Combining Lexical & Semantic retrievals for Query Search

Sanchit Goel
UC San Diego
s5goel@ucsd.edu

Arya Rahnama
UC San Diego
erahnama@ucsd.edu

Abstract

This study explores the integration of lexical and semantic retrieval methods to improve query search accuracy [1] [2]. Traditional search engines often rely on either lexical or semantic-based approaches. This integrated approach leverages the strengths of both methods to improve overall retrieval accuracy and contextual relevance. We evaluated our system using two datasets, Scifact and Tiny MS Marco, to measure its effectiveness. Our results demonstrate that ensemble methods outperform individual approaches, with significant improvements in metrics such as nDCG and MAP. The ensemble of DistilBERT and BM25 achieved notable performance gains, showing enhanced retrieval accuracy across both datasets. Even without fine-tuning, the combination of lexical and semantic methods provided robust performance improvements.

Demo:
https://www.youtube.com/watch?v=fiRoZcSJFoA

1 Introduction

Search engines are fundamental tools in information retrieval, heavily influencing our online interactions by determining the relevance and accuracy of retrieved content. Search engines must balance speed with accuracy to be effective. Modern search engines typically employ a multi-stage pipeline architecture. This architecture begins with a fast and efficient retriever that scans the entire corpus to generate an initial set of candidate documents. The retriever often relies on either the simple lexical matching techniques or a semantic matching technique. The lexical retrieval are often implemented using the BM25 algorithm, which is based on term frequency and document length. While these methods are individually efficient for straightforward queries, they struggle with more complex queries that require an understanding of the context as well as word-matching.

Following the initial retrieval stage, one or more rerankers refine the search results. Rerankers apply more sophisticated algorithms to reorder the initial set of documents, improving the relevance and precision of the final results presented to the user. This stage often involves deeper analysis, leveraging complex models to enhance the initial retrieval set.These models utilize advanced machine learning techniques, particularly deep learning, to understand and encode the meaning of queries and documents into dense vector representations.

Our project aims to bridge the gap between these two approaches by combining lexical and semantic retrieval methods into a single, cohesive search system. We propose an ensemble search system that leverages the strengths of both BM25 for precise word-level matching and advanced language models like DistilBERT and Llama for semantic understanding. This integrated approach is designed to enhance overall retrieval performance by balancing the precision of lexical methods with the contextual relevance provided by semantic models. Ultimately, our goal is to contribute to the development of more advanced and contextually aware search engines.

2 Related Work

The field of information retrieval has seen significant advancements with the development of both lexical and semantic retrieval models. Traditional lexical models like BM25 have been widely used due to their efficiency and robustness in scoring documents based on term frequency and document length. However, the limitations of these models in handling semantic nuances have driven the exploration of integrating semantic retrieval methods.

In the Complementary Lexical Retrieval Model with Semantic Residual Embeddings, Gao et al. (2021) represented a notable advancement in this in-

[1] https://github.com/AbdullahAshfaq/slm4search
[2] https://www.youtube.com/watch?v=fiRoZcSJFoA

tegration. CLEAR improves retrieval effectiveness by leveraging negative examples from documents incorrectly retrieved by lexical models, training an embedding model that distinguishes relevant from irrelevant documents based on semantic content. This approach introduces a dynamic margin in the loss function, which adjusts according to the lexical scores of query-document pairs, thereby enhancing the model's ability to learn semantic information that complements lexical retrieval when necessary.

Further research has highlighted the importance of model architecture and fine-tuning strategies in semantic search pipelines. Studies comparing task-specific and general language models have found that smaller, task-specific models can outperform larger, general models in certain tasks, emphasizing the need for tailored approaches in achieving high performance in semantic textual similarity tasks.

Mitra and Craswell (2017) found that the Neural ranking models for information retrieval mark another significant development. These models employ neural networks to rank search results based on queries, learning language representations directly from raw text. They encompass a range of techniques from shallow to deep neural IR methods, and propose future directions for enhancing neural IR.

Our project builds upon these advancements by combining BM25 with cutting-edge semantic models like DistilBERT and Rep-Llama. By harnessing the strengths of both lexical and semantic retrieval methods, we aim to develop a more effective and contextually aware search system, capable of addressing the inherent limitations of traditional information retrieval approaches.

3 Methodology

3.1 Lexical Retrieval

Lexical retrieval is a fundamental approach in information retrieval systems that focuses on matching the terms in a user's query with the terms in a document. This technique relies on the lexical properties of the text, meaning it looks for exact word matches between the query and the documents in the database. The primary advantage of lexical retrieval is its simplicity and directness, which allows for efficient processing, especially with large datasets. It does not consider the semantic meaning of the words but strictly their occurrence and frequency. This method is highly effective for queries where precise keyword matches are crucial, such as finding documents that contain specific terms or phrases exactly as they appear in the query.

In lexical retrieval, the most common models used are based on the bag-of-words approach, where documents and queries are represented as sets of words without considering the order or context. Each word's presence and frequency within a document are the main factors influencing the retrieval process. Despite its straightforward nature, lexical retrieval remains a cornerstone of search engines and information retrieval systems due to its robustness and the ability to handle large-scale text data efficiently.

3.1.1 BM25

BM25, or Best Matching 25, is one of the most popular algorithms used in lexical retrieval. It is a probabilistic retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document. The BM25 algorithm enhances the basic bag-of-words model by incorporating several key factors that improve its effectiveness and relevance in information retrieval tasks.

3.1.2 Why BM25?

BM25 is favored for its simplicity and computational efficiency, making it particularly suitable for large-scale retrieval tasks where processing speed is critical. Despite its simplicity, BM25 often performs competitively with more complex models, especially in scenarios where exact keyword matches are essential. Its ability to balance between term frequency and document length normalization helps in maintaining high retrieval performance across diverse datasets.

3.1.3 Key Features of BM25

- **Term Frequency (TF):** BM25 considers the frequency of each query term in the document. Higher term frequency typically indicates higher relevance, as it suggests that the document extensively covers the topic represented by the query term.

- **Inverse Document Frequency (IDF):** This feature weighs down the importance of common terms that appear across the corpus, giving more weight to rare, informative terms. IDF helps in differentiating between documents that contain common words and those that contain more unique, query-specific words.

- **Document Length Normalization:** BM25 adjusts the score based on the length of the document to avoid bias towards longer documents. This normalization ensures that longer documents, which naturally have more terms, do not unfairly dominate the search results.

The formula for BM25 incorporates these features and is defined as

$$\text{Score}(D, Q) = \sum_{t \in Q} \text{IDF}(t) \cdot \frac{f(t, D) \cdot (k_1 + 1)}{f(t, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{\text{avgdl}})}$$

Where:

- Q is the query
- D is the document
- t is the query term
- $f(t, D)$ is the term frequency of t in D
- $|D|$ is the length of the document
- avgdl is the average document length in the corpus
- k_1 and b are hyperparameters that control term frequency saturation and document length normalization, respectively.

BM25's effectiveness and efficiency make it a robust choice for information retrieval, balancing the need for precision and computational practicality.

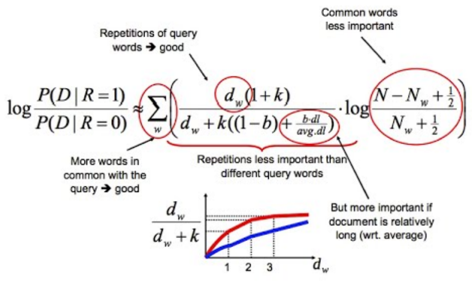
Common words less important
Repetitions of query words → good
$\log \frac{P(D|R=1)}{P(D|R=0)} \approx \sum_w \left(\frac{d_w(1+k)}{d_w + k((1-b) + \frac{b \cdot dl}{avg.dl})} \cdot \log \frac{N - N_w + \frac{1}{2}}{N_w + \frac{1}{2}} \right)$
More words in common with the query → good
Repetitions less important than different query words
$\frac{d_w}{d_w + k}$
But more important if document is relatively long (wrt. average)
1
2
3
d_w

Figure 1: An intuitive overview of the BM25 score.

3.2 Semantic Retrieval

Semantics is a branch of linguistics concerned with how the words in a sentence interact with each other to convey meaning. For semantic retrieval, we can't use BOW approaches as those don't take the position of a word with respect to others in a sentence and assume the same representation of the word irrespective of usage. The best-performing semantic retrieval methods involve the use of neural networks. In particular, the performance of transformer architecture is impressive because it learns which word relates to which other words.

Neural network approaches require optimization of network parameters in a supervised, unsupervised, or self-supervised manner. So we need to create a dataset that contains positive and negative labels.

Negative Generation

A negative sample means that for a query, which documents are not relevant. Usually, the commonly available datasets like MS Marco and Scifact contain only the positive labels i.e. they only provide one relevant document to each query. The task of generating negative labels is tricky but important as we will see in the experimentation section how the choice of negative labels affects the results. There are numerous ways to generate negative labels. These can be generated randomly, labelled by humans, or in a data-driven way.

Zhan et al. (2021) showed that random negatives have the worst performance. The reason is because the negatives are very easy to distinguish. We need harder negatives for training which look very similar to the positive but don't answer the query. Hard negatives can be labeled by humans and these would have higher quality. However, the neural network approaches are data-hungry and require lots of labelled data. So a data-driven approach to generating hard negatives maybe best. For this purpose, we used BM25 to generate hard negatives.

BM25 is an effective way to generate hard negatives. The procedure is that for each query, we use BM25 to retrieve relevant documents. The documents which are not in the positive samples are considered negative. There is noise in this method as a query has several relevant documents which we may incorrectly label as irrelevant. But overall, this performs better than the random negatives.

After getting positive and negative samples, we train the models using multiple negatives ranking loss which tries to reduce distance from the positive label and increase distance with negative labels (Henderson et al., 2017)

$$L = -\frac{1}{N} \cdot \frac{1}{K} \cdot \sum_{i=1}^{K} \left[S(x_i, y_i) - log \sum_{j=1}^{K} e^{S(x_i, y_i)} \right]$$

SBERT

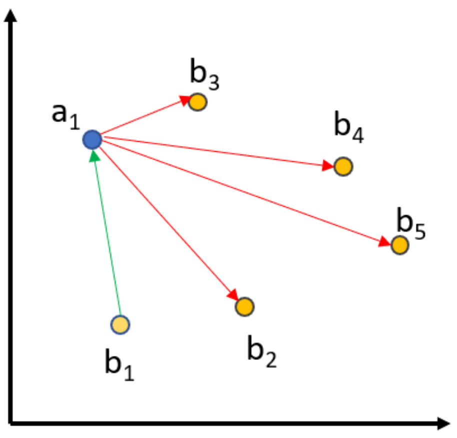

Figure 2: Multiple Negative Ranking Loss

When comparing sentences, we need to find the representation of both query and corpus. Simple BERT has good performance but is slow in this task. SBERT is a modification to it that uses Siamese and triplet network structure to derive semantically meaningful embeddings that can be compared by cosine similarity and reduces execution time from 65 hours to 5 seconds (Reimers and Gurevych, 2019).

DistilBERT

It is smaller, faster, and cheaper than BERT with over 95% of performance and 40% of the size of BERT. In IR, speed is important so we went with this model from the BERT family of models. It learns bidirectional representation from the unlabelled text by jointly conditioning the context in all layers. It uses knowledge distillation during the pretraining phase and reduces the size of BERT. (Sanh et al., 2020)

Biencoder Llama

A more state-of-the-art neural network that we tried was Llama2-7b. It is a a unidirectional model and not trained for IR specifically so we had to finetune it. We used repllama for this purpose which has a biencoder architecture but with a Llama backbone. (Ma et al., 2023) The llama model is finetuned by appending a special token at the end. The encoding of that token represents the sentence.

$$V_T = Decoder(t_1 t_2 ... t_k < /s >)[-1]$$

Then we use InfoNCE loss (Noise contrastive estimation) which optimizes the negative log probability of classifying the positive sample correctly. (Parulekar et al., 2023)

After we have the encodings by various models, we can compute the similarity between queries and documents using cosine similarity.

We used RepLlama to evaluate bi-encoder Llama architecture. It has been trained on the actual MS Marco dataset. (Ma et al., 2023)

3.3 Ensemble Approach

After getting similarity scores by both lexical (BM25) and semantic (DistilBERT or Llama) methods, we can ensemble their scores in multiple ways. One way is to use a weighted average which is tuned on a validation set. We decided to use this approach due to it's simplicity. The overall architecture of the ensemble approach is shown in figure 3

4 Experiments and Discussion

4.1 Datasets

In our project, we utilized two datasets, Scifact and Tiny MS Marco, to evaluate the performance of our retrieval system by combining lexical and semantic techniques.

4.1.1 Scifact Dataset

The Scifact dataset is specifically designed for the retrieval and validation of scientific literature. It includes a collection of scientific documents and queries that focus on understanding and validating scientific claims. This dataset is particularly valuable for testing our retrieval system's ability to handle complex, domain-specific language and concepts. The detailed and precise nature of the queries in Scifact requires our system to accurately retrieve relevant scientific documents to either support or refute specific claims. By using Scifact, we hope to demonstrate the system's capability to navigate the nuanced and technical language of scientific research, thus validating its utility in academic and professional research contexts.

4.1.2 Tiny MS Marco Dataset

Tiny MS Marco is a subset of the larger MS Marco dataset, which originally contains 8.8 million documents. For our experiments, we sampled 500,000 documents from this corpus. This dataset includes real-world search queries sourced from Bing, making it highly relevant for practical search engine applications. The queries in Tiny MS Marco are diverse, covering a wide range of topics from simple factual questions to more complex informational needs. This variety makes it an excellent choice

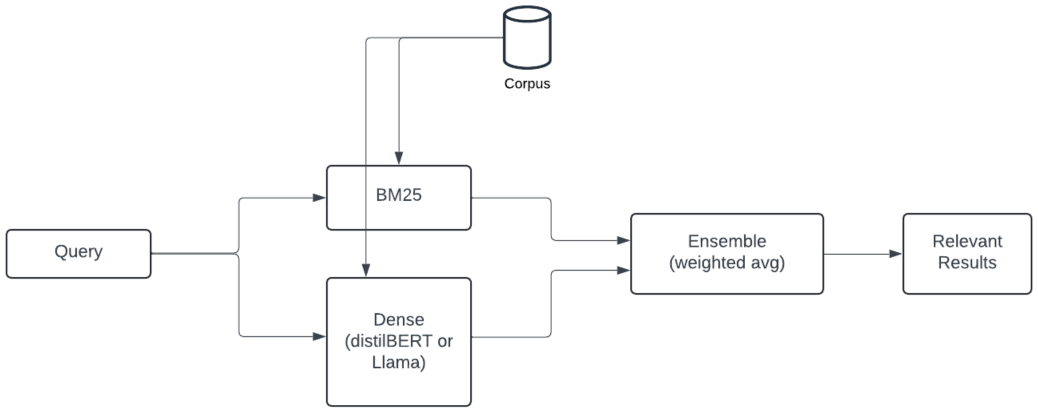

Figure 3: Ensemble approach architecture

for testing the robustness and adaptability of our retrieval system across different types of user inquiries. The real-world nature of the queries helps in assessing the practical effectiveness of our approach in everyday search scenarios, ensuring that the system can handle various levels of query complexity and deliver relevant results efficiently.

By using these datasets, we aim to demonstrate the effectiveness of combining lexical and semantic retrieval methods in enhancing search accuracy and relevance. Scifact helps us focus on domain-specific retrieval performance, while Tiny MS Marco provides a broad spectrum of query types, ensuring our system's applicability in real-world search environments.

4.1.3 Metrics

In our project, we utilized several evaluation metrics to measure the performance of our retrieval system. These metrics include nDCG@10, MAP@10, Recall@10, and P@10. Each of these metrics provides a different perspective on how well our system retrieves and ranks relevant documents.

Normalized Discounted Cumulative Gain (nDCG@10):

nDCG@10 measures the quality of the ranking of the retrieved documents by considering the position of each relevant document in the ranking. The gain is discounted logarithmically, meaning that documents appearing earlier in the list receive higher relevance scores. This metric provides a normalized score between 0 and 1, where 1 indicates a perfect ranking; this is crucial because it rewards methods that place relevant documents higher in the search results, which is essential for user satisfaction. Users typically look at the top results first, so having relevant documents appear early in the ranking can significantly improve the user's search experience. We can compute nDCG_p as

$$\mathrm{nDCG}_p = \frac{\mathrm{DCG}_p}{\mathrm{IDCG}_p},$$

$$\mathrm{DCG}_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)},$$

$$\mathrm{IDCG}_p = \sum_{i=1}^{|REL_p|} \frac{2^{rel_i} - 1}{\log_2(i+1)},$$

where rel_i is the graded relevance of the result at position i and REL_p represents the list of relevant documents (ordered by their relevance) in the corpus up to position p.

Mean Average Precision (MAP@10):

MAP@10 averages the precision scores at each point a relevant document is retrieved, up to the 10th position. This metric penalizes models that fail to place true positives at the top of the ranking. It provides a single score that quantifies the goodness of the sorting based on the score function. MAP@10 is valuable for assessing the overall precision of the retrieval method across multiple queries. It highlights the importance of retrieving relevant documents early in the ranking, ensuring that users find useful information quickly. We can compute MAP_n as

$$\mathrm{MAP}_n = \frac{1}{n}\sum_{i=1}^{n} \mathrm{AP}_i,$$

$$\mathrm{AP}_n = \frac{1}{GTP}\sum_{i=1}^{n} \mathrm{P}_i \cdot rel_i,$$

where rel_i is the graded relevance of the result at position i, P_i is the precision at position i, and GTP is is number of ground truth positives.

Recall (Recall@10):

Recall@10 measures the proportion of relevant documents retrieved within the top 10 results out of all relevant documents available. This metric gives an indication of the system's ability to capture a significant portion of the relevant documents. Recall@10 is important because it indicates the coverage of the relevant documents within the top results. Ensuring that a significant portion of relevant documents is retrieved helps in providing comprehensive search results to the user, even if not all relevant documents are highly ranked. We can compute R_n as

$$\mathrm{R}_n = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|},$$

where we should have $|\{\text{retrieved documents}\}| = n$.

Precision (P@10):

P@10 calculates the proportion of relevant documents among the top 10 retrieved documents. It focuses on the accuracy of the top results, providing a straightforward measure of how many of the top documents are relevant. This metric is particularly important for user experience, as users typically review only the top results. High precision in the top 10 results ensures that users are more likely to find the information they are looking for quickly, improving their satisfaction with the search system. We can compute P_n as

$$\mathrm{P}_n = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|},$$

where we should have $|\{\text{retrieved documents}\}| = n$.

4.2 Results

The experimental results are presented in Tables 1 and 2, which detail the performance metrics for the Scifact and MS Marco Tiny datasets, respectively. These results highlight the effectiveness of combining lexical and semantic retrieval methods to enhance search accuracy.

Performance on Scifact Dataset:

The Scifact dataset evaluation underscores the superior performance of ensemble methods over both individual lexical and semantic approaches. Notably, the ensemble of distilBERT_bm and BM25 achieved an nDCG@10 score of 0.8109, surpassing the standalone BM25 (0.67764) and distilBERT_bm (0.74305). Similarly, the ensemble method also demonstrated improvements in MAP@10, Recall@10, and P@10, with scores of 0.7814, 0.8885, and 0.0987, respectively, compared to their individual counterparts.

RepLlama, despite not being fine-tuned on Scifact, exhibited strong performance with an nDCG@10 of 0.75995. However, its combination with BM25 further elevated the performance, resulting in an nDCG@10 of 0.7647 and MAP@10 of 0.7185. This indicates the robustness of the ensemble approach in leveraging complementary strengths from both retrieval methods

Performance on MS Marco Tiny Dataset:

In the MS Marco Tiny dataset, the ensemble approaches consistently outperformed their individual counterparts. The combination of distilBERT_r and BM25 yielded an nDCG@10 of 0.5963 and MAP@10 of 0.1647, compared to 0.48734 and 0.13795 for BM25 alone. DistilBERT_bm and BM25 ensemble also showed significant improvement, with an nDCG@10 of 0.574 and MAP@10 of 0.1618.

RepLlama, when combined with BM25, achieved an nDCG@10 of 0.6922 and MAP@10 of 0.2026, demonstrating a notable enhancement over the individual performance of RepLlama with an nDCG@10 of 0.70889 and MAP@10 of 0.21312. This reinforces the value of ensemble methods in providing a more balanced and contextually rich retrieval experience.

4.3 Conclusion

In this paper, we explored the potential of combining lexical and semantic retrieval techniques to enhance the performance of information re-

	Scifact			
	nDCG@10	**MAP@10**	**Recall@10**	**P@10**
Lexical				
BM25	0.67764	0.63245	0.80044	0.089
Semantic				
distilBERT_r	0.63538	0.58474	0.77967	0.08733
distilBERT_bm	0.74305	0.71684	0.80894	0.08933
RepLlama	0.75995	0.71105	0.89867	0.10167
Lexical + Semantic				
distilBERT_r + bm25	0.7391	0.6932	0.8681	0.0967
distilBERT_bm + bm25	0.8109	0.7814	0.8885	0.0987
RepLlama + bm25	0.7647	0.7185	0.8937	0.1013

Table 1: Performance comparison of different models on Scifact dataset

	MS Marco Tiny			
	nDCG@10	**MAP@10**	**Recall@10**	**P@10**
Lexical				
BM25	0.48734	0.13795	0.16663	0.54074
Semantic				
distilBERT_r	0.55349	0.15388	0.18187	0.60556
distilBERT_bm	0.4875	0.13311	0.15667	0.53889
RepLlama	0.70889	0.21312	0.24466	0.76296
Lexical + Semantic				
distilBERT_r + bm25	0.5963	0.1647	0.19	0.663
distilBERT_bm + bm25	0.574	0.1618	0.192	0.6278
RepLlama + bm25	0.6922	0.2026	0.2351	0.7389

Table 2: Performance comparison of different models on MS Marco Tiny dataset

trieval systems. Our experiments, conducted on the Scifact and MS Marco Tiny datasets, demonstrate that ensemble methods integrating BM25 with advanced language models such as distilBERT and RepLlama significantly outperform their standalone counterparts.

The results show that ensemble approaches, particularly the combination of distilBERT_bm and BM25, consistently achieve higher nDCG@10, MAP@10, Recall@10, and P@10 scores compared to individual lexical and semantic methods. This indicates a balanced improvement in both precision and contextual understanding, addressing the limitations of traditional IR methods.

Notably, RepLlama, despite not being fine-tuned on the Scifact dataset, benefited from the ensemble approach, highlighting that even in zero-shot scenarios, where fine-tuning data and resources are unavailable, combining lexical and semantic methods can still enhance retrieval performance without additional fine-tuning.

These findings underscore the effectiveness of ensemble retrieval methods in creating more accurate and contextually aware search systems. The integration of lexical precision with semantic context understanding not only boosts retrieval metrics but also paves the way for more sophisticated and reliable IR systems in diverse application domains.

Future work could explore further optimization of ensemble weights and the integration of additional retrieval techniques to continue improving search accuracy and efficiency. Additionally, extending this approach to other datasets and real-world scenarios could provide deeper insights into its applicability and robustness across different contexts.

References

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021. Complementing lexical retrieval with semantic residual embedding.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun hsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval.

Bhaskar Mitra and Nick Craswell. 2017. Neural models for information retrieval.

Advait Parulekar, Liam Collins, Karthikeyan Shanmugam, Aryan Mokhtari, and Sanjay Shakkottai. 2023. Infonce loss provably learns cluster-preserving representations.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives.