder architecture based on BERT-type models demonstrated strong performance for various open-domain question answering tasks.

An important recent innovation for passage retrieval is the introduction of dense retrieval models that take advantage of a bi-encoder design. Bi-encoders produce two corresponding embeddings for a given two-sentence pair (e.g. a query and a passage), which can then be compared efficiently using cosine similarity.

Bi-encoders are used whenever a sentence embedding is needed in a vector space for efficient comparison in applications such as information retrieval, semantic search, or clustering. Cross-encoders would be the wrong choice for these applications, because a cross-encoder does not produce a sentence embedding; it processes both sentences simultaneously through the Transformer network, which is very computationally expensive with such a large scale of data.

With sufficient labeled data, we can learn encoders (typically, Transformer-based models) that project queries and documents into a dense (semantic) representation space (e.g. 768 dimensions) where the relevance ranking can be recast as a nearest neighbor search over representation vectors [4].

Bi-encoders can be used also as re-rankers, working not on all documents in corpora, but only on subsets of them. There are settings in which the first-stage retriever returns a limited number of documents and passes them to the re-ranker. Re-ranking can be also performed as a second stage retrieve on a limited collection of documents from stage 1 without any top-k constraints.

The two most popular bi-encoders are DPR and ANCE. DPR [5] is a two-tower bi-encoder trained with the hard negatives and single-batch negatives of a single BM25. The Multi-DPR model is a BERT-base-uncased model trained on four QA datasets: NQ, TriviaQA, WebQuestions, and CuratedTREC. ANCE [6] is a bi-encoder that uses approximate nearest neighbor negative contrastive learning, which selects hard training negatives globally from the entire corpus.

### C. Dense re-ranking

Modern search engines are developed as multi-stage retrieve & ranking architectures in which a first-stage retriever generates candidate documents that are then re-ranked by deep learning models. Re-ranking requires feeding the model both the query and the candidate text. Neural re-ranking approaches use the output of a first-stage retrieval system to create a better order of the retrieved documents.

Significant improvements in the performance of re-ranking has been achieved using the cross-encoder mechanism based on BERT-type models. However, it entails the disadvantage of high computational overhead, because cross-encoders do not scale well for large datasets.

In the generative model era, the T5 model [7] was applied in an identical manner to classical cross-encoders, and yielded SOTA results in zero-shot scenarios [8]. MonoT5 is an adaptation of the T5 model [7] proposed by Nogueira [9]. The model uses query–document pairs as input and generates probability scores that quantify the relevance between them. The model is asked to generate either a “true” or a “false” token for a source prompt that contains a query and a document, from which we can extract the probability of relevance used to sort the candidates. In [8], large re-rankers such as monoT5-3B outperform distilled ones and dense models of equivalent size on zero-shot tasks. This approach outperformed other fine-tuned re-ranking models significantly in data scarcity scenarios. The average results of monoT5-3B [8] demonstrate that strong zero-shot effectiveness in new text domains can be achieved by increasing the number of model parameters and without fine-tuning in-domain data. In summary, the monoT5 model, fine-tuned on the MS MARCO passage dataset, achieves state-of-the-art results on the TREC Deep Learning Track, as well as impressive zero-shot effectiveness on BEIR and many other datasets [4].

## III. Dataset

Quality, representativeness, and quantity are crucial aspects of any dataset. Neural language models must usually be pre-trained and fine-tuned on high-quality labeled examples, such as documents, queries, or passages. For many languages, the available training and test datasets are limited or biased. In the sub-sections below, we present the dataset provided by the organizers, then our Polish translation of MS MARCO—a collection of datasets that focuses on deep learning in search.

### A. The PolEval dataset

The PolEval organizers prepared datasets that consist of question–passage pairs from domains as diverse as general knowledge, legal matters, and the FAQs section of Polish e-commerce website Allegro[1].

*Training dataset*. The training set consists of 5000 trivia questions: the type of general-knowledge questions that are typical on popular television quiz shows. Each question in the training set has up to five passages from the Polish-language version of Wikipedia manually assigned. These contain the answer to the question. The training set consists of 16 389 question–passage pairs. Additionally, the PolEval organizers released a Wikipedia corpus of 7 million passages. The raw Wikipedia dump was parsed using WikiExtractor and split into passages at the ends of the paragraphs, or if the passage was longer than 500 characters. The training set comprises only data from the general knowledge domain[2].

*Test dataset*. There were three partial test sets with questions from different domains. The first consists of 1291 general knowledge questions that are similar to those from the training set. The second consists of 900 questions and 921 passages regarding the Polish e-commerce platform Allegro. The dataset was created based on FAQs available on the Allegro portal. Each question–passage pair was verified manually. The third dataset contains over 700 questions from the legal domain. The dataset was built in reverse because some part of the questions were created by random selection of the provisions and asking questions based on their content. The legal-domain-oriented passages count approximately 26 000 provisions extracted from more than 1000 laws published between 1993 and 2004.

Using those test sets, the organizers created the validation, test-A, and test-B datasets. They contain 599, 1200, and 1709 examples, respectively.

### B. The Polish translation of MS MARCO

Microsoft Machine Reading Comprehension (MS MARCO)[3] [10] is a collection of datasets that focuses on the evaluation of modern machine learning methods in different search challenges. The first dataset was a question-answering set that features 100 000 real Bing questions and human-generated answers. Over time, the collection has expanded to at least one-million-questions, a natural language generation dataset, a passage ranking dataset, a keyphrase extraction dataset, a crawling dataset, and a conversational search dataset.

MS MARCO has one drawback: it is a large-scale dataset that focuses chiefly on the English language. A translation is available of these English corpora. MMARCO[4] [11] is a multilingual version of the MS MARCO passage ranking dataset that comprises 13 languages. It was created using machine translation. This dataset demonstrates good transfer learning capabilities, as well as being a popular choice for the evaluation of deep learning models. Using the machine translation approach to create new datasets minimizes the high costs of extensive manual data annotation processes. However, MMARCO does not contain data for the Polish language.

To address this at the National Information Processing Institute (with the assistance of Dr. Sławomir Dadas), we prepared the Polish training set using a translation of MS MARCO into Polish (approximately 39 million triplet translations) with two type of models: a) mbart-large-50-one-to-many-mmt (a fine-tuned checkpoint for multilingual machine translation of the mBART-large-50 model) [12] and b) our in-house English–Polish convolutional neural machine translation models trained using the Fairseq sequence modeling toolkit[5]. We used different neural machine translation models because the quality of Polish translations varies across the MS MARCO dataset. Finally, we mixed them heuristically. The process of machine translation lasted a few days and consumed 8xV100 GPUs.

## IV. Approach

Our approach is inspired by [13], in which the authors describe NeuralSearchX, a metasearch engine based on a multi-purpose large re-ranking model that merges results and highlights sentences.

Our solution is a multi-stage neural information retrieval system. The first stage involves a candidate passage retrieval step in which passages are retrieved using federated search over sparse (BM25) and dense (two FAISS indices built using dedicated RoBERTa-based encoders, the result of the bi-encoders' training) indices. The second stage involves a re-ranking step of the previously selected passages with a fine-tuned neural model, mT5. The model scores each document by its relevance to a given query. The top-scoring documents are then retained as a final result.

### A. Candidate passage retrieval

In the first stage, we used both the sparse and dense retrieval methods provided by the BEIR library [2]. BEIR is a heterogeneous benchmark that contains diverse information retrieval tasks. It also provides a common and easy framework for evaluation of various natural-language-processing-based retrieval models within the benchmark. We opted to use BEIR to take advantage of its support for lexical and dense retrievers.

In our solution, we use Elasticsearch[6]—which applies the BM25 method for scoring documents against queries—as a lexical retriever. BM25 is Elasticsearch's default similarity ranking algorithm. Elasticsearch is a distributed search and analytics engine built on Apache Lucene. Since its release, Elasticsearch has quickly become the most popular search engine in developed systems and is commonly used for full-text search when huge masses of textual data are involved. To support Polish language, we installed the morfologik plugin[7] in our Elasticsearch instance. This plugin is crucial for

[1] https://beta.poleval.pl/challenge/2022-passage-retrieval
[2] http://poleval.pl/tasks/task3
[3] https://microsoft.github.io/msmarco/
[4] https://github.com/unicamp-dl/mMARCO
[5] https://github.com/sdadas/polish-nlp-resources#convolutional-models-for-fairseq
[6] https://www.elastic.co
[7] https://github.com/allegro/elasticsearch-analysis-morfologik

normalization purposes—specifically, we used it to perform lemmatization of words to improve the search efficiency.

The dense retrievers in the proposed approach are Sentence-BERT-type encoders. We trained them in the Bi-encoder architecture, and used them with FAISS index[8] support. FAISS is a library for the efficient similarity search and clustering of dense vectors. Bi-encoder architectures are used because sentence embedding in a vector space is needed for efficient comparison during semantic search or clustering. The queries and passages are passed independently to the sentence transformer to produce fixed-size embeddings. These can then be compared using cosine similarity to identify matching passages for a given query. The training of the dense retrievers is performed by fine-tuning encoders—specifically, we fine-tuned the RoBERTa model used in our bi-encoder architecture. During training, we used a loss function called MultipleNegativesRankingLoss, $number\_of\_epochs = 1-10$ and $batch\_size = 32$. We pass triplets in the format: $(query, positive\_passage, negative\_passage)$, where negative passage is a hard negative example (not positive one, but lexically similar to the positive one) that is retrieved by lexical search in the whole passage corpora. We used Elasticsearch to obtain ($max = 10$) hard negative examples for given positive passages. We trained two types of dense retrievers: a) using RoBERTa-base-v2[9] as a transformer model, and training one epoch on 500 000 triplets (135 000 Poleval ones and 370 000 Polish MS Marco ones randomly selected); and b) using RoBERTa-large-v2[10] as a transformer model, and training 10 epochs on a few million triplets (several million Polish MS Marco ones randomly selected). We then fine-tuned for one epoch on all 135 000 PolEval triplets.

More information on how to train bi-encoders effectively can be found in the sentence-transformers github[11].

Since all of the retrievers are independent, we could run them in parallel—therefore not creating a significant overhead in the process and maintaining an adequate latency.

In the last phase, we collected the top-$K$ results from three retrievers (one sparse and two dense), where the limit $K$ is set for each retriever, respectively.

### B. Re-ranking

After the candidate passage retrieval, the next step involved merging all of the candidate passages into a single list, and then ranking the documents so that the most relevant ones were at the top of the results list. To re-rank, we used generative models of type T5 [7]—specifically, the multilingual version of mT5: the mt5-13b-mmarco-100k model[12]. Re-ranking is performed in the Polish language, however, we used the multi-language model already fine-tuned as a re-ranker, using the following prompt:

$$Query : \{query\_text\}\ Document : \{pass\_text\}\ Relevant :$$

The model was asked to generate for a given prompt either a "true" or a "false" token, from which we could extract the probability of relevance used to sort the candidates. Some pseudo-code of the batch prediction is contained in Listing 1.

The mT5 model used in our solution contains 13 billion parameters; in other words, it is an XXL model. It is based on the T5 model [7], and its adaptation to re-ranker was proposed by Nogueira [9]. It has recently been demonstrated that this model yields state-of-the-art results in zero-shot scenarios [8]. We used a variant based on the multilingual version of T5 called mT5, which was pre-trained on the multilingual mC4 dataset. The mt5-13b-mmarco-100k has been already fine-tuned for re-ranking, using the mMARCO dataset, a multilingual version of the MS MARCO passage ranking dataset that comprises 13 languages and was created using machine translation.

The mT5-based re-ranker computes the relevance (as a probability) of each passage for a given query. After all passages are scored, the list of results is re-ordered according to those scores.

**Algorithm 1** Batch re-ranking relevance predictions using the T5 class model, function: get list of query–passage pairs to verify and return list of probabilities of their relevance.

```
def predict(self,
        query_passage_pairs: List[Tuple[str, str]],
        batch_size: int = 16) -> List[float]:

    probability_scores = []

    # create batches
    batches = []
    for i in range(0, len(query_passage_pairs),
                       batch_size):
        batches.append(
        query_passage_pairs[i: i + batch_size])

    for batch in tqdm(batches):
        #"Query: {q_txt} Passage: {p_txt} Relevant:"
        prompts = [f"Query: {q_p_pair[0]} " \
                f"Document: {q_p_pair[1]} Relevant:"
                for q_p_pair in batch]

        res = self.modelT5.predict_in_batch(prompts)

        for label, prob in zip(res[0], res[1]):
            final_prob = prob
            if label != 'true':
                final_prob = 1 - final_prob

            probability_scores.append(final_prob)
    return probability_scores
```

## V. Results

Using the validation, test, and train PolEval datasets and our own (the translated MS MARCO dataset), we present the various approaches toward a specific information retrieval challenge: the problem of passage retrieval where the sentence-

[8]https://github.com/facebookresearch/faiss

[9]https://huggingface.co/sdadas/polish-roberta-base-v2

[10]https://huggingface.co/sdadas/polish-roberta-large-v2

[11]https://github.com/UKPLab/sentence-transformers/blob/master/examples/training/ms_marco/train_bi-encoder_mnrl.py

[12]https://huggingface.co/unicamp-dl/mt5-13b-mmarco-100k

TABLE I
EVALUATION OF DIFFERENT RE-RANKERS THAT HAVE THE SAME CLASSICAL, LEXICAL RETRIEVER: BM25, AND TOP-K=100 RESULTS RETRIEVED FROM BM25. NDCG METRICS WERE CALCULATED ON THE POLEVAL VALIDATION DATASET.

| Method name | Retriever | Re-ranker | NDCG@10(%) |
|---|---|---|---|
| baseline | BM25 (default) | None | 21.05 |
| baseline@plugged | BM25 (morfologik) | None | 27.67 |
| bi-enc@base | BM25 (morfologik) | Bi-encoder (RoBERTa-base) | 38.03 |
| gpt3@curie | BM25 (morfologik) | GPT3 (curie) | 40.06 |
| bi-enc@large | BM25 (morfologik) | Bi-encoder (RoBERTa-large) | 41.19 |
| mT5@base | BM25 (morfologik) | mT5-base-mmarco | 42.87 |
| **mT5@xxl** | **BM25 (morfologik)** | **mT5-13b-mmarco** | **45.88** |

TABLE II
EVALUATION OF DIFFERENT RETRIEVERS' SETTINGS (SPARSE—BM25, AND TWO DENSE RETRIEVERS BASED ON BI-ENCODER ARCHITECTURE, AND TWO TYPES OF ROBERTA MODELS) OF OUR SOLUTION. THE NDCG METRICS WERE CALCULATED ON THE POLEVAL TEST-A DATASET, THE FIRST RELEASED DATASET THAT WAS USED FOR PUBLIC LEADERBOARD PURPOSES.

| Top@K sparse retriever – BM25 | Top@K dense retriever – bi-encoder (RoBERTa-base) | Top@K dense retriever – bi-encoder (RoBERTa-large) | NDCG@10(%) |
|---|---|---|---|
| 1 | 28 | 28 | 73.71 |
| 7 | 50 | 0 | 74.61 |
| 7 | 0 | 50 | 74.71 |
| **7** | **25** | **25** | **75.32** |
| 7 | 45 | 45 | 74.83 |

level queries are given and the corpus is counted in millions of passages.

First we evaluated various re-ranking methods on the PolEval validation dataset. In table I, we present some of the results, where we have one fixed retriever (BM25) plugged with the morfologik analyzer to introduce Polish lemmatization[13]. We tested the following re-rankers:

1) a bi-encoder based on RoBERTa-base, fine-tuned for one epoch on approximately 500 000 triplets, combining 135 000 PolEval training triplets and randomly selected triplets from the Polish MS Marco dataset;
2) a bi-encoder based on RoBERTa-large, initially fine-tuned for 10 epochs on a few million triplets (several million Polish MS Marco triplets randomly selected), and next fine-tuned for one epoch on 500 000 triplets, combining 135 000 PolEval training triplets and not-yet-used triplets from the Polish MS Marco dataset;
3) a generative model: GPT3 (curie), fine-tuned for one epoch on approximately 200 000 triplets, combining 135 000 PolEval training triplets and randomly selected triplets from the Polish MS Marco dataset;
4) a generative model: mT5-base-mmarco, fine-tuned for one epoch on 135 000 PolEval training triplets (Polish MS Marco was omitted because it is covered semantically by the mMARCO multi-language dataset);
5) a generative model: mT5-13b-mmarco, no fine-tuning.

As presented in Table I, the best results were achieved by mT5-13b-mmarco, an mT5-XXL model, fine-tuned on mMARCO (the multi-language version of MS Marco). Impressive results were achieved in the NeuCLIR track of TREC 2022 by the Unicamp team [14]. Despite the mT5 model being fine-tuned only on query–document pairs of the same language, it proved to be viable for cross-lingual information retrieval tasks, where query–document pairs are in different languages. The results in [14] demonstrate outstanding performance across all tasks and languages. For that reason, we used the same model for the Polish language—even though the mMARCO dataset does not contain Polish. We attempted to fine-tune the mT5-13b-mmarco model using the PolEval training dataset, but this process demanded very expensive infrastructure: at least four GPUs with high RAM capacity (such

[13]https://github.com/allegro/elasticsearch-analysis-morfologik

as the A100 80GB), and $ddp\_sharded$ strategy during fine-tuning to shard the entire model weights across all available GPUs. This approach enables the model's size to be scaled while using efficient communication to reduce overhead. Our initial experiments toward distributed fine-tuning failed to demonstrate any statistically significant improvement. This means that we could have trained it better. Ultimately, our solution used the original mT5-13b-mmarco, without fine-tuning on the PolEval datasets.

After selecting the best re-ranker, we analyzed how to improve the retrieval phase. After a number of experiments, we realized that bi-encoders as dense retrievers complement the BM25 results. Table II presents the evaluation of different retrievers' settings (sparse—BM25, and two dense retrievers based on bi-encoder architecture, as well as two types of RoBERTa models) in our solution. The results suggest that all three retrievers are needed. The first three rows of the table present different distributions of all 57 results sent for re-ranking: a) in the first row, we almost eliminated the BM25 results; b) in the second and third rows, we used only one dense retriever; c) we used seven results from BM25, and 25 from each of the dense retrievers; and d) we attempted to verify whether more dense retrievers results would help the results. Results a) and b) prove that all of of the retrievers are neeeded; result d) proves that increasing the number of dense retrievers by too many fails to improve the final NDCG metric.

## VI. Conclusion

The goal of the PolEval 2022 Passage Retrieval task was to develop a system for cross-domain question answering retrieval in the Polish language.

The participants were given a training set that comprised question–passage pairs from the general knowledge domain, as well as three separate test sets with unpaired questions and passages from different domains: general knowledge, legal matters, and customer support. For each test question, participants were tasked with retrieving ordered lists of the ten most relevant passages from the provided corpora. The systems were scored using the NDCG metric.

Our solution was a multi-stage neural information retrieval system. The first stage involved a candidate passage retrieval step in which passages were retrieved using federated search over sparse (BM25) and dense indices (two FAISS indices built using dense retrievers based on bi-encoder architecture and Polish RoBERTa models). The second stage involved a re-ranking step of the previously-selected passages with a neural model, mt5-13b-mmarco. The model scored each passage by its relevance for a given query. The top-scoring passages were then retained as the final result.

Our system achieved second place in the competition. The results between top three entries did not differ significantly, and the quality of the solutions was high.

## Acknowledgment

We thank Dr. Slawomir Dadas for his support—particularly in the bi-encoder training, and the Polish translation of the MS MARCO dataset.

## References

[1] S. Robertson, H. Zaragoza *et al.*, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[2] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. [Online]. Available: https://openreview.net/forum?id=wCu6T5xFjeJ

[3] R. Nogueira, J. Lin, and A. Epistemic, "From doc2query to doctttttquery," *Online preprint*, vol. 6, 2019.

[4] J. Lin, D. Alfonso-Hermelo, V. Jeronymo, E. Kamalloo, C. Lassance, R. Nogueira, O. Ogundepo, M. Rezagholizadeh, N. Thakur, J.-H. Yang *et al.*, "Simple yet effective neural ranking and reranking baselines for cross-lingual information retrieval," *arXiv preprint arXiv:2304.01019*, 2023.

[5] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," *arXiv preprint arXiv:2004.04906*, 2020.

[6] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. Bennett, J. Ahmed, and A. Overwijk, "Approximate nearest neighbor negative contrastive learning for dense text retrieval," *arXiv preprint arXiv:2007.00808*, 2020.

[7] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[8] G. M. Rosa, L. Bonifacio, V. Jeronymo, H. Abonizio, M. Fadaee, R. Lotufo, and R. Nogueira, "No parameter left behind: How distillation and model size affect zero-shot retrieval," *arXiv preprint arXiv:2206.02873*, 2022.

[9] R. Nogueira, Z. Jiang, R. Pradeep, and J. Lin, "Document ranking with a pretrained sequence-to-sequence model," in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020. doi: 10.18653/v1/2020.findings-emnlp.63 pp. 708–718. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.63

[10] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, "Ms marco: A human generated machine reading comprehension dataset," *choice*, vol. 2640, p. 660, 2016.

[11] L. Bonifacio, V. Jeronymo, H. Q. Abonizio, I. Campiotti, M. Fadaee, R. Lotufo, and R. Nogueira, "mmarco: A multilingual version of the ms marco passage ranking dataset," *arXiv preprint arXiv:2108.13897*, 2021.

[12] Y. Tang, C. Tran, X. Li, P.-J. Chen, N. Goyal, V. Chaudhary, J. Gu, and A. Fan, "Multilingual translation with extensible multilingual pretraining and finetuning," *arXiv preprint arXiv:2008.00401*, 2020.

[13] T. S. Almeida, T. Laitz, J. Seródio, L. H. Bonifacio, R. Lotufo, and R. Nogueira, "Neuralsearchx: Serving a multi-billion-parameter reranker for multilingual metasearch at a low cost," *arXiv preprint arXiv:2210.14837*, 2022.

[14] V. Jeronymo, R. Lotufo, and R. Nogueira, "Neuralmind-unicamp at 2022 trec neuclir: Large boring rerankers for cross-lingual retrieval," *arXiv preprint arXiv:2303.16145*, 2023.