Adapting General-Purpose Embedding Models to Private Datasets Using Keyword-based Retrieval

Anonymous ACL submission

Abstract

Text embedding models play a cornerstone role in AI applications, such as retrieval-augmented generation (RAG). While general-purpose text embedding models demonstrate strong performance on generic retrieval benchmarks, their effectiveness diminishes when applied to private datasets (e.g., company-specific proprietary data), which often contain specialized terminology and lingo. In this work, we introduce BMEmbed, a novel method for adapting general-purpose text embedding models to private datasets. By leveraging the well-established keyword-based retrieval technique (BM25), we construct supervisory signals from the ranking of keyword-based retrieval results to facilitate model adaptation. We evaluate BMEmbed across a range of domains, datasets, and models, showing consistent improvements in retrieval performance. Moreover, we provide empirical insights into how BM25-based signals contribute to improving embeddings by fostering alignment and uniformity, highlighting the value of this approach in adapting models to domain-specific data. We release the source code[1] for the research community.

1 Introduction

Text embeddings serve as a cornerstone for various AI applications, particularly in information retrieval and retrieval-augmented generation (RAG) systems (Izacard et al., 2022; Gao et al., 2023). With the widespread adoption of AI, companies like OpenAI and Cohere now provide general-purpose text embedding APIs, enabling organizations to quickly integrate AI into their RAG systems. However, while these general-purpose embedding models show impressive performance on generic benchmarks, they often face significant challenges when applied to private datasets, such as domain-specific or company-specific proprietary data, which often contain specialized terminology and jargon (Anderson et al., 2024; Tang and Yang, 2024a).

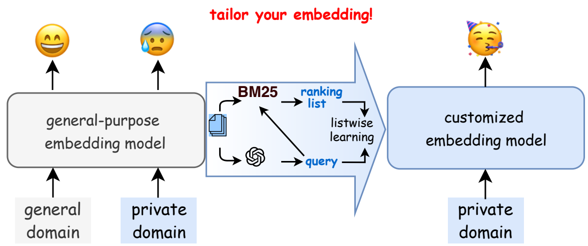

Figure 1: An illustration of tailoring an embedding model to a private domain.

For instance, consider a pharmaceutical company that seeks to build a RAG system over its vast internal dataset. The company's employees may query the system for information about an internal product code (e.g., Product Code: PHX-121). However, general-purpose models, not trained on this proprietary dataset, may fail to properly interpret or retrieve relevant documents containing such specific terms, leading to suboptimal answers.

Current practices in RAG systems often attempt to address this issue by combining traditional keyword-based retrieval with embedding-based retrieval. One popular hybrid approach is reciprocal rank fusion (RRF), which reranks results based on a mathematical formula without fine-tuning the underlying embedding model (Cormack et al., 2009). While simple and effective, RRF remains heuristic, with its effectiveness potentially limited by the lack of fine-tuning to the private dataset. This leads us to the following question: *Can we fine-tune general-purpose embedding models to better align with private datasets?*

One of the key challenges in adapting embedding models to domain-specific datasets is the lack of available tuning signals. While general-purpose embedding models are often trained on large, curated QA datasets using contrastive learning (Tan

[1]The code is available at: `https://anonymous.4open.science/r/BMEmbed-2031`.

et al., 2022; Zhou et al., 2022; Moreira et al., 2024), private datasets, which often consist of free-text data without annotations, pose a particular challenge. This leads to an important sub-question: *How can we generate supervisory signals for adapting general-purpose embedding models to private, unlabeled datasets?*

In this work, we introduce BMEmbed, an automated framework designed to adapt general-purpose text embedding models to private datasets. Our method leverages BM25 (Robertson and Zaragoza, 2009), a well-established keyword-based retrieval function based on TF-IDF, to generate supervisory signals from the ranking of keyword-based retrieval results. The BMEmbed framework consists of three main components: (1) domain query generation, where a large language model generates synthetic queries based on domain-specific events extracted from the private corpus; (2) relevant sampling, which uses BM25 to retrieve lexically related paragraphs and samples from different intervals of the ranking list to ensure informative signals; and (3) listwise fine-tuning, where the embedding model is optimized using a listwise loss function on the curated ranking lists, fully leveraging the ranking supervision. Unlike traditional in-batch negative contrastive learning (van den Oord et al., 2018; Chen et al., 2020), our approach uses ranked BM25 results to guide the fine-tuning process.

We evaluate BMEmbed across multiple domains and datasets, using two general-purpose embedding models with varying scales. Compared to base embedding models, BMEmbed consistently achieves substantial improvements in retrieval accuracy. Our experiments further show that BMEmbed outperforms or achieves competitive performance compared to two commonly used techniques in current RAG systems: (1) fine-tuning with in-batch negative contrastive learning, and (2) the RRF hybrid approach. To better understand the inner workings of BMEmbed, we investigate the alignment and uniformity properties of the adapted embeddings (Wang and Isola, 2020). We find that BMEmbed successfully improves embedding uniformity while maintaining good alignment, leading to improved retrieval performance.

In summary, this paper introduces a simple yet effective method for adapting general-purpose text embedding models to private datasets. Given the increasing adoption of RAG systems across industries, we believe our method provides a practical solution to enhance domain specificity, leading to more accurate and contextually relevant retrieval results in real-world applications.

2 Background

2.1 Text Embedding Models

Text embedding refers to the numerical representation of a piece of text that captures its semantic meaning, transforming texts of varying lengths into fixed-size vectors. Previously, fine-tuning models like BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020) to adapt to embedding downstream tasks was the dominant approach (Reimers and Gurevych, 2019; Ni et al., 2022). However, with the development of LLMs, the landscape is shifting. The focus has now moved toward building LLM-based, general-purpose embedding models, including Qwen (Li et al., 2023), LLM2Vec (BehnamGhader et al., 2024), NV-Embed (Lee et al., 2024), etc. These LLM-based embedding models have demonstrated their superiority on massive text datasets, e.g., MTEB (Muennighoff et al., 2023).

Current embedding models (Izacard et al., 2022; Wang et al., 2022; Li et al., 2023; Chen et al., 2024) are primarily trained using contrastive learning, with the widely adopted InfoNCE loss(van den Oord et al., 2018) as the objective, which aims to distinguish semantically relevant text pairs from irrelevant ones. While effective, the performance of contrastive learning heavily depends on the selection of high-quality positive and negative samples (Tan et al., 2022; Zhou et al., 2022; Moreira et al., 2024). When adapting the embedding model to a specific domain, constructing relevant and irrelevant samples from a private corpus can be a challenging task. In this work, we propose leveraging BM25 to construct lexically relevant samples, addressing the challenge of sample selection in an unsupervised manner.

2.2 Keyword-based Retrieval: BM25

BM25 (Robertson and Zaragoza, 2009)is a well-established retrieval method based on TF-IDF, which ranks documents by considering the uniqueness and significance of terms relevant to a given query. The BM25 score for document d with respect to query q is defined as:

$$\text{BM25}(d, q) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{f(t, d) \cdot (k_1 + 1)}{f(t, d) + k_1 \cdot \left(1 - b + b \cdot |\hat{d}|\right)}$$

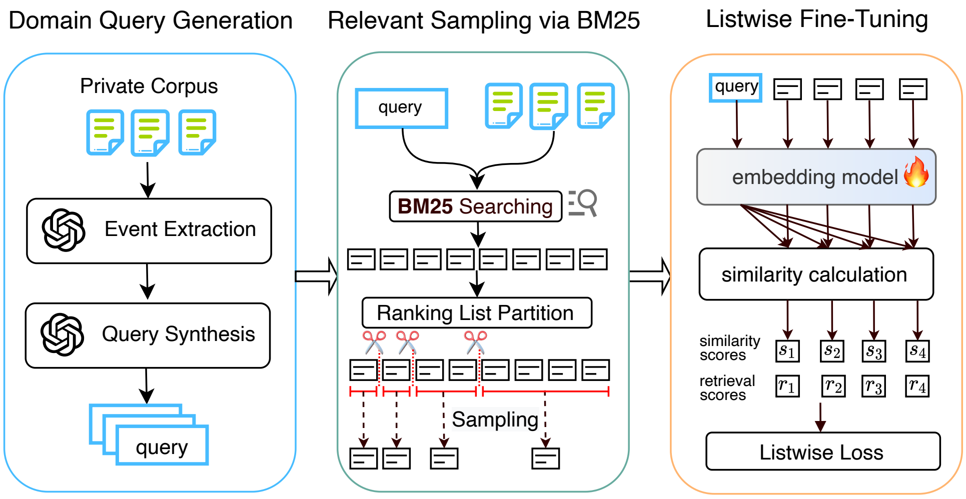

Figure 2: An overview of the BMEmbed framework.

where $f(t, d)$ is the term frequency of term t in document d, $|\hat{d}|$ is the normalization of document length, $\sum_{t \in q} \mathrm{IDF}(t)$ is the inverse document frequency of term t in the corpus, k_1 and b are hyper parameters that control the impact of term frequency and document length, respectively. Previous works have demonstrated the effectiveness of using BM25 as a weak supervision signal for training small models (Dehghani et al., 2017; Haddad and Ghosh, 2019; Karpukhin et al., 2020).

Despite significant progress in dense retrieval (Karpukhin et al., 2020; Xin et al., 2022), BM25 remains a robust retrieval algorithm. Its rule-based, keyword matching approach enables strong generalization, maintaining competitive performance in scenarios where keyword matching is more crucial than semantic matching. As a result, hybrid approaches, such as Reciprocal Rank Fusion (RRF) (Cormack et al., 2009), have been used to combine and rerank results from both dense retrieval models (embedding-based) and sparse retrieval models (BM25-based). However, RRF relies on heuristics to rank these hybrid results. In contrast, this paper aims to fine-tune general-purpose embedding models to a specific dataset, enabling true adaptation rather than simply combining results from different retrieval methods.

3 BMEmbed: Domain Adaptation for General-Purpose Embeddings

In this section, we present BMEmbed, an automated framework designed to tailor general-purpose embedding models to private datasets consisting of unannotated text. The method contains three steps, and the overall process is illustrated in Figure 2.

3.1 Domain Query Generation

The first step is to prompt an LLM (e.g., GPT-4) to generate synthetic queries focused on domain-specific events in the private corpus, rather than on general concepts.

Event Extraction We require the LLM to extract all the events and their associated arguments from the private corpus. In addition, the original context from which the events are extracted is also generated, serving as the evidence for the queries used in the baseline method in subsequent experiments.

Query Synthesis Then, we feed both the corpus and the extracted events into the LLM, prompting it to automatically generate queries Q for each event. The detailed prompts are provided in Appendix A.

3.2 Relevant Sampling via BM25

The second step is to construct ranked retrieval results using keyword retrieval method BM25.

BM25 Searching We divide the private corpus into multiple chunks and calculate the BM25 score between query $q \in Q$ and each chunk. The top-k scoring chunks, denoted as $C = \{c_1, c_2, \ldots, c_k\}$, are selected, where each chunk c_i is associated with its respective BM25 score r_i.

Ranking List Partition We further partition C into m intervals, denoted as $\{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_m\}$. This approach allows positives and negatives to be sampled from different intervals, which amplifies the scope of sampling space across diverse relevance tiers, effectively mitigating noise in BM25 pseudo labels. The partitioning can follow either a uniform or a fine-to-coarse strategy. Uniform intervals divide the range of BM25 scores into equally sized segments, ensuring a consistent distribution of samples across all intervals. In contrast, fine-to-coarse partitioning strategy intervals prioritize finer segmentation of higher-relevance scores, leading to more granular sampling for positively ranked examples. For instance, given $m = 4$, the top-20 ranking list can be divided into intervals $[0, 2), [2, 6), [6, 12), [12, 20)$ using the fine-to-coarse strategy, whereas the uniform strategy divides it into $[0, 5), [5, 10), [10, 15), [15, 20)$.

Ranking-Based Sampling For each interval $\mathcal{P}_j$, we randomly select one sample p_j along with its retrieval score r_j, forming a ranking list $[q, p_1, p_2, \ldots, p_m, r_1, r_2, \ldots, r_m]$.

3.3 Listwise Fine-Tuning

Since BM25 retrieval results produce a ranked list, we hypothesize that this ranking contains valuable information that can be better utilized through a listwise training objective, rather than the commonly used in-batch negative contrastive learning objective, where ranking information is typically ignored. To this end, we employ a listwise training objective to fully leverage the ranking information obtained from BM25 retrieval.

Given $[q, p_1, p_2, \ldots, p_m, r_1, r_2, \ldots, r_m]$ and a base embedding model $e(\cdot)$, we first obtain the embeddings of q and p_j for $j \in [1, \ldots, m]$, denoted as $e(q)$ and $e(p_j)$, respectively. Then, we calculate the cosine similarity $s_j = \mathrm{sim}(e(q), e(p_j))$. Following the work of ListNet (Cao et al., 2007), the listwise loss is calculated as follows:

$$\mathcal{L}(s, r) = -\sum_{q \in Q} \sum_{j=1}^{m} p_j^r \log(p_j^s)$$

where r=$\{r_1, r_2, \ldots, r_m\}$, s=$\{s_1, s_2, \ldots, s_m\}$, p^r and p^s are the distributions normalized by softmax over the r and s, respectively. We introduce a temperature scaling factor α on the target score list r, with:

$$p_j^r = \frac{\exp\left(\frac{r_j}{\alpha}\right)}{\sum_{i=1}^{m} \exp\left(\frac{r_i}{\alpha}\right)}$$

Here, α controls the sharpness of the target distribution, with smaller values leading to a more concentrated distribution, and larger values resulting in a smoother distribution.

Dataset	Multihop	Finance	LegalBench
evaluation queries	2,255	498	1,676
corpus tokens	1,453k	840k	7,109k
synthesized queries	5,972	1,009	685
chunk size	256	1,024	1,024
k	1,000	1,000	4,000
m	9	6	6

Table 1: Statistics and implementation details of the datasets.

4 How does BMEmbed Perform?

4.1 Experimental Setup

Base Embedding Models We use the following two general-purpose embedding models: gte-Qwen2-1.5B-instruct[2], a small yet strong model, and e5-mistral-7B-instruct[3], a larger model based on Mistral-7B. Both two models perform competitively on the MTEB leaderboard (Muennighoff et al., 2023).

Baselines We compare models fine-tuned by BMEmbed with the following methods: 1) **BM25**, with parameters k_1=1.2 and b=0.75; 2) **Base**, the base embedding model. 3) **CL**, the embedding model fine-tuned using contrastive objective InfoNCE loss (van den Oord et al., 2018), where LLM-generated evidence is used as positives (as detailed in Section 3.1), along with in-batch negatives. 4) **RRF**, Reciprocal Rank Fusion (Cormack et al., 2009), which is a hybrid search method combining rankings from multiple sources into a unified ranking:

$$RRF(d) = \sum_{a \in A} \frac{1}{u + a(d)}$$

where d is a document, A is the set of rankers (retrievers), $a(d)$ is the rank of document d in ranker a, and u is a constant set to 40. Here we combine BM25 rankings with the base embedding model. 5) **RRF+BMEmbed**, the combination of the BM25 and the BMEmbed-finetuned model.

"Private" Datasets In our experiments, we choose three publicly available retrieval datasets as evaluation benchmarks. However, these datasets are released after the base embedding models, meaning the models are unlikely to have been trained on them. Therefore, while the datasets are

[2]https://huggingface.co/Alibaba-NLP/gte-Qwen2-1.5B-instruct

[3]https://huggingface.co/intfloat/e5-mistral-7b-instruct

Method	Multihop-RAG			Finance-RAG			LegalBench-RAG		
	Hit@10	Hit@4	MAP@10	Hit@10	Hit@4	MAP@10	Hit@10	Hit@4	MAP@10
BM25	79.02	65.01	25.93	57.43	46.18	37.46	14.62	7.58	1.62
Qwen2-1.5B									
Base	76.50	59.69	22.22	53.82	41.37	32.84	23.09	16.65	6.34
CL	74.72	55.96	21.48	58.43	43.57	35.20	25.48	17.90	5.45
BMEmbed	83.06	68.34	26.54	57.03	45.38	36.21	**28.52**	**20.64**	**7.47**
RRF	82.04	66.30	25.80	63.45	49.80	40.97	24.76	18.32	6.45
RRF+**BMEmbed**	**84.35**	**71.09**	**28.30**	**64.46**	**51.61**	**41.62**	28.46	19.69	7.19
e5-mistral-7B									
Base	75.39	54.99	20.33	48.80	36.55	28.10	23.75	17.42	6.48
CL	69.40	48.34	16.67	57.43	46.79	35.08	21.06	16.65	5.37
BMEmbed	85.63	**71.49**	27.60	62.25	48.39	38.40	**27.27**	19.03	7.08
RRF	82.13	67.58	27.04	61.85	47.39	39.55	24.34	**19.09**	**7.23**
RRF+**BMEmbed**	**85.72**	71.44	**28.36**	**64.06**	**52.21**	**41.92**	**27.27**	19.03	7.08

Table 2: Retrieval performance of different methods across three datasets. **Best** results are highlighted for each embedding model on each dataset.

publicly available, they effectively simulate "private" datasets in our experiments, also ensuring fair comparison and reproducibility.

Specifically, the three datasets are: Multihop-RAG (Tang and Yang, 2024b), a multi-hop question answering (QA) dataset from the financial news domain; Finance-RAG[4], a long-context QA dataset based on financial reports, released as part of the ACM-ICAIF'24 FinanceRAG competition; and LegalBench-RAG (Pipitone and Alami, 2024), a challenging long-context legal domain QA dataset. Each dataset contains questions, their corresponding relevant evidence, and the original corpus. We use the evidence as the label to evaluate the retrieval performance. Detailed statistics are provided in Table 1.

Implementation and Training Details For domain query generation, we use GPT-4o for accurate event extraction and GPT-4o-mini for query synthesis to minimize costs. We generate 5,972, 1,009, and 685 queries for Multihop-RAG, Finance-Bench, and Legal-Bench, respectively, based on corpus size. A real case, including the input corpus, intermediate events, and the final generated query, is showcased in Appendix B. During relevant sampling, we set the chunk size of 256 for Multihop-RAG and 1,024 for the other two datasets with long context. The fine-to-coarse partitioning strategy is used by default. We adopt m=9 for Multihop-RAG and m=6 for the others, with k=1,000 for MultiHop-RAG and Finance-RAG, and k=4,000 for LegalBench-RAG. The impact of different m and partitioning strategies is further discussed in Section 5.2. The results under different k are shown in Appendix C. For finetuning, we use a fixed batch size of 16 for CL, while the batch size is equivalent to m for BMEmbed. The temperature α is set to a moderate value between 1.0 to 5.0, with further adjustments on different datasets and models, which we provide a detailed discussion in Section 5.3. We finetune the model using LoRA (Hu et al., 2022) with a rank of 16 for 1,000 steps. Training Qwen on 4×3090 GPUs takes about 1.5 hours, while training e5-mistral on 8×H800 GPUs takes approximately one hour.

[4] https://www.kaggle.com/competitions/icaif-24-finance-rag-challenge

4.2 Results and Discussion

Table 2 presents the experimental results of BMEmbed and all baselines across two embedding models and three datasets. It can be observed:

1) The vanilla embedding models perform sub-optimally in specific domains. In most cases, base models underperform BM25 on Multihop-RAG and Finance-RAG, even with large model sizes. This finding highlights the necessity of further adaptation when applying general-purpose embedding models to specific domains.

2) Contrastive learning does not consistently lead to performance improvements for embedding model adaptation. Surprisingly, we find that applying CL to base models do not always improve performance. We hypothesize that noise in the positive evidence generated by the LLM might interfere with model optimization. This indicates that contrastive learning is sensitive to the quality of positive and negative samples, and such an approach does not always result in promising improvements

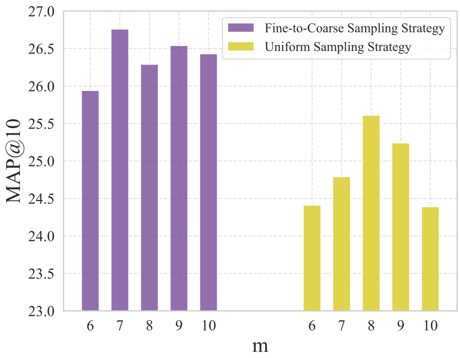

Figure 3: Retrieval performance of MAP@10 for different m and sampling strategies.

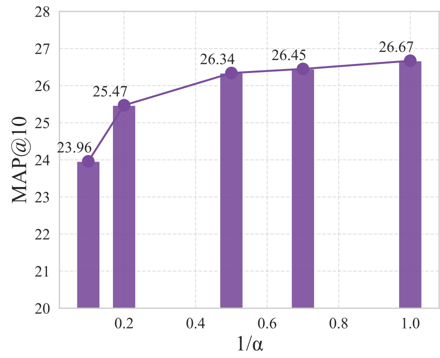

Figure 5: Retrieval performance of MAP@10 for different α.

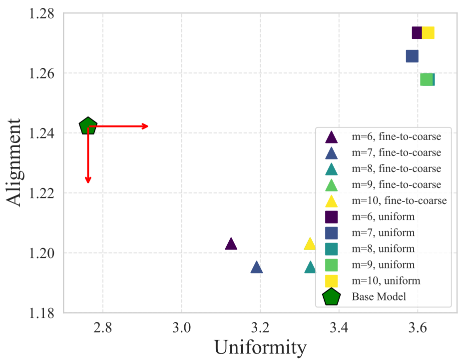

Figure 4: Alignment and uniformity for different m and sampling strategies.

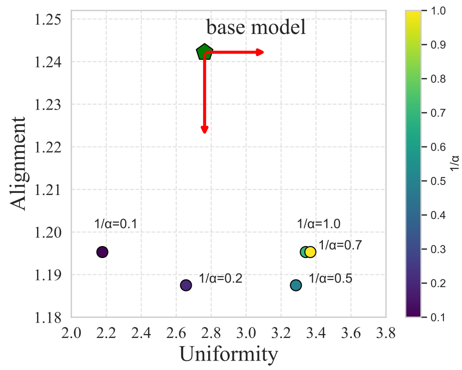

Figure 6: Alignment and uniformity for different α.

for embedding adaptation.

3) Our BMEmbed consistently delivers improvements, benefiting from the supervision signals provided by BM25. Our framework boosts the base models across all embedding models and datasets, especially on the metrics Hit@4. Compared to RRF which combines BM25 ranking information with dense retrieval from embedding models, BMEmbed achieves a remarkable improvement, which illustrates that our framework deeply deciphers the ranking confidence signals from BM25, achieving a better embedding model adaptation.

4) Furthermore, BMEmbed can be combined with other hybrid retrieval methods to achieve further enhancement. This is demonstrated in experiments comparing RRF+BMEmbed with RRF alone. In most cases, RRF+BMEmbed shows clear performance gains, except in the case of LegalBench-RAG, where the BM25 baseline performs poorly and BMEmbed+RRF does not achieve further performance gains.

5 Why BMEmbed Enhances Embedding Adaptation? An Investigation from Uniformity and Alignment

In this section, we further investigate why BMEmbed leads to improvements. We conduct ablation experiments to study how our samplers and temperature interact with retrieval performance. Moreover, we introduce the **Alignment** and **Uniformity** properties, which reflect the quality of the embedding, to gain a deeper theoretical understanding. The reported experiments are based on the Multihop-RAG dataset and the Qwen2-1.5B model by default. The complete ablation study setup and results are presented in Appendix C. As observed in the ablation study, our experiments empirically reveal **a strong agreement between embedding properties and retrieval performance, suggesting that the enhancement from BMEmbed results from the optimized embedding properties.** In this section, we discuss our key observations and conclusions.

Method	Multihop-RAG		Finance-RAG		LegalBench-RAG	
	Alignment↓	Uniformity↑	Alignment↓	Uniformity↑	Alignment↓	Uniformity↑
Qwen2-1.5B						
Base	1.2422	2.7665	1.1562	1.6567	**1.3203**	1.1599
CL	1.3516	2.8022	1.2188	**2.9437**	2.0000	**2.2382**
BMEmbed	**1.2031**	**3.3266**	**1.1484**	2.6631	1.6691	2.1426
e5-mistral-7B						
Base	1.1875	1.7430	1.1797	1.0353	**1.2891**	0.7317
CL	1.5156	2.7649	1.3281	3.0445	2.7969	**1.7913**
BMEmbed	**1.1797**	**3.7768**	**1.0859**	**3.2144**	1.6797	1.6182

Table 3: Alignment and Uniformity of Embedding Models. Lower alignment (↓) and higher uniformity (↑) are preferred. **Best** results are highlighted for each embedding model on each dataset.

5.1 Alignment and Uniformity

A good embedding should bring similar data points closer together while preserving as much useful information as possible (Bachman et al., 2019; Hjelm et al., 2019) to distinguish different data points, leading to *lower alignment and higher uniformity*. Here, we adopt alignment and uniformity for evaluating an embedding following the work of Wang and Isola (2020), with further details and discussion in Appendix D.

5.2 Ablation Study of Different Partitions

To explore the effect of different partitions during relevant sampling via BM25 in BMEmbed, we investigate the impact of various partition factors, including the number of partitions and the partitioning strategies. Specifically, we conduct experiments with m ranging from 6 to 10, using both uniform and fine-to-coarse sampling strategies, with the temperature α set to 1 and k set to 1,000.

Figure 3 shows the relationship between retrieval metrics MAP@10 and fine-tuning with different m and sampling strategies, while Figure 4 presents a comparison of uniformity and alignment of the fine-tuning models shown in previous figure. We observe that **the fine-to-coarse strategy achieves better retrieval performance and superior alignment compared to the uniform strategy**. In contrast, the uniform strategy is suboptimal in retrieval performance due to its overly uniform embedding distribution, which leads to a loss of alignment. In addition, as m increases from 6 to 7 under the fine-to-coarse sampling strategy, we observe a measurable improvement in MAP@10 performance, suggesting that moderately expanding the sampling scope captures more relevant items. However, further increasing m causes performance fluctuations and a gradual decline in overall effectiveness. These findings highlight the importance of carefully calibrating m to optimize retrieval performance.

5.3 Ablation Study of Listwise Fine-Tuning with Varying Temperatures

We examine the effect of varying temperatures α. For convenience, we work with its reciprocal, $1/\alpha$, with values of 0.1, 0.2, 0.5, 0.7, and 1.0. We set k=500, m=10, and adopt the fine-to-coarse sampling strategy.

Figure 5 shows the trend between MAP@10 and fine-tuning with different $1/\alpha$, with the corresponding alignment and uniformity results shown in Figure 6. Our analysis shows that **smaller temperature achieve better retrieval performance by fostering good uniformity in the embedding distribution.** In contrast, as temperature increases, uniformity decreases, even lowering it compared to the base model. This is because the higher temperature smooths the label distribution, which diminishes the distinction between learning samples and causes the embeddings to become overly clustered. Such clustering may hurt the performance of downstream tasks which require clear distinction between embeddings, as observed in our experiments, where it led to a degradation in retrieval performance.

5.4 BMEmbed Balances Alignment and Uniformity Optimization

Our ablation experiment and analysis have demonstrated that using the fine-to-coarse strategy with a smaller temperature is an effective way to leverage BM25, supported by both theoretical reasoning and practical results. Since main experiment we conducted in Section 4.2 is based on this strategy,

Original Query	Query with Masked Keywords
Does "The New York Times" article attribute the success of the Buffalo Bills' **defense** to the contributions of Jordan Poyer, while the "Sporting News" article suggests that the Baltimore **Ravens**' defense needs to improve before their game against the Cincinnati **Bengals**?	Does "The New York Times" article attribute the success of the Buffalo Bills' `[MASK]` to the contributions of Jordan Poyer, while the "Sporting News" article suggests that the Baltimore `[MASK]`' defense needs to improve before their game against the Cincinnati `[MASK]`?

Table 4: An example of a comparison between original query and masked query example.

Model	Mask	Hit@4	Hit@10	MAP@10
Base	✓	22.13 (↓37.56)	33.17 (↓43.33)	6.46 (↓15.76)
	×	59.69	76.50	22.22
BMEmbed	✓	23.41 (↓**44.93**)	34.41 (↓**48.65**)	7.02 (↓**19.44**)
	×	68.34	83.06	26.54

Table 5: A comparative experiment involving query masking between two models.

here we report the uniformity and alignment of corresponding fine-tuned embedding models in Table 3 for further analysis.

Embedding models fine-tuned with BMEmbed achieve better retrieval results due to increased uniformity compared to the base model, while maintaining relatively low alignment. Comparing with CL with in-batch negatives, we observe that although uniformity has increased significantly, it does not effectively maintain or improve the alignment of the base model. This imbalance leads to instability in retrieval performance, and in some cases, even performance degradation. Specifically, we identify the ideal optimization direction, as indicated by the red arrow in the in Figure 4. BMEmbed achieves this theoretical direction on both Multihop-RAG and Finance-RAG, demonstrating its potential to balance the optimization of both uniformity and alignment.

6 How Does BM25 Boost Embedding? A Masked Keyword Analysis of Pattern Utilization

To investigate how BM25-driven signals boost embedding adaptation, we conduct an interesting masked keyword study on queries. Specifically, we compare retrieval performance between the base Qwen2-1.5b model and its fine-tuned variant (BMEmbed) under two conditions: 1) original queries and 2) keyword-masked queries. The experimental pipeline involves three steps: First, we employ LLMs to extract domain-specific keywords from each query. The prompts are detailed in Appendix E. Next, we mask these identified keywords (see Table 4 for examples) while preserving syntactic structure. The number of query pairs constructed is 2,255. Finally, we evaluate both models' performance degradation when processing masked versus original queries.

As shown in Table 5, we compare the retrieval performance between BMEmbed and the base model under keyword-masked conditions. The experimental results reveal that masking critical keywords eliminates BMEmbed's performance advantage, reducing both models to comparable accuracy levels. This demonstrates that the fine-tuning improvements primarily stem from the model's enhanced focus on domain-specific keywords, consistent with our hypothesis. Furthermore, these findings prove that our list-wise fine-tuning strategy successfully enables BMEmbed to internalize private domain features through BM25-guided pattern learning.

7 Conclusion

With the growing adoption of AI in real-world applications, particularly RAG systems, adapting general-purpose models to domain-specific data remains a critical challenge. In this paper, we present BMEmbed, a novel method for adapting text embedding models to private datasets (e.g., company-specific proprietary data). Since private datasets often contain specialized terminology and domain-specific language, we leverage keyword-based retrieval as a supervisory signal to fine-tune general-purpose embedding models. Experimental results demonstrate that BMEmbed effectively enhances retrieval performance, producing more accurate query results on private datasets. As AI continues to transform industries, we hope that our proposed method can further advance the adoption and adaptation of AI in domain-specific applications, ensuring more effective and contextually relevant retrieval.

8 Limitations

This study has several limitations that present opportunities for future research. First, our current method primarily focuses on the retrieval task in embedding models. However, text embeddings are also widely used in domain-specific NLP tasks such as clustering and semantic textual similarity (STS). An interesting direction for future research is exploring task-specific supervisory signals to better adapt general-purpose embedding models to private datasets for applications beyond retrieval, including clustering and STS. Second, while our method aims to develop embedding models tailored to private datasets (such as company-specific proprietary data), we evaluate it on public datasets. These datasets are chosen because they are released after the base embedding models we assess, ensuring fair comparison and public reproducibility. However, applying this method to proprietary datasets in real-world RAG scenarios remains an important next step. We hope future research will explore these practical applications to further validate and refine our approach.

References

Peter Anderson, Mano Vikash Janardhanan, Jason He, Wei Cheng, and Charlie Flanagan. 2024. Greenback bears and fiscal hawks: Finance is a jungle and text embeddings must adapt. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: EMNLP 2024 - Industry Track, Miami, Florida, USA, November 12-16, 2024*, pages 362–370. Association for Computational Linguistics.

Philip Bachman, R. Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15509–15519.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *CoRR*, abs/2404.05961.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 129–136. ACM.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 2318–2335. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 758–759. ACM.

Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural ranking models with weak supervision. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 65–74. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *CoRR*, abs/2312.10997.

Dany Haddad and Joydeep Ghosh. 2019. Learning more from less: Towards strengthening weak supervision for ad-hoc retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 857–860. ACM.

R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models. *CoRR*, abs/2405.17428.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *CoRR*, abs/2308.03281.

Gabriel de Souza P Moreira, Radek Osmulski, Mengyao Xu, Ronay Ak, Benedikt Schifferer, and Even Oldridge. 2024. Nv-retriever: Improving text embedding models with effective hard-negative mining. *arXiv preprint arXiv:2407.15831*.

Niklas Muennighoff, Nouamane Tazi, Loïc 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, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 2006–2029. Association for Computational Linguistics.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1864–1874. Association for Computational Linguistics.

Nicholas Pipitone and Ghita Houir Alami. 2024. Legalbench-rag: A benchmark for retrieval-augmented generation in the legal domain. *CoRR*, abs/2408.10343.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Haochen Tan, Wei Shao, Han Wu, Ke Yang, and Linqi Song. 2022. A sentence is worth 128 pseudo tokens: A semantic-aware contrastive learning framework for sentence embeddings. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 246–256. Association for Computational Linguistics.

Yixuan Tang and Yi Yang. 2024a. Do we need domain-specific embedding models? an empirical investigation. *CoRR*, abs/2409.18511.

Yixuan Tang and Yi Yang. 2024b. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *CoRR*, abs/2401.15391.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *CoRR*, abs/2212.03533.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9929–9939. PMLR.

Ji Xin, Chenyan Xiong, Ashwin Srinivasan, Ankita Sharma, Damien Jose, and Paul Bennett. 2022. Zero-shot dense retrieval with momentum adversarial domain invariant representations. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4008–4020. Association for Computational Linguistics.

Kun Zhou, Beichen Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Debiased contrastive learning of unsupervised sentence representations. In *Proceedings of the 60th Annual Meeting of the Association* for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 6120–6130. Association for Computational Linguistics.

A Prompts Used for Domain Query Generation

The LLM prompts used in the domain query generation stage are detailed as follows:

Event Extraction Prompt:
Given a document, please extract all the events and their associated topics and organization in the context.
Note: 1. The event should not contain ambiguous references, such as 'he',' she,' and 'it', and should use complete names.
2. You should give at least one passage in the original text associated to the event you extract, DO NOT make up any event.
3. If there are multiple paragraphs associated to the extracted event, please list and number all of them.
4. If the event does not contain some of the arguments mentioned above, please leave it empty.
5. The type of Event involves fine-grained events and general events, where fine-grained events focus on specific facts and details while general events are summarizations of happened fine-grained events.
6. Please return the fine-grained events first, then return general events.
The document is:
{doc}
Please return the extracted event in the following format with following arguments:
[Event]:
[Topic]:
[Original context]: 1. ......
2. ......
...
[Type]:
Events you extract are:

Query Synthesis Prompt:
Given several events and their original source document, please ask several questions according to the infomation and give the original reference paragraph following this format:
[Envent]:
[Question]:
Note: 1. Don't need to mention all the arguments in your question.
2. You can involve the original document information, but make sure that your question is about the topic of the given event.
3. You should ask questions separately to different events.
Document:
{doc}
Event:
{event}
Your question towards given event:

B Case Study of Query Generation

In this section, we present a real query generation process, showcasing the input document, intermediate extracted events, and the final generated query.

Document Chunk from Multihop-RAG:
Table of Contents Table of Contents Echo, Fire TV, and Kindle deals Apple deals TV deals Laptop deals Headphone and earbud deals Tablet deals Gaming deals Speaker deals Vacuum deals Kitchen deals Smart home deals Fitness deals Beauty tech deals Drone deals Camera deals Lego deals Gift card deals UPDATE: Nov. 27, 2023, 5:00 a.m. EST This post has been updated with all of the latest Cyber Monday deals available at Amazon. Amazon is dragging out the year's biggest shopping holiday(s) into 11 days of deals. The retail giant began its Black Friday sale in the early morning of Friday, Nov. 17 (a week ahead of schedule) and was on top of making the switch to Cyber Monday language in the wee hours of Saturday, Nov. 25. Official Cyber Monday mode, which is currently on through Monday, Nov. 27, includes both a ton of deals carried over from Black Friday plus some new ones.

GPT-4o Extracted Events:
Fine-Grained Events
1. [Event]: Amazon began its Black Friday sale.
[Topic]: Black Friday sale
[Original context]: "The retail giant began its Black Friday sale in the early morning of Friday, Nov. 17 (a week ahead of schedule)..."
[Type]: Fine-grained
2. [Event]: Amazon switched to Cyber Monday language.
[Topic]: Cyber Monday sale
[Original context]: "...and was on top of making the switch to Cyber Monday language in the wee hours of Saturday, Nov. 25."
[Type]: Fine-grained
3. [Event]: Amazon's Cyber Monday mode includes deals carried over from Black Friday plus some new ones.
[Topic]: Cyber Monday sale
[Original context]: "Official Cyber Monday mode, which is currently on through Monday, Nov. 27, includes both a ton of deals carried over from Black Friday plus some new ones."
[Type]: Fine-grained

GPT-4o-mini Generated Query:
1. [Event]: Amazon began its Black Friday sale. [Question]: How did Amazon's early start to the Black Friday sale impact customer engagement compared to previous years?
2. [Event]: Amazon switched to Cyber Monday language. [Question]: What strategies did Amazon employ to transition from Black Friday to Cyber Monday promotions?
3. [Event]: Amazon's Cyber Monday mode includes deals carried over from Black Friday plus some new ones. [Question]: What types of new deals can customers expect during Amazon's Cyber Monday sale compared to those from Black Friday?

C Ablation Study

We conduct three sets of experiments on Multihop-RAG and Qwen model while controlling different variables, investigating four key factors according to our pipeline:

- selection of k, we explore values of k at 200, 500, and 1000;
- selection of m, we examine m values ranging from 6 to 10;

- sampling strategy, compared fine-to-coarse and uniform approaches, fixing the first partition from 0 to 3 for an informative positive sample, while dividing the remaining partitions based on the chosen strategy. Specifically, when using the fine-to-coarse strategy, for a given k and m, the length of the next interval is twice the length of the previous interval. This can be represented by the formula: $L(\mathcal{P}_{i+1}) = 2L(\mathcal{P}_i)$;
- hyperparameter α, for convenience, we work with its reciprocal, $1/\alpha$, with values of 0.1, 0.2, 0.5, 0.7, and 1.0.

Our experiments are structured as follows:

1. We fix temperature = 1 and k=1000, and conduct experiments with different values of m and sampling strategies.
2. We fix temperature = 1, m=10, and the fine-to-coarse strategy, then investigate different values of k.
3. We fix k=500, m=10, and the fine-to-coarse strategy, then examine the effect of varying temperature.

Our ablation experiment results in Table 6 demonstrate that, **fine-tuned embedding model with lower alignment and higher uniformity tend to achieve better result on retrieval task.** We observe a strong correlation between retrieval performance and these two properties. Specifically, embedding models with better alignment tend to achieve superior retrieval results. Moreover, when alignment is similar, models with larger uniformity exhibit better retrieval performance. This suggests that we can leverage our strategy to adjust alignment and uniformity, ultimately optimizing retrieval performance.

D Alignment and Uniformity: Details and Discussion

In the work of Wang and Isola (2020), **Alignment**, which measures how well similar data points are positioned in the embedding space, is quantified by the mean Euclidean distance between the embeddings of all positive pairs. **Uniformity**, which reflects how well the data points are distributed across the embedding space, is quantified using the Gaussian potential kernel, capturing the pairwise similarity across all data points in the distribution, they are denoted as follows:

$$\text{Alignment} = \mathbb{E}_{x,y \in pos}[\|e(x) - e(y)\|_2^2]$$
$$\text{Uniformity} = log\mathbb{E}_{x,y \in p_{data}}[exp(-2\,\|h(x) - h(y)\|_2^2)] \quad (860)$$

where $x, y \in pos$ represents the positive pairs in the dataset, and p_{data} is the data distribution of all data points, $e(\cdot)$ is the embedding model that maps input data points to their corresponding embeddings in a high-dimensional space. In our experiments, $x, y \in pos$ refer to the question and its corresponding evidence chunk, while we randomly sample chunks from each document, forming a set of p_{data} to compute uniformity.

Since fine-tuning can further amend the model's alignment (Gao et al., 2021), making it difficult to compare across different models, we introduce a scaling factor to address this. A model with high alignment does not necessarily perform worse in retrieval than one with low alignment. If a high-alignment model also ensures that negative samples are more dispersed relative to positive ones, it can still achieve strong retrieval performance. Considering this, we define the distance between the query and its nearest embedding in the database as a scaling factor for alignment. In the following experiments, we use the normalized version of alignment, which denotes as follows:

$$\text{Alignment}_{norm} = \mathbb{E}_{x,y \in pos}[\frac{\|e(x) - e(y)\|_2^2}{\|e(x) - e(y_{\text{nearest}})\|_2^2}] \quad (884)$$

, where $e(y_{\text{nearest}})$ refers to the closest embedding in the database to the question embedding $e(x)$. Finally, the original uniformity is a negative value, in our experiments, we report the absolute value of uniformity. This makes comparison and analysis easier, and a larger absolute value indicates that the embedding model distribution is more uniform.

E Prompts Used for Keywords Masking

The LLM prompts used in the keywords masking experiments are detailed as follows:

Method	Alignment	Uniformity	Hit@10	Hit@4	Hit@1	MAP@10
Base	1.2422	2.7624	76.50	59.69	33.97	22.22
m=6 k=1000 fine-to-coarse	1.2031	3.1258	82.44	67.45	38.94	25.94
m=7 k=1000 fine-to-coarse	1.1953	3.1907	83.99	69.00	41.02	26.76
m=8 k=1000 fine-to-coarse	1.1953	3.3276	83.33	68.91	39.38	26.29
m=9 k=1000 fine-to-coarse	1.2031	3.3266	83.06	68.34	40.58	26.54
m=10 k=1000 fine-to-coarse	1.2031	3.3267	83.55	68.43	40.04	26.43
m=6 k=1000 uniform	1.2734	3.6012	80.27	64.79	36.98	24.41
m=7 k=1000 uniform	1.2656	3.5860	81.37	65.19	36.76	24.79
m=8 k=1000 uniform	1.2578	3.6276	82.35	67.49	38.67	25.61
m=9 k=1000 uniform	1.2578	3.6222	81.46	65.90	38.18	25.24
m=10 k=1000 uniform	1.2734	3.6265	80.71	64.26	36.50	24.39
k=1000 unifom m=10	1.2734	3.6265	80.71	64.26	36.50	24.39
k=500 unifom m=10	1.2578	3.6303	81.46	65.45	36.76	24.72
k=200 unifom m=10	1.2422	3.6452	82.97	66.39	37.69	25.23
k=1000 fine-to-coarse m=10	1.2031	3.3267	83.55	68.43	40.04	26.43
k=500 fine-to-coarse m=10	1.1953	3.3675	83.50	68.74	40.71	26.67
k=200 fine-to-coarse m=10	1.1953	3.3896	83.10	68.65	38.85	26.11
$1/\alpha$=0.1 k=1000 fine-to-coarse m=10	1.1953	2.1774	78.14	63.02	35.48	23.96
$1/\alpha$=0.2 k=1000 fine-to-coarse m=10	1.1875	2.6560	81.46	66.43	37.83	25.47
$1/\alpha$=0.5 k=1000 fine-to-coarse m=10	1.1875	3.2849	82.88	67.63	40.09	26.34
$1/\alpha$=0.7 k=1000 fine-to-coarse m=10	1.1953	3.3411	83.10	68.29	39.96	26.45
$1/\alpha$=1.0 k=1000 fine-to-coarse m=10	1.1953	3.3675	83.50	68.74	40.71	26.67

Table 6: Ablation study.

Masked Keyword Prompt:
Given a query and a paragraph including the answer of the query, please extract all the common keywords that query and paragraph both have:
Note:
1. The definition of keywords is: words in the query and paragraph that are particularly distinctive and related to the main topic. Less important pronouns or frequently occurring words do not fall into this category.
2. The words you extract must appear in both the query and the paragraph.
3. Do not output other format, just list all the words as follows:
investigation, Eastwood, Filing
Query:
{query}
Paragraph:
{paragraph}
keywords: