BRIDGE-Embed: A Novel LLM-based Document and Sentence Retrieval Method

Tianzhe Ning[1], Yifei Wu[1], and Guohua Liu[1] ✉

Donghua University, Shanghai, China
2222751@mail.dhu.edu.cn, wunein@mail.dhu.edu.cn, ghliu@dhu.edu.cn

Abstract. Large Language Models (LLMs) have shown remarkable performance in generating dense embeddings for retrieval tasks, yet their high-dimensional outputs pose significant challenges in storage and computation. Existing sparse indexing methods, meanwhile, often face vocabulary-based limitations, computational inefficiency, and performance degradation in sparse and short-text scenarios. This paper introduces a novel LLM-based document and sentence retrieval method that addresses these limitations by simultaneously generating high-quality dense and sparse representations in a single inference pass. Key innovations include enhanced sparse representation richness through Medusa heads, a sparsity-optimized Matryoshka loss function, and experimental validation demonstrating the effectiveness of our method across various retrieval tasks. Our approach provides an efficient and accurate solution that balances the strengths of both dense and sparse representations.

Keywords: Large language model · Dense-sparse representation · Medusa head

1 Introduction

Large Language Models (LLMs) have become powerful text embedding generators for dense document retrieval, achieving top performance on benchmarks like MTEB [11]. However, their high-dimensional embeddings pose significant storage and computational challenges, and dimension reduction techniques like Matryoshka loss [14] often lead to performance degradation, especially at lower dimensions.

The Retrieval-Augmented Generation (RAG) community has revisited sparse indexing methods, which offer storage efficiency while maintaining retrieval performance [7]. Some approaches attempt to generate sparse representations from dense embeddings [23], but they still face challenges: vocabulary-based transformations rely on word roots, reducing discriminative power; sparse representation training introduces high memory overhead, particularly with large vocabularies; performance degrades in short-text scenarios, especially in Semantic Textual Similarity (STS) tasks [9], where minor word variations significantly impact retrieval accuracy.

To overcome these limitations, we propose a novel LLM-based document and sentence retrieval method. We prompt an LLM with a single word and extract the last token's hidden state as the dense representation; simultaneously, Medusa head logits are used to generate a refined sparse representation. As shown in Fig. 1, this allows us to obtain high-quality dense and sparse representations in a single forward pass. Our key contributions include:

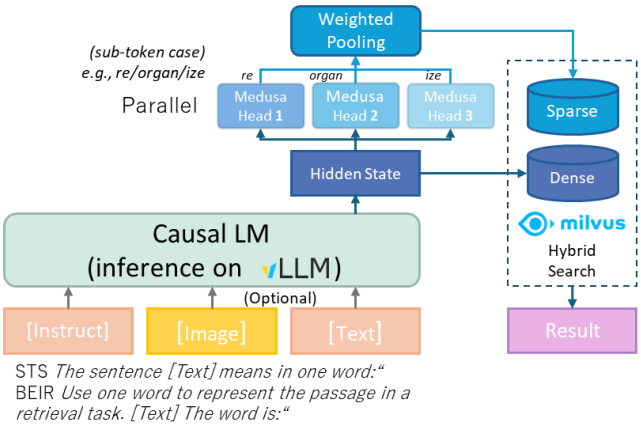

Fig. 1. Overview of our framework in inference.

- **Enhanced Sparse Representation Richness**: Medusa heads introduce parallel LM head decoding, capturing richer semantic information across short and long texts.
- **Sparsity-Optimized Loss Function**: Our S-Matryoshka loss ensures robust sparse embeddings with low memory overhead and reduced performance degradation at low top-K values.
- **Compatibility and Efficiency**: Extensive experiments validate the effectiveness of our method across retrieval tasks, demonstrating its efficiency and compatibility with acceleration frameworks like vLLM.

2 Proposed Method

We enhance sparse embeddings using Medusa heads and contrastive training, as illustrated in Fig. 2. The last token's output is extracted to adapt Medusa heads, followed by self-distillation for dense embedding training and joint dense-sparse training via contrastive learning. To improve low Top-K performance, we design a sparse Matryoshka loss and an RFF-Ranker loss for hybrid search.

2.1 Medusa Heads for Enhancing Semantic Richness in Sparse Representations

Inspired by [2], we use Medusa heads to enhance the semantic richness of sparse representations. Originally designed for token prediction to accelerate LLM inference, Medusa heads now serve to generate sparse representations, improving the model's representation learning. This process is shown in the left panel of Fig. 2. Given the model's final hidden states h_t at position t, we append N additional decoding heads:

$$p_t^{(n)} = \text{softmax}\left(W_2^{(n)} \cdot \left(\text{SiLU}(W_1^{(n)} \cdot h_t) + h_t\right)\right), \tag{1}$$

where $W_2^{(n)} \in \mathbf{R}^{V \times d}$, and $W_1^{(n)} \in \mathrm{R}^{d \times d}$. Here, d denotes the output dimension of the LLM's final hidden layer, and V represents the vocabulary size. The n in the top

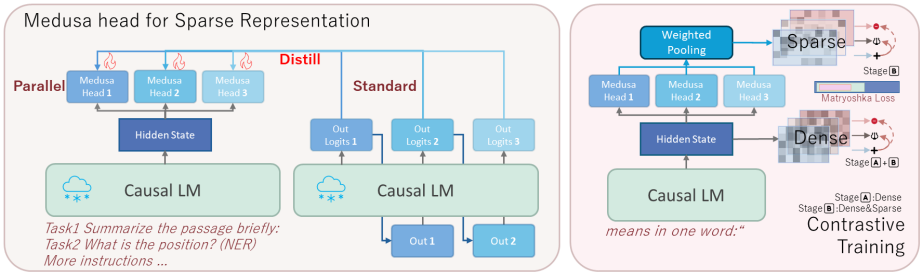

Fig. 2. Left: Self-distillation to align the Medusa heads. **Right**: Contrastive learning to encourage a better sparse and dense representation. Both parts contribute to a better hybrid representation.

right corner of W represents the weight matrix corresponding to the n-th Medusa Head. To ensure alignment between the Medusa heads and the original model, we employ a self-distillation mechanism:

$$\mathcal{L}_{\text{LM-distill}} = KL(p^{(0)}_{\text{original},t} || p^{(0)}_{t}), \tag{2}$$

where $p^{(0)}_{\text{original},t}$ represents the probability distribution of the original model at position t.

To maintain sparsity while leveraging information from all Medusa heads, we introduce a novel logits aggregation method:

$$\mathbf{h}_{\text{logits}} = \sum_{n=1}^{N} (1 - p^{(n)}_{\text{eos,-1}}) \cdot p^{(n)}_{\text{-1}}, \tag{3}$$

where $p^{(n)}_{\text{eos,-1}}$ denotes the probability of the end-of-sequence token from the n-th Medusa head, and $p^{(n)}_{\text{-1}}$ represents the logits from the n-th Medusa head at the final sequence output position (aka last token). This formulation enables us to incorporate information from all N Medusa heads to enrich semantic information.

2.2 Contrastive Loss for Dense and Sparse

Contrastive learning has become a powerful technique in natural language processing tasks. As shown in [9, 6], it effectively learns both dense and sparse representations by creating meaningful contrasts within the data. The contrastive loss is defined as:

$$L_{\text{con}} = -\log \frac{e^{sim(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{B} (e^{sim(\mathbf{h}_i, \mathbf{h}_j)/\tau} + e^{sim(\mathbf{h}_i, \mathbf{h}_j^-)/\tau})}, \tag{4}$$

where B is the batch size, $\mathbf{h}$, $\mathbf{h}^+$, and $\mathbf{h}^-$ represent the anchor, positive, and negative samples, respectively, and τ is a temperature. The numerator measures the similarity between the anchor and positive sample, while the denominator sums the similarities of all samples. The goal is to minimize the contrastive loss by bringing similar samples closer and pushing dissimilar ones apart.

2.3 Sparsity-Optimized Matryoshka Loss

Building upon the initial dense logits representation $\mathbf{h}_{\text{logits}} \in \mathbf{R}^V$, we introduce the Sparsity-Optimized Matryoshka Loss, enhancing the model's ability to capture semantic information at various levels of granularity while maintaining computational efficiency. We first define a series of sparse embeddings (with k dimension):

$$\mathbf{h}_{\text{sparse}} = \text{Norm}(\text{SoftTopK}(\mathbf{h}_{\text{logits}}, \tau)_{1:d_k}), \tag{5}$$

where $\text{Norm}(\cdot)$ is the L2 normalization and SoftTopK is the differentiable top-k operation with temperature τ, d_k is top-k index for building **memory-saving** torch **sparse** `COO` tensor. The `SoftTopK` operation computes a differentiable sparse representation by applying a temperature-controlled softmax followed by top-k masking, expressed as:

$$\text{SoftTopK}(\mathbf{x}, k, \tau) = \text{softmax}(\mathbf{x}/\tau) \odot \mathbf{1}[\text{rank}(\text{softmax}(\mathbf{x}/\tau)) \leq k] \tag{6}$$

where τ is the temperature parameter, $\odot$ denotes element-wise multiplication, and $\mathbf{1}[\cdot]$ is the indicator function selecting the top-k elements. The Matryoshka loss is then formulated as a weighted sum of contrastive losses:

$$\mathcal{L}_{\text{Matryoshka}} = \sum_{k=1}^{m} w_k \cdot \mathcal{L}_{\text{con}}(\mathbf{h}_{\text{sparse}}), \tag{7}$$

Here, w_k are dimension-specific weights, and $\mathcal{L}_{\text{con}}$ is the contrastive loss. To further promote sparsity during training, we incorporate a sparsity regularization term:

$$\mathcal{L}_{\text{sparse}} = \lambda \sum_{k=1}^{m} \|\mathbf{h}_{\text{sparse}}\|_1, \tag{8}$$

here λ is the sparsity coefficient. This loss term introduces the L1 norm and a sparsity regularization weight λ, which limits the number of non-zero elements in the embedding vector, thereby generating sparse representations in the model. The total loss becomes:

$$\mathcal{L}_{\text{S-Matryoshka}} = \mathcal{L}_{\text{Matryoshka}} + \mathcal{L}_{\text{sparse}} \tag{9}$$

2.4 Re-ranking Loss for Hybrid Search

In hybrid search systems, effectively combining results from multiple retrieval methods is crucial. Milvus [21] employs the Reciprocal Rank Fusion (RRF) for this purpose. To align our training process with this approach, we propose an adaptive training strategy based on the RRF concept. The original RRF-Ranker function is defined as follows:

$$\text{RRF\_score}(d) = \sum_{i=1}^{D} \frac{1}{s + \text{rank}_i(d)} \tag{10}$$

Here, D represents the number of different retrieval routes, $\text{rank}_i(d)$ is the rank position of retrieved document d by the ith retriever, and s is a smoothing parameter. To adapt this

concept for our training process, we design a loss function that more closely approximates the original RRF. Given that our training data consists of triplets, we introduce a softened ranking function:

$$\text{SoftRank}(sim) = \frac{1}{1 + e^{-sim/\tau}}, \tag{11}$$

here, τ is a temperature hyperparameter that controls the smoothness of the ranking. Using this soft ranking, we can compute an RRF-inspired score for both positive and negative samples:

$$\begin{aligned} \text{RRF}_{pos} = &\frac{1}{s + \text{SoftRank}(sim_{\text{sparse}})} \\ &+ \frac{1}{s + \text{SoftRank}(sim_{\text{dense}})} \end{aligned} \tag{12}$$

This formulation combines the contributions from both sparse and dense representations, mirroring the hybrid nature of our search system. Finally, we define our loss as a margin-based hinge loss:

$$\mathcal{L}_{\text{RFF}} = \max(0, \text{RRF}_{neg} - \text{RRF}_{pos} + \text{margin}) \tag{13}$$

3 Experiments and Results

3.1 Experimental Setup

Data and Metrics The **training data** consists of triples: an anchor, a positive, and a negative sample, using MS MARCO [1], MNLI [13], and SNLI [10]. For **testing**, we use BEIR [20] and the MS MARCO development set for retrieval tasks, and the STS-Benchmark for Semantic Textual Similarity (STS) tasks. Performance is evaluated using nDCG@10 for BEIR, MRR@10 for MS MARCO, and Spearman correlation for STS.

Training Details Experiments were conducted on 8 NVIDIA Tesla A100 80G GPUs using PyTorch 2.3.1. The key parameters are: $N = 3$ decoding heads, temperature $\tau = 0.05$. Training consists of two stages: dense embedding training followed by joint dense-sparse training. The first stage utilizes the sentence-transformers library [18] with a batch size of 25600, employing Gradient Cache [8]. In the second stage, as Gradient Cache is incompatible with Matryoshka loss, we use QLoRA [4] with $r = 64$ and $\alpha = 16$, torch sparse COO tensor, achieving a batch size of 1024. The sparse Matryoshka loss is applied with $K \in \{128, 256, 512\}$, and the RRF-Ranker smoothing parameter is set to $s = 60$.

Baselines We compare our method with state-of-the-art baselines, all fine-tuned on the training set for fair evaluation.

Sparse Methods: **BM25**: A classical probabilistic retrieval function. **SPLADE++** [6]: An enhanced sparse retrieval model with strong performance on BEIR and MSMARCO. **SPLATE** [5]: A sparse retrieval model adapted from ColBERTv2.

Hybrid Methods: **BGE-M3** [3]: A multi-lingual, multi-functional embedding model. **PromptReps** [23]: A method leveraging LLMs to generate dense and sparse representations without additional training.

Dense Methods: **GritLM-7B** [16]: A unified text embedding and generation model. **gte-Qwen2-7B-instruct** [15]: An instruction-tuned Qwen2 variant for text embeddings. **nomic-embed-text-v1.5** [17]: A long-context embedder outperforming OpenAI models.

Our Methods: We evaluate several variants of our approach in both sparse and hybrid settings: **Scratch**, trained on Qwen2-7B-instruct [22]; **gte-Qwen2**, based on gte-Qwen2-7B-instruct; and **GritLM**, based on GritLM-7B.

Research Questions To evaluate the performance of our method, we design experiments to answer the following research questions:

RQ1: How effective is our method compared to existing baselines?

RQ2: How effective is our sparsity-optimized loss and re-ranker-optimized loss?

RQ3: What about using tricks like Chinese prompt or remove markup syntax tokens?

RQ4: How effective is our method is image-text matching?

3.2 RQ1: Performance Comparison

Table 1. Results comparison between our method and other baselines, under sparse, dense and hybrid settings. (**Bold**: the best. Underlined: the second best.)

Method	Dim(K)	MSMARCO	BEIR	STS-b
Sparse				
BM25	100	34.3	49.0	-
SPLADE++	100	38.0	50.7	-
SPLATE	100	<u>40.4</u>	49.9	-
Hybrid				
BGE-M3	>1024	26.7	50.7	76.3
PromptReps	>4096	24.6	44.1	79.7
Dense				
GritLM-7B	4096	35.1	59.2	85.6
gte-Qwen2-7B	4096	39.2	58.6	86.9
nomic-v1.5	768	36.3	54.8	85.5
	128	33.6	49.4	84.3
	256	34.8	52.5	85.3
Ours (Sparse)				
Ours (scratch)	128	35.8	52.9	<u>87.6</u>
	256	36.2	53.7	**87.9**
Ours (gte-Qwen2)	128	40.1	57.3	86.2
	256	39.5	56.9	86.8
	512	39.3	57.9	86.4
Ours (GritLM)	128	34.5	57.2	84.1
	256	35.0	58.6	85.0
	512	36.1	**59.2**	85.3
Ours (Hybrid): Dense(4096) + Sparse(128)				
Ours (scratch)	4224	35.8	54.9	87.2
Ours (gte-Qwen2)	4224	**40.5**	<u>59.0</u>	87.6
Ours (GritLM)	4224	36.1	58.9	83.9

Table 1 compares our method with various baselines across tasks and embedding dimensions. Our sparse representations show strong efficiency, especially at lower dimensions. At 128 dimensions, our gte-Qwen2-based sparse model achieves an MRR@10 of 40.1 on MSMARCO, outperforming its dense counterpart, demonstrating that our method maintains high performance with reduced dimensionality. However, the gap is smaller in STS tasks, where the sparse model performs slightly worse.

Increasing the dimensionality from 128 to 256 generally improves performance, but the degree of improvement varies across tasks and models. For instance, the GritLM-based sparse model shows significant improvement in BEIR scores, while the gte-Qwen2-based model sees a slight drop in MSMARCO performance, suggesting that optimal dimensionality may depend on the task and model.

Our hybrid approach, combining sparse and dense representations, performs best overall. The gte-Qwen2-based hybrid model achieves top scores in MSMARCO and near-top scores in BEIR, demonstrating the complementarity of sparse and dense representations.

Our sparse representations also excel in STS tasks, especially for models trained from scratch, achieving up to 87.9% correlation. This success is attributed to the simpler structure of STS sentences, which allows our sparse method to effectively capture key semantic details.

3.3 RQ2: Effectiveness of Sparsity-Optimized Loss

Table 2. Results comparison between effectiveness of sparsity-optimized loss.

Method	Dim	MSMARCO	STS-b
Ours (Sparse)			
Ours (gte-Qwen2)	128	**40.1**	86.2
	256	39.5	**86.8**
w/o Medusa Head	128	36.3	83.2
	256	37.1	84.4
w/o S-Matryoshka	128	38.1	84.9
	256	39.1	85.1
Ours (Hybrid): Dense + Sparse			
Ours (gte-Qwen2)	4096 + 128	**40.5**	**87.6**
	4096 + 256	40.5	87.5
w/o Re-Ranking loss	4096 + 128	40.1	87.4
	4096 + 256	39.3	87.0
Ours (Hybrid Matryoshka): Dense + Sparse			
Ours (gte-Qwen2)	256 + 128	39.0	85.1
	256 + 256	39.8	85.8

Table 2 shows the results of our ablation study, highlighting the impact of sparsity-optimized loss components on model performance.

Medusa Head and S-Matryoshka loss improve performance, especially for lower-dimensional representations. For the 128-dimensional sparse model, MSMARCO increases from 36.3 to 40.1 and STS-b from 83.2 to 86.2. The performance gain decreases as dimensions increase. The re-ranking loss benefits the STS-b task, improving the score

from 87.4 to 87.6 in the hybrid model (4096 + 128), enhancing fine-grained semantic capture.

The hybrid Matryoshka approach, applying Matryoshka loss to both sparse and dense embeddings, offers a good balance between performance and efficiency. The 256 + 256 configuration achieves competitive scores, showing its potential in resource-constrained scenarios. Performance on the STS task remains closely tied to dense embedding quality, demonstrating the complementary roles of sparse and dense representations in capturing semantic similarities.

3.4 RQ3: Impact of different vocabulary processing techniques

Table 3. Results comparison between different vocab processing techniques after training.

Method	MSMARCO	STS-b
Ours (Sparse@128)		
Ours (scratch)	35.8	**87.6**
Prompt in Chinese	34.5	83.8
w/o Syntax Token	**36.1**	87.6
w/o Frequent sub-word token	32.8	82.8

We can draw several insights about the impact of different vocabulary processing techniques on our sparse embedding model, based on the results presented in Table 3. The baseline model trained from scratch shows robust performance across both MSMARCO and STS-b tasks. Interestingly, using Chinese prompts during training led to a noticeable decrease in performance, particularly on the STS-b task. This suggests that introducing multilingual elements may overly strain the model's capacity, potentially compromising its effectiveness in monolingual contexts. Removing syntax tokens such as '\n' yielded a marginal improvement, indicating that these tokens play a minimal role in the model's understanding. However, the most significant impact was observed when removing frequently used sub word tokens shorter than a certain length. This approach resulted in a substantial performance drop across both tasks, highlighting the critical role that short tokens, including roots and affixes, play in our Medusa Head architecture. Given these results, we believe that a more promising approach to enhance performance might be to increase the vocabulary size, a strategy that has shown potential in recent LLM research [19].

3.5 RQ4: Performance comparison in Image-Text matching

Our proposed method demonstrates competitive performance in image-text matching in Table 4. The hybrid approach shows notable improvements in image retrieval (COCO I-R@1), outperforming both our base model and the E5-V [12]. This indicates that our method effectively leverages the strengths of both dense and sparse representations in multi-modal contexts.

Table 4. Results comparison in image-text matching, using same base model LLaVA-NeXT-8B and training dataset. We maintain the sparse setting of $K = 128$.

Method	COCO I-R@1	COCO T-R@1	STS-b
Ours	50.4	61.3	87.5
$\text{Ours}_{\text{Hybird}}$	**53.1**	61.2	87.4
E5-V	52.0	**62.0**	**87.9**

However, the results also reveal some limitations. In text retrieval and STS tasks, our method's performance is slightly lower than E5-V. This suggests that sparse representations, while effective overall, may face challenges with short texts and strong negative examples typical in image-text retrieval tasks. Despite these challenges, the competitive performance across all metrics demonstrates the viability of our approach in multi-modal scenarios. The results underscore the potential of our method while highlighting areas for future improvement, particularly in handling short text inputs and enhancing discriminative power in image-text matching tasks.

4 Conclusion

We proposed a novel LLM-based method for document and sentence retrieval that addresses key limitations of existing dense and sparse embeddings. Our approach enhances sparse representations with Medusa heads, optimizes sparsity with Matryoshka loss, and ensures efficiency, as validated by ablation studies. These innovations improve retrieval performance and efficiency, particularly in resource-constrained scenarios. Our method demonstrates robustness across various tasks, advancing hybrid text embedding for retrieval.

References

1. Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., et al.: Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268 (2016)
2. Cai, T., Li, Y., Geng, Z., Peng, H., Lee, J.D., Chen, D., Dao, T.: Medusa: Simple llm inference acceleration framework with multiple decoding heads. In: Forty-first International Conference on Machine Learning
3. Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D., Liu, Z.: Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. arXiv preprint arXiv:2402.03216 (2024)
4. Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L.: Qlora: efficient finetuning of quantized llms. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. pp. 10088–10115 (2023)
5. Formal, T., Clinchant, S., Déjean, H., Lassance, C.: Splate: Sparse late interaction retrieval. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2635–2640 (2024)

6. Formal, T., Lassance, C., Piwowarski, B., Clinchant, S.: From distillation to hard negative sampling: Making sparse neural ir models more effective. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2353–2359 (2022)
7. Formal, T., Piwowarski, B., Clinchant, S.: Splade: Sparse lexical and expansion model for first stage ranking. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2288–2292 (2021)
8. Gao, L., Zhang, Y.: Scaling deep contrastive learning batch size under memory limited setup. In: Proceedings of the 6th Workshop on Representation Learning for NLP (2021)
9. Gao, T., Yao, X., Chen, D.: Simcse: Simple contrastive learning of sentence embeddings. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 6894–6910 (2021)
10. Glockner, M., Shwartz, V., Goldberg, Y.: Breaking nli systems with sentences that require simple lexical inferences. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 650–655 (2018)
11. Jiang, T., Huang, S., Luan, Z., Wang, D., Zhuang, F.: Scaling sentence embeddings with large language models. arXiv preprint arXiv:2307.16645 (2023)
12. Jiang, T., Song, M., Zhang, Z., Huang, H., Deng, W., Sun, F., Zhang, Q., Wang, D., Zhuang, F.: E5-v: Universal embeddings with multimodal large language models. arXiv preprint arXiv:2407.12580 (2024)
13. Kim, S., Kang, I., Kwak, N.: Semantic sentence matching with densely-connected recurrent and co-attentive information. In: Proceedings of the AAAI conference on artificial intelligence. pp. 6586–6593 (2019)
14. Kusupati, A., Bhatt, G., Rege, A., Wallingford, M., Sinha, A., Ramanujan, V., Howard-Snyder, W., Chen, K., Kakade, S., Jain, P., et al.: Matryoshka representation learning. Advances in Neural Information Processing Systems **35**, 30233–30249 (2022)
15. Li, Z., Zhang, X., Zhang, Y., Long, D., Xie, P., Zhang, M.: Towards general text embeddings with multi-stage contrastive learning. arXiv preprint arXiv:2308.03281 (2023)
16. Muennighoff, N., Hongjin, S., Wang, L., Yang, N., Wei, F., Yu, T., Singh, A., Kiela, D.: Generative representational instruction tuning. In: ICLR 2024 Workshop: How Far Are We From AGI (2023)
17. Nussbaum, Z., Morris, J.X., Duderstadt, B., Mulyar, A.: Nomic embed: Training a reproducible long context text embedder (2024)
18. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (2019)
19. Tao, C., Liu, Q., Dou, L., Muennighoff, N., Wan, Z., Luo, P., Lin, M., Wong, N.: Scaling laws with vocabulary: Larger models deserve larger vocabularies. arXiv preprint arXiv:2407.13623 (2024)
20. Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., Gurevych, I.: BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (2021)
21. Wang, J., Yi, X., Guo, R., Jin, H., Xu, P., Li, S., Wang, X., Guo, X., Li, C., Xu, X., et al.: Milvus: A purpose-built vector data management system. In: Proceedings of the 2021 International Conference on Management of Data. pp. 2614–2627 (2021)
22. Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., et al.: Qwen2 technical report. arXiv preprint arXiv:2407.10671 (2024)
23. Zhuang, S., Ma, X., Koopman, B., Lin, J., Zuccon, G.: Promptreps: Prompting large language models to generate dense and sparse representations for zero-shot document retrieval. arXiv preprint arXiv:2404.18424 (2024)