IRise: Study of A Comprehensive Retrieval System Boosted by Neural Search Final Report

Devrim Çavuşoğlu
devrim.cavusoglu@metu.edu.tr

Ezgi Çavaş
ezgi.cavas@metu.edu.tr

Abstract

We present an information retrieval (IR) system designed as a retrieval from a large collection of documents boosted by neural search. We used the MSMarco test split from BEIR. This dataset comprises 8.8 million documents and 9.2 thousand query relevance judgements for 43 queries. Our approach begins with a Vector Space Model (VSM baseline), gradually enhancing performance through experimentation. The system consists of four main modules: data parsing and indexing, query processing, ranking and retrieval, and a user interface. The system features a web-based user interface using lightweight libraries for streamlined development.
Implementation: The code and other supplementary material (i.e. data, documentation etc.) are available at https://github.com/devrimcavusoglu/irise-retrieval-system.

1 Introduction

With the exponential growth of digital information, retrieving relevant documents quickly and accurately has become a significant challenge in the field of information retrieval (IR). Traditional retrieval models particularly Vector Space Models (VSMs), while efficient, often fall short in capturing the semantic similarities between queries and documents. This limitation can lead to less relevant search results, affecting the overall user experience and efficiency of IR systems.

To address these challenges, we developed IRise, an information retrieval system that integrates traditional retrieval models with modern text embedding techniques. Our system employs a two-stage retrieval process: an initial retrieval using a traditional model (e.g., TF-IDF or BM25) followed by a re-ranking step using advanced text embeddings. This approach aims to balance retrieval speed with the relevancy of the results.

The IRise project is particularly interesting because it leverages the strengths of both traditional and modern techniques to improve information retrieval performance. By combining the efficiency of traditional models with the semantic understanding of text embeddings, IRise provides a more accurate and user-friendly search experience. This hybrid approach has practical significance in various applications, from academic research to commercial search engines, where both speed and relevancy are crucial.

The highlights of the IRise project include:

- Utilization of the MSMarco dataset [2], which comprises approximately 9 million documents.
- Implementation of a two-stage retrieval process that first retrieves initial results using traditional models and then re-ranks these results using text embeddings.
- Extensive evaluation and experimentation with different retrieval models and preprocessing techniques to optimize performance.
- Development of a web-based user interface to provide an accessible and user-friendly search experience.

2 Related Work

The traditional VSMs and weighted VSMs [13, 18] have been a go to solution for retrieval systems on large collections. These models use statistics like term or document frequencies to weight document vectors. Through time the weighting strategies have been evolved to perfect document vectors in terms of semantic/relevance performance [17].

For the information retrieval with neural models, a popular and a prominent choice is retrieve-and-rerank system for balancing speed and prediction performance trade-off [11**?**].

For neural IR models, there is a leaderboard[1] (English) based on their retrieval performance on semantic textual similarity (STS) assessment. The leaderboard utilizes MTEB [8], a comprehensive benchmark including cross domain dataset that are for various tasks, for comparing the different models. There's a slight trade-off between the model size and the performance.

For the ANN part, there is an ANN Benchmark[2] [4] for approximate nearest neighbor algorithms. Here, there are several options designed as a high-level interfaces, libraries, or frameworks such as Faiss [3] or Scann [15] libraries.

3 System Architecture

The IRise system is designed with a comprehensive architecture that integrates both traditional and modern retrieval techniques to enhance the relevancy and speed of document retrieval. The system architecture can be divided into two main parts: offline and online components. A high-level overview of the system architecture is shown in Figure 1.

[1] https://huggingface.co/spaces/mteb/leaderboard
[2] https://ann-benchmarks.com/

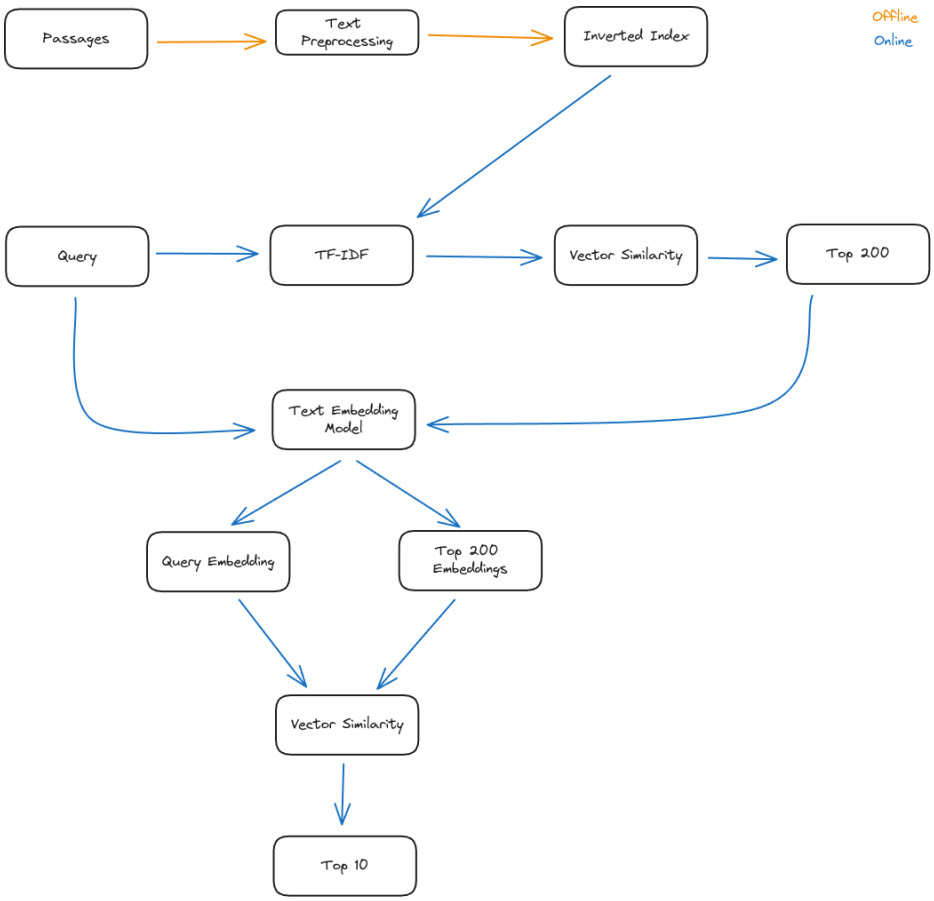

Figure 1. High-level system architecture of IRise.

3.1 Dataset and Preprocessing

We have utilized ir-datasets [6] package for data wrangling. The dataset used in this project is the BEIR's [16] split of the MSMARCO dataset[3] [2], specifically the test split. This dataset comprises approximately 8.8 million documents and includes around 9.2 thousand query relevance judgements for 43 queries. The topics for the queries vary from scientific concepts to general vocabulary knowledge.

During preprocessing, the dataset undergoes several steps to prepare it for indexing:

- **Case Folding:** Texts are converted to lowercase.
- **Stopword Removal:** Utilizing an English stopword list to eliminate common but uninformative words.
- **Stemming:** Applying Porter's stemming algorithm [10] to reduce words to their root forms.
- **Tokenization:** The default English tokenizer is used to split the text into tokens.

[3] https://ir-datasets.com/beir.html#beir/msmarco/test

3.2 Indexing and Initial Ranking

The indexing process is crucial for efficient retrieval. We utilized PyTerrier [7] in the experimentation phase and Whoosh [1] in the final version of the system for indexing the dataset and searching over the collection. The indices created include both direct and inverted indices, facilitating fast search and retrieval operations.

3.3 Retrieve-and-Rerank

The initial retrieval is conducted using traditional VSMs, specifically TF-IDF (Term Frequency-Inverse Document Frequency) [14] and BM25 (Best Matching 25) [12]. These models are effective for quickly retrieving relevant documents from the large collection of documents:

- **TF-IDF:** A statistical measure used to evaluate the importance of a word in a document relative to a corpus.
- **BM25:** An improvement over TF-IDF that incorporates term frequency saturation and document length normalization.

The initial ranking results are obtained by searching the inverted index, and the top 200 results are selected for further

processing. We used TF-IDF in the final version of the system due to efficient and fast retrieval of the initial results.

To improve the relevance of the search results, a re-ranking stage is implemented using a text embedding model. We use the GTE-base model [5], which captures the semantic similarity between the query and the documents:

- **Text Embedding Model (GTE-base):** This model generates embeddings for both the query and the top 200 documents retrieved from the initial ranking. The embeddings are then compared to re-rank the documents based on semantic similarity.

The re-ranking process refines the top 200 results to the top 10 most relevant documents, ensuring higher quality search results. Cosine similarity measure is used for vector based comparison.

3.4 User Interface

The user interface (UI) is a crucial component for interacting with the IRise system. Currently, the UI is web-based, designed for ease of use and accessibility:

- **Query Input:** Users can enter free-form text queries in UTF-8 encoding.
- **Results Display:** The top 10 re-ranked documents are displayed to the user in a table form, with relevant passages from the collection.
- **No Wildcard Queries or Proximity Operators:** The system focuses on relevance and simplicity, not taking complex query operators into consideration. Relevance feedback is not implemented.

3.5 System Workflow

The overall workflow of the IRise system can be summarized as follows:

1. **Data Ingestion and Preprocessing:** The dataset is ingested and preprocessed using stopword removal, stemming, and tokenization.
2. **Indexing:** The preprocessed documents are indexed using PyTerrier and Whoosh, creating both direct and inverted indices.
3. **Initial Retrieval:** User queries are processed using traditional VSMs (TF-IDF/BM25) to retrieve the top 200 documents.
4. **Re-ranking:** The top 200 documents are re-ranked using the GTE-base text embedding model to generate the top 10 most relevant documents.
5. **Results Display:** The top 10 documents are displayed to the user through the web-based UI.

This architecture ensures a balance between retrieval speed and relevancy, leveraging the strengths of both traditional and modern retrieval techniques to enhance the overall performance of the system.

4 Evaluation

The IRise system aims to address the challenges of retrieving relevant documents quickly and accurately by integrating traditional retrieval models with modern text embedding techniques. This hybrid approach is particularly useful in scenarios where both speed and relevance are crucial, such as academic research, legal document search, and commercial search engines. By leveraging the strengths of both types of models, IRise enhances the overall user experience and retrieval performance.

4.1 Experiments

To evaluate the effectiveness of our system, we conducted a series of experiments focusing on different aspects of the retrieval process. These experiments involved:

- Exploring various vector space models (VSMs), specifically TF-IDF and BM25.
- Combining VSM scores using weighted averages, controlled by a parameter lambda λ.
- Testing different text preprocessing techniques, such as stop-word removal and stemming, for creating the inverted index.

We performed a performance comparison between the initial retrieval results obtained using VSMs and the re-ranked results obtained using the text embedding model. The goal was to assess improvements in relevancy metrics and retrieval times.

Evaluation Metrics. To evaluate the performance of our system, we used the following standard information retrieval metrics:

- **P@10 (Precision at 10):** The proportion of relevant documents among the top 10 retrieved documents.
- **MRR (Mean Reciprocal Rank):** The average of the reciprocal ranks of the first relevant document.
- **MAP (Mean Average Precision):** The mean of the average precision scores for each query.
- **nDCG@10 (Normalized Discounted Cumulative Gain at 10):** A measure of ranking quality that considers the positions of relevant documents.
- **Speed (s/it):** The average time taken per iteration (in seconds) to retrieve and rank documents.

4.2 Results

The evaluation results of our experiments are summarized in Table 1. The table shows the performance metrics for both the initial retrieval models (TF-IDF and BM25) and the re-ranked results using the embedding model. The evaluation results are obtained by TREC Eval Metrics [9].

The results indicate significant improvements in relevancy metrics after re-ranking with the embedding model. Specifically:

Model	P@10	MRR	MAP	nDCG@10	Speed (s/it)
TF-IDF	14.52	25.44	0.7	11.44	8.54
BM25	55.00	77.70	7.90	43.55	11.65
TF-IDF(Re-ranked)	56.67	86.19	7.91	49.20	**11.05**
BM25(Re-ranked)	**79.05**	**91.67**	**12.93**	**67.18**	14.86

Table 1. Evaluation results.

- The P@10 metric increased from 14.52 to 56.67 for TF-IDF and from 55.00 to 79.05 for BM25.
- The MRR metric improved from 25.44 to 86.19 for TF-IDF and from 77.70 to 91.67 for BM25.
- The MAP metric showed notable enhancements, particularly with BM25 (7.90 to 12.93).
- The nDCG@10 metric, which considers the ranking positions of relevant documents, improved significantly, especially with BM25 (43.55 to 67.18).
- The retrieval speed showed a slight increase in processing time due to the re-ranking stage but remained within acceptable limits for practical use.

Overall, the evaluation demonstrates that integrating traditional VSMs with modern text embedding techniques substantially enhances retrieval performance, making IRise a valuable system for information retrieval tasks.

5 Conclusion

In this project, we proposed and implemented an information retrieval system, IRise, capable of efficiently searching through a document set of millions. The system employs a two-stage retrieval process, initially using traditional vector space models (VSMs) such as TF-IDF and BM25, followed by a re-ranking stage using a modern text embedding model. This approach effectively balances retrieval speed and relevancy, making it suitable for large-scale document retrieval tasks.

5.1 Achievements

With respect to our initial proposal, we have achieved the following:

- Successfully implemented a two-stage retrieval system that significantly improves relevancy metrics.
- Conducted extensive experiments to explore various VSMs and preprocessing techniques.
- Demonstrated that re-ranking with an embedding model (GTE-base) enhances the performance of initial retrieval results.
- Developed a web-based user interface for easy interaction with the system.

5.2 Limitations

Despite our achievements, there were a few aspects that we could not fully address:

- We could not explore a wide range of neural models due to time constraints.
- Advanced preprocessing techniques, such as custom tokenization or context-aware stop-word removal, were not implemented. Also no support of wildcard queries or proximity search.
- Real-time relevance feedback from users was not incorporated, which could further refine retrieval performance.

5.3 Future Work

There are several avenues for future work that could further enhance the IRise system:

- Exploring more advanced embedding models, such as BERT or GPT-based models, for improved semantic understanding.
- Optimizing preprocessing techniques to include context-aware tokenization and advanced stop-word removal.
- Implementing user feedback mechanisms to dynamically adjust retrieval models based on user interactions.
- Enhancing the user interface to support more complex query types and display richer information about retrieved documents.

5.4 Reflections

Working on the IRise project has been a valuable and challenging experience. The effort required was okay for a small to mid-size research project, involving preliminary study and research, experimentation, and development. Some of the key challenges we faced included:

- Managing the large dataset efficiently to ensure fast retrieval times.
- Balancing the trade-off between retrieval speed and relevancy, especially when integrating advanced models. Experimentation took the most the development time to decide on a final version of the parameters or settings to be used on the live system.
- Debugging and optimizing the implementation to handle edge cases and ensure robustness.

This project has significantly contributed to our understanding of information retrieval concepts and their practical applications. It provided hands-on experience with various retrieval models, preprocessing techniques, and evaluation

metrics. Additionally, it highlighted the importance of continuous experimentation and iteration in developing effective retrieval systems.

Overall, the IRise project was instrumental in deepening our knowledge and skills in information retrieval, aligning well with the course material, and preparing us for future challenges in this field.

References

[1] Matt Chaput. 2009. Pure-Python full-text search library. https://github.com/mchaput/whoosh

[2] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the TREC 2019 deep learning track. *arXiv preprint arXiv:2003.07820* (2020).

[3] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The Faiss library. (2024). arXiv:2401.08281 [cs.LG]

[4] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating Large-Scale Inference with Anisotropic Vector Quantization. In *International Conference on Machine Learning*. https://arxiv.org/abs/1908.10396

[5] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281* (2023).

[6] Sean MacAvaney, Andrew Yates, Sergey Feldman, Doug Downey, Arman Cohan, and Nazli Goharian. 2021. Simplified Data Wrangling with ir_datasets. In *SIGIR*.

[7] Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation inInformation Retrieval using PyTerrier. In *Proceedings of ICTIR 2020*.

[8] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive Text Embedding Benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Andreas Vlachos and Isabelle Augenstein (Eds.). Association for Computational Linguistics, Dubrovnik, Croatia, 2014–2037. https://doi.org/10.18653/v1/2023.eacl-main.148

[9] Joao Palotti, Harrisen Scells, and Guido Zuccon. 2019. TrecTools: an open-source Python library for Information Retrieval practitioners involved in TREC-like campaigns *(SIGIR'19)*. ACM.

[10] Martin F Porter. 1980. An algorithm for suffix stripping. *Program* 14, 3 (1980), 130–137.

[11] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[12] Stephen E Robertson and K Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information science* 27, 3 (1976), 129–146.

[13] Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Commun. ACM* 18, 11 (1975), 613–620.

[14] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[15] Philip Sun. 2020. Announcing ScaNN: Efficient Vector Similarity Search. https://research.google/blog/announcing-scann-efficient-vector-similarity-search/

[16] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. 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)*. https://openreview.net/forum?id=wCu6T5xFjeJ

[17] Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to BM25 and language models examined. In *Proceedings of the 19th Australasian Document Computing Symposium*. 58–65.

[18] Peter D Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research* 37 (2010), 141–188.