Distribution-Driven Dense Retrieval: Modeling Many-to-One Query-Document Relationship

Junfeng Kang[1], **Rui Li**[1], **Qi Liu**[1, 2*], **Zhenya Huang**[1, 2],
Zheng Zhang[1], **Yanjiang Chen**[1], **Linbo Zhu**[2], **Yu Su**[2,3]

[1]State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China
[2]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
[3]School of Computer Science and Artificial Intelligence, Hefei Normal University
{kangjf, ruili2000, zhangzheng, yjchen}@mail.ustc.edu.cn, {qiliuql, huangzhy}@ustc.edu.cn,
lbzhu@iai.ustc.edu.cn, yusu@hfnu.edu.cn

Abstract

Dense retrieval has emerged as the leading approach in information retrieval, aiming to find semantically relevant documents based on natural language queries. Given that a single document can be retrieved by multiple distinct queries, existing methods aim to represent a document with multiple vectors. Each vector is aligned with a different query to model the many-to-one relationship between queries and documents. However, these multiple vector-based approaches encounter challenges such as Increased Storage, Vector Collapse, and Search Efficiency. To address these issues, we introduce the Distribution-Driven Dense Retrieval framework (DDR). Specifically, we use vectors to represent queries and distributions to represent documents. This approach not only captures the relationships between multiple queries corresponding to the same document but also avoids the need to use multiple vectors to represent the document. Furthermore, to ensure search efficiency for DDR, we propose a dot product-based computation method to calculate the similarity between documents represented by distributions and queries represented by vectors. This allows for seamless integration with existing approximate nearest neighbor (ANN) search algorithms for efficient search. Finally, we conduct extensive experiments on real-world datasets, which demonstrate that our method significantly outperforms traditional dense retrieval methods.

Code — https://github.com/tojunfeng/DDR

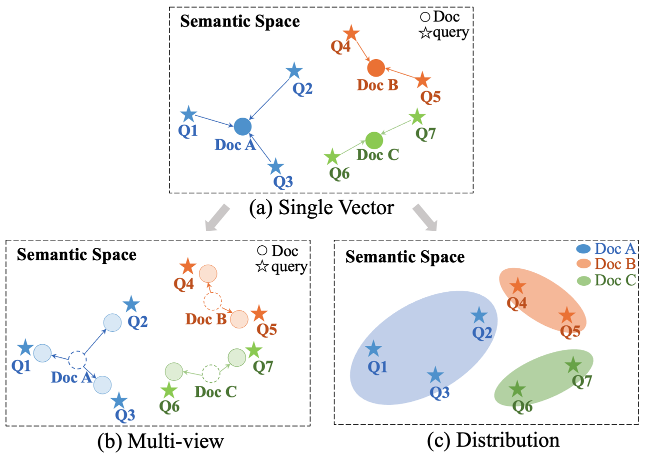

Figure 1: (a) The single vector approach represents query and document with a single vector, subsequently aligning the query vector with the relevant document vector. (b) The multi-view approach involves splitting the document into multiple vector representations, each aligning with different queries. (c) Our method is capable of capturing the relationships between multiple queries corresponding to the same document by representing queries as vectors and documents as distributions.

Introduction

Dense retrieval aims to search for semantically relevant documents given a natural language query. These approaches (Lee, Chang, and Toutanova 2019; Karpukhin et al. 2020; Qu et al. 2021) involve transforming the input documents and queries into dense vectors, and then performing information retrieval by calculating the similarity between these vectors. Various methods such as pre-training (Gao and Callan 2021; Wang et al. 2023), negative sample mining (Xiong et al. 2021; Qu et al. 2021), and knowledge distillation (Ren et al. 2021; Santhanam et al. 2022) have been proposed and have made great progress.

Despite effectiveness, these methods face a challenge where multiple different queries retrieve the same document. As depicted in Figure 1(a), queries *Q1*, *Q2*, and *Q3* all retrieve the same document *Doc A*. This many-to-one relationship makes it challenging for a document represented by a single vector to effectively align with all relevant query vectors. As a result, the previous single vector-based methods have somewhat lost their effectiveness. To address this problem, researchers have observed that queries and documents exhibit the following characteristics (Tang et al. 2021; Zhang et al. 2022; Luan et al. 2021): 1) Documents are typically longer and contain more semantic information compared to queries; 2) A query usually corresponds to a segment of the target document, and one document can be retrieved by different queries. Consequently, they have proposed a **multi-view** approach to mitigate this issue. Specifically, they have

[*]Corresponding Author.

opted for representing a document with multiple vectors (as illustrated in Figure 1(b)). This strategy ensures that each vector captures distinct semantic details of the document, aligning it with different queries.

Although employing multiple vectors to represent documents can partially address this issue, it still faces several problems: (1) *Increased Storage*: In practical retrieval scenarios, the storage cost for indexing all documents is already substantial. Using multiple vectors to represent a single document can multiply the storage requirements. For instance, the storage space required by MVR (Zhang et al. 2022) is eight times greater than that of single-vector methods. (2) *Vector Collapse*: The training data annotations are quite sparse, with most documents in the dataset corresponding to only one labeled query. This sparsity of labeled data increases the risk that multiple learned vectors for a given document will collapse into nearly identical representations (Luan et al. 2021; Liao et al. 2023). (3) *Search Efficiency*: Multi-vector methods require interactions between multiple vectors from the query and the document (e.g., MaxSim) to determine the similarity scores, resulting in higher retrieval latency compared to single-vector methods. For instance, even though ColBERT (Khattab and Zaharia 2020) introduces a two-stage retrieval approach to optimize retrieval efficiency, its latency is over 100 times greater than single-vector methods (Li et al. 2023).

Given that these problems of multi-vector methods arise from using *multiple vectors* to model a single document, a natural question arises: ***How can we address the phenomenon where multiple queries retrieve the same document without using multiple vectors?*** Inspired by the research on word embeddings (Vilnis and McCallum 2014; Athiwaratkun and Wilson 2017), we employ vectors to represent queries and distributions to represent documents. This novel modeling approach brings us the following benefits: 1) based on the correspondence between vectors and distributions, it can capture the relationships between multiple queries corresponding to the same document, thereby enhancing retrieval efficiency; 2) by modeling documents as distributions, we circumvent the need to model a single document as multiple vectors, thus avoiding issues such as *Increased Storage* and *Vector Collapse*.

While the distribution-based method in dense retrieval has brought about certain conveniences, it also introduces a new challenge. Specifically, when assessing the relevance between documents and queries, unlike traditional methods that use the dot product to measure similarity, the most direct computation method in the distribution-based approach involves evaluating the likelihood of the query vector within the probability density function representing the document distribution. Consequently, existing approximate nearest neighbor (ANN) algorithms optimized for dot product calculations are not suitable for the online search phase. This leads to the need to calculate the similarity between the query q and *every* document d individually, posing efficiency challenges in large-scale retrieval. Therefore, another critical question arises: ***How can we address the issue of search efficiency for distribution-based dense retrieval?*** In response, we delve into a thorough theoretical analysis and discover that the similarity calculation method in the distribution-based approach can be converted into dot product calculations, involving the query vector, the mean, and the covariance of the document distribution. Building upon this theoretical insight, we propose a comprehensive framework called **Distribution**-Driven **D**ense **R**etrieval (**DDR**), where documents are represented using multivariate Gaussian distributions and queries are represented as vectors. The relevance between documents and queries is computed using dot product calculations that consider the query vector, the mean, and the covariance of the document distribution. In this way, this framework allows for seamless integration with existing ANN algorithms, maintaining retrieval efficiency on par with single-vector methods.

To validate the efficacy of our method, we conducted experiments on several large-scale web search datasets, including MSMARCO Passage Ranking (Nguyen et al. 2016), the TREC Deep Learning Track dataset (Craswell et al. 2020), and the BEIR zero-shot dataset (Thakur et al. 2021). The results demonstrated significant improvement compared to both single-vector and multi-vector approaches.

Our primary contributions are as follows:

- We propose modeling the query as a vector and the document as a distribution to capture the relationships between multiple queries corresponding to the same document, which also avoids the issues of *Increased Storage*, *Vector Collapse*, *Search Efficiency* typically associated with traditional multi-view modeling.
- We introduce the DDR framework and demonstrate how it can be seamlessly integrated into existing approximate nearest neighbor algorithms to enable efficient retrieval.
- We conducted experiments across various retrieval scenarios to validate the effectiveness of DDR framework.

Related Work

Dense Retrieval

With the development of pre-trained language models (Dong et al. 2019; Clark et al. 2020; Devlin et al. 2019), Transformer-based models have achieved significant advancements in text representation. The pre-training models specifically for retrieval tasks (Li et al. 2024a,b; He et al. 2023) has attracted interest from researchers. They employed methods such as weak-decoder (Lu et al. 2021) and Masked Auto-Encoder (Xiao et al. 2022; Wang et al. 2023) to enhance the [CLS] token's ability to better aggregate semantic information.

In the fine-tuning stage, various enhancements have been implemented, including negative sample mining (Xiong et al. 2021), data augmentation (Qu et al. 2021; Wang et al. 2024b), knowledge distillation (Lin, Yang, and Lin 2021) and ensemble learning methods (Zhao et al. 2023), among others. (Zhang et al. 2022; Luan et al. 2021; Tang et al. 2021) argue that a single vector struggles to encapsulate the rich information of documents. Thus, they represent documents with multiple vectors while keeping queries represented with a single vector, and then employ a MaxSim operation to calculate the similarity between documents and queries. Meanwhile, (Khattab and Zaharia 2020; Santhanam et al. 2022) use several vectors to represent passages and queries at the token level, introducing a late interaction mechanism for modeling the fine-grained similarity between queries and documents. (Gao, Dai, and Callan 2021; Li et al. 2023) propose implementing different lexical routing strategies to improve the performance of multi-vector methods.

Representation via Distribution

In the early stages, researchers explored the relationship between the distributional behaviors of word pairs and lexical entailment (Geffet and Dagan 2005), as well as the idea of representing words as regions within semantic space (Erk 2009). Building on this foundation, subsequent studies on word vectors led a group of researchers (Vilnis and McCallum 2014; Athiwaratkun and Wilson 2017) to investigate word embeddings using multivariate Gaussian distributions. In information retrieval domain, MRL (Zamani and Bendersky 2023) pioneered the use of distributions to model information uncertainty, leading to improved retrieval results. GMMFormer (Wang et al. 2024a) utilizes a Gaussian Mixture Model to implicitly represent clip features for video retrieval. Unlike previous research, we propose addressing the many-to-one problem between queries and documents by capturing their connections using Gaussian distributions.

Methodology

In this section, we will thoroughly explore the practical applications and various aspects of our proposed **D**istribution-Driven **D**ense **R**etrieval (DDR) framework.

Overview

Dense Retrieval (Karpukhin et al. 2020) is a task that involves retrieving the k most relevant documents $\{d_j\}_{j=1}^k$ from a large corpus for a given query q.

For this task, previous studies typically employ two separate encoders to generate dense embeddings for the query and document, respectively. Subsequently, the similarity score between the query and document is measured by computing the dot product of their embeddings.

In the DDR framework, we model the entailment relationship between queries and documents by representing queries as vectors and documents as multivariate Gaussian distributions. To determine the similarity score between the query and document, we evaluate the likelihood of the query vector under the probability density function (PDF) that represents the document distribution.

Model Architecture

As illustrated in Figure 2, we will introduce the architecture of DDR by discussing two main aspects: query representation, document representation.

Query Representation Pre-trained models have significantly improved performance across a wide range of downstream tasks. Notably, research has demonstrated that the [CLS] token often captures the overall meaning of an entire document (Devlin et al. 2019). For a given query consisting of N tokens, we leverage the [CLS] embedding vector generated by the encoder to serve as the semantic vector of the query:

$$[h_{[\text{CLS}]}, h_1, \cdots, h_N] = \text{Encode}_Q([\text{CLS}] \circ q \circ [\text{SEP}]), \quad (1)$$

$$\mathbf{q} = h_{[\text{CLS}]}. \quad (2)$$

Document Representation Each document d is represented by a k-dimensional independent Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In this context, $\boldsymbol{\mu}$ denotes a $k \times 1$ mean vector, and $\boldsymbol{\Sigma}$ is a *diagonal* covariance matrix of size $k \times k$. Using a diagonal matrix to model covariance can reduce storage requirements and improve computational efficiency. The distribution can be expressed as follows:

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}\left(\begin{pmatrix}\mu_1 \\ \mu_2 \\ \vdots \\ \mu_k\end{pmatrix}, \begin{pmatrix}\sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_k^2\end{pmatrix}\right). \quad (3)$$

For simplicity in notation, the mean vector $\boldsymbol{\mu}$ is defined as $(\mu_1, \mu_2, \dots, \mu_k)^T$, and the diagonal covariance matrix $\boldsymbol{\Sigma}$ is represented by its diagonal elements, which we can denote using the variance vector $\boldsymbol{\sigma^2} = (\sigma_1^2, \sigma_2^2, \dots, \sigma_k^2)^T$.

For a document consisting of N tokens, we similarly use the [CLS] embedding vector from the encoder to represent the mean vector $\boldsymbol{\mu}$ of the document's distribution:

$$[h_{[\text{CLS}]}, h_1, \cdots, h_N] = \text{Encode}_D([\text{CLS}] \circ d \circ [\text{SEP}]), \quad (4)$$

$$\boldsymbol{\mu} = h_{[\text{CLS}]}. \quad (5)$$

For the document variance vector $\boldsymbol{\sigma^2}$, we employ the attention mechanism to integrate the hidden states of all tokens. Specifically, given that the size of the hidden state dimension is d, we start by transforming $H = [h_{[\text{CLS}]}, h_1, h_2, \cdots, h_N] \in \mathbb{R}^{(N+1)\times d}$ into the corresponding query matrix Q, key matrix K, and value matrix V:

$$Q = H \cdot W_Q, \quad (6)$$

$$K = H \cdot W_K, \quad (7)$$

$$V = H \cdot W_V, \quad (8)$$

where $W_Q \in \mathbb{R}^{d\times d}, W_K \in \mathbb{R}^{d\times d}$, and $W_V \in \mathbb{R}^{d\times d}$ are trainable parameters.

Next, we compute the attention scores and obtain the resulting matrix, from which we extract the first vector, denoted as $h_{[\text{VAR}]}$:

$$[h'_{[\text{CLS}]}, h'_1, \cdots, h'_N] = \text{Softmax}\left(\frac{Q \cdot K^\top}{\sqrt{d}}\right) V, \quad (9)$$

$$h_{[\text{VAR}]} = h'_{[\text{CLS}]}. \quad (10)$$

Given the function $f(x) = \frac{1}{\beta}\log(1 + \exp(\beta x))$, which possesses the following properties: $\lim_{x\to-\infty} f(x) = 0$, $\lim_{x\to+\infty} \frac{f(x)}{x} = 1$, and is monotonic as well as differentiable, we can smooth $h_{[\text{VAR}]}$ to obtain the variance vector $\boldsymbol{\sigma^2}$ using the following transformation:

$$\boldsymbol{\sigma}^2 = \frac{1}{\beta}\log\left(1 + \exp\left(\beta h_{[\text{VAR}]}\right)\right). \quad (11)$$

This process ensures that the variance vector $\boldsymbol{\sigma^2}$ is differentiable and positive.

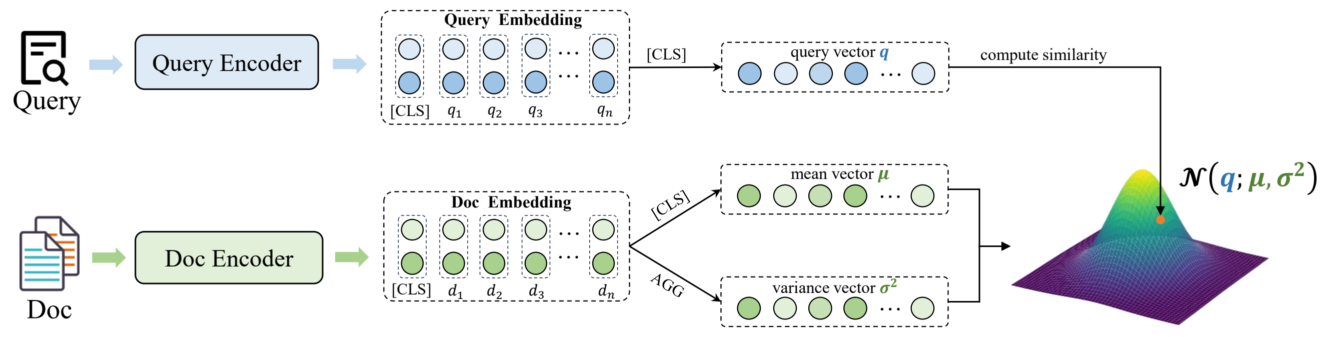
Query
Query Encoder
Query Embedding
[CLS] q_1 q_2 q_3 q_n
[CLS]
query vector $\boldsymbol{q}$
compute similarity
Doc
Doc Encoder
Doc Embedding
[CLS] d_1 d_2 d_3 d_n
[CLS]
AGG
mean vector $\boldsymbol{\mu}$
variance vector $\boldsymbol{\sigma}^2$
$\mathcal{N}(\boldsymbol{q};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$

Figure 2: The overall of DDR framework. The query vector and the mean of the document distribution are both directly represented by the [CLS] embedding, while the variance vector for the document is derived by aggregating all tokens using an attention mechanism.

Similarity Computation

Our method cannot directly use vector dot product to calculate similarity as before. In the DDR framework, we determine the relevance between a document and a query by evaluating the PDF of the document's distribution at the point where the query vector is located. For a given document d, its distribution is modeled as a multivariate Gaussian distribution. The PDF of this distribution is given by:

$$f_d(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^k|\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \tag{12}$$

where $\boldsymbol{\mu}$ represents the mean vector and $\mathbf{\Sigma}$ denotes the non-singular diagonal covariance matrix of the multivariate Gaussian distribution.

Using this PDF, we can compute the probability density value $f_d(\mathbf{q})$ of the query vector $\mathbf{q}$ under the distribution of document d. This probability density value can be interpreted as the relevance score between the query and the document. Specifically, the relevance score $S(d,q)$ can be defined as follows:

$$S(d,q) = f_d(\mathbf{q}). \tag{13}$$

By substituting the query $\mathbf{q}$ into the aforementioned PDF, we obtain:

$$S(d,q) = \frac{1}{\sqrt{(2\pi)^k|\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{q}-\boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{q}-\boldsymbol{\mu})\right). \tag{14}$$

To simplify the computation, we can apply the logarithm to the above formula, transforming the product and exponential operations into addition, thereby improving numerical stability. The logarithmic relevance score $\log S(d,\mathbf{q})$ can be expressed as:

$$-\frac{k}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{\Sigma}| - \frac{1}{2}(\mathbf{q}-\boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{q}-\boldsymbol{\mu}). \tag{15}$$

This form offers better robustness in practical computations because it avoids directly handling extremely small probability density values while maintaining the relative magnitude relationships of the relevance scores.

Training

Knowledge distillation methods are widely used in dense retrieval (Qu et al. 2021; Santhanam et al. 2022; Hofstätter et al. 2021). Following (Zamani and Bendersky 2023), we adopt the cross-encoder architecture to implement the teacher, which takes as input the concatenation of query and passage and models the semantic interaction between query and passage representations. For each query q, let D_q denote the set of documents used for knowledge distillation. The loss function is defined as follows:

$$\mathcal{L} = \sum_{d_i \in D_q} P_{\mathrm{t}}(q,d_i)\log\frac{P_{\mathrm{t}}(q,d_i)}{P_{\mathrm{s}}(q,d_i)}, \tag{16}$$

where $P_{\mathrm{t}}(q,d_i)$ and $P_{\mathrm{s}}(q,d_i)$ represent the normalized probabilities from the re-ranker teacher and the retriever student, respectively. For each query q, D_q includes:

- positive samples from labeled data
- The top 100 BM25 negative samples for query q
- The top 100 hard negative samples for query q retrieved by student model every epoch

Within this framework, we enable the student model to learn the semantic interaction information between queries and passages from the teacher model, thereby enhancing its performance in information retrieval tasks.

Efficient Search

In the search phase, directly using Equation (15) to calculate similarity scores requires computing the query against each document. To meet the high efficiency demands of online search scenarios (Sun et al. 2024), many Approximate Nearest Neighbor (ANN) algorithms, such as PQ (Jegou, Douze, and Schmid 2010) and HNSW (Malkov and Yashunin 2018), have been employed. However, current ANN algorithms only support similarity functions such as dot product, negative Euclidean distance. To apply ANN to the DDR model, we first transform equation (15) as follows:

$$-\frac{k}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{k}\log(\sigma_i^2) - \frac{1}{2}\sum_{i=1}^{k}\frac{(q_i-\mu_i)^2}{\sigma_i^2}, \tag{17}$$

where q_i represents the i-th component of the query vector $\mathbf{q}$, and u_i and σ_i^2 denote the i-th components of the document mean vector $\boldsymbol{\mu}$ and variance vector $\boldsymbol{\sigma^2}$, respectively.

Since our focus is solely on the relative magnitudes of the scores, we eliminatethe constant terms from equation (17) and simplify as follows:

$$\begin{aligned} & -\frac{1}{2}\sum_{i=1}^{k}\log(\sigma_i^2) - \frac{1}{2}\sum_{i=1}^{k}\frac{(q_i-\mu_i)^2}{\sigma_i^2} \\ =& -\frac{1}{2}\sum_{i=1}^{k}\log(\sigma_i^2) - \frac{1}{2}\sum_{i=1}^{k}\frac{q_i^2}{\sigma_i^2} + \sum_{i=1}^{k}\frac{q_i\mu_i}{\sigma_i^2} - \frac{1}{2}\sum_{i=1}^{k}\frac{\mu_i^2}{\sigma_i^2} \\ =& -\frac{1}{2}\sum_{i=1}^{k}\left(log(\sigma_i^2) + \frac{\mu_i^2}{\sigma_i^2}\right) + \sum_{i=1}^{k}\frac{q_i\mu_i}{\sigma_i^2} - \frac{1}{2}\sum_{i=1}^{k}\frac{q_i^2}{\sigma_i^2} \end{aligned} \tag{18}$$

We represent the query and document in the following ways respectively:

$$\overrightarrow{v_q} = [1, q_1, q_2, \cdots, q_k, q_1^2, q_2^2, \cdots, q_k^2], \tag{19}$$

$$\overrightarrow{v_d} = [\gamma, \frac{\mu_1}{\sigma_1^2}, \frac{\mu_2}{\sigma_2^2}, \cdots, \frac{\mu_k}{\sigma_k^2}, \frac{-1}{2\sigma_1^2}, \frac{-1}{2\sigma_2^2}, \cdots, \frac{-1}{2\sigma_k^2}], \tag{20}$$

where $\gamma = -\frac{1}{2}\sum_{i=1}^{k}\left(\log(\sigma_i^2) + \frac{\mu_i^2}{\sigma_i^2}\right)$.

In this manner, the dot product of $\overrightarrow{v_q}$ and $\overrightarrow{v_d}$ is equivalent to the retrieval score as defined in equation (18). Besides $\overrightarrow{v_q} \in \mathbb{R}^{1\times(2k+1)}$ is independent of the document, whereas $\overrightarrow{v_d} \in \mathbb{R}^{1\times(2k+1)}$ is independent of the query. Consequently, we can utilize existing approximate nearest neighbor algorithms, such as HNSW (Malkov and Yashunin 2018), and tools like FAISS (Johnson, Douze, and Jégou 2019), to index all $\overrightarrow{v_d}$ vectors, facilitating efficient retrieval of any query vector $\overrightarrow{v_q}$.

Discussion

In this section, we will discuss the differences of DDR compared to existing methods.

In single vector retrieval methods, the similarity score between a query vector $\overrightarrow{v_q} = [q_1, q_2, \cdots, q_k]$ and a document vector $\overrightarrow{v_d} = [d_1, d_2, \cdots, d_k]$ is calculated using the dot product formula:

$$\text{sim}(q, d) = \sum_{i=1}^{k} q_i d_i. \tag{21}$$

The equation (21) can be interpreted as follows: the document d defines a *"plane"* represented by the function $f(\boldsymbol{x}) = \sum_{i=1}^{k} d_i x_i$. The similarity score between the query and the document is determined by the value of the query vector $\overrightarrow{v_q}$ on the *"plane"*, which is represented by $f(\overrightarrow{v_q})$.

However, our proposed DDR enables the document d to define a *"quadratic surface"*:

$$g(\boldsymbol{x}) = \gamma + \sum_{i=1}^{k}\left(\frac{\mu_i}{\sigma_i^2}x_i + \frac{-1}{2\sigma_i^2}x_i^2\right), \tag{22}$$

Dataset	#Queries	#Docs	Avg. Len.
MSMARCO	6,980	8,841,823	56
TREC DL '19	43	8,841,823	56
TREC DL '20	54	8,841,823	56
NQ	3,452	2,681,468	79
FiQA	648	57,638	132
Touché-2020	49	382,545	292
Climate-FEVER	1,535	5,416,593	85

Table 1: Characteristics and statistics of the datasets in our experiments. Here, "Avg. Len." is the abbreviation of average document length.

where $\gamma = -\frac{1}{2}\sum_{i=1}^{k}\left(\log(\sigma_i^2) + \frac{\mu_i^2}{\sigma_i^2}\right)$.

Furthermore, when the covariance matrices of two documents d and d' are both identity matrices, we have:

$$\text{sim}(q, d) - \text{sim}(q, d') = \frac{1}{2}\sum_{i=1}^{k}\left((q_i - \mu_{d'i})^2 - (q_i - \mu_{di})^2\right). \tag{23}$$

This indicates that when the covariance matrix is the identity matrix, our proposed DDR method simplifies to a dense retrieval method that measures similarity using negative Euclidean distance.

Experiments

Datasets and Evaluation

We conducted experiments on MS MARCO (Nguyen et al. 2016), TREC Track 2019 and 2020 (Craswell et al. 2020), followed by additional experiments on zero-shot datasets (Thakur et al. 2021). MS MARCO, a substantial English corpus created by Microsoft, is based on Bing search results and contains approximately 500k labeled queries and 8.8 million passages. The zero-shot datasets we used include NQ, FiQA, Touché-2020, and Climate-FEVER. Their characteristics and statistics are detailed in Table 1.

For evaluation, we employed various performance metrics tailored to each dataset. For MS MARCO, we used Mean Reciprocal Rank at 10 (MRR@10), Recall at 50 (R@50), and Recall at 1000 (R@1k). For the TREC Deep Learning (DL) track and the zero-shot datasets, we adopted Normalized Discounted Cumulative Gain at 10 (NDCG@10) as our primary evaluation metric. These metrics provide a robust framework for assessing the retrieval performance across different datasets and experimental setups.

Experimental Setup

We implemented and trained our model using PyTorch, optimizing the network parameters with the AdamW (Loshchilov and Hutter 2017) optimizer. We applied a linear learning rate schedule with a warmup phase of 1,000 steps, setting the learning rate to 2×10^{-5}. The parameter β was selected from $\{0.1, 0.2, 0.5, 1, 2, 5, 10\}$. The mean and variance vectors of the document distribution are set to

Model	PLM	Params	Storage	MS MARCO DEV MRR@10	MS MARCO DEV R@50	MS MARCO DEV R@1k	TREC-DL'19 NDCG@10	TREC-DL'20 NDCG@10
Some Sparse Retrieval Models (For Reference)								
BM25	-	-	0.67GB	0.185	0.585	0.857	0.497	0.487
DeepCT	-	-	0.67GB	0.243	0.690	0.910	0.550	0.556
SPLADE	-	-	2.60GB	0.340	-	0.965	0.683	0.671
Single Vector Dense Retrieval Models								
DPR	BERT-Base	110M	26GB	0.319	-	0.941	0.611	0.591
ANCE	BERT-Base	110M	26GB	0.330	-	0.959	0.645	0.646
ADORE	BERT-Base	110M	26GB	0.347	-	-	0.683	0.666
RocketQA	ERNIE-Base	110M	26GB	0.370	0.855	0.979	-	-
Contriever-FT	BERT-Base	110M	26GB	-	-	-	0.621	0.632
RocketQAv2	ERNIE-Base	110M	26GB	0.388	0.862	**0.981**	-	-
Margin-MSE	DistilBERT	66M	26GB	0.326	-	0.958	0.687	0.654
TAS-B	DistilBERT	66M	26GB	0.347	-	0.978	0.717	0.686
CLDRD	DistilBERT	66M	26GB	0.382	-	-	0.725	0.687
MRL	DistilBERT	66M	26GB	0.393	0.862†	0.977†	0.738	**0.701**
DDR$_{k=383}$(ours)	DistilBERT	66M	26GB	**0.396**	**0.864**	0.980	**0.739**	0.684
Multi Vector Dense Retrieval Model								
MVR	BERT-Base	110M	208GB	0.349†	0.828†	0.968†	0.687†	0.661†
COIL-full	BERT-Base	110M	79GB	0.353	-	0.967	0.704	0.688
ColBERTv2	BERT-Base	110M	154GB	0.397	0.868	**0.984**	-	-
CITADEL	BERT-Base	110M	81GB	0.399	-	0.981	0.703	**0.702**
DDR(ours)	DistilBERT	66M	52GB	**0.402**	**0.874**	**0.984**	**0.731**	0.698

Table 2: Main results on MS-MARCO passage and TREC. Results with † are from our reproduction, "-" denotes the results that are not applicable or available. Storage refers to the disk space occupied by the Marco corpus index, without quantization.

768 dimensions. For fair comparison with existing single-vector models, we add dense projection layers on mean vector and variance vector to make their dimensions to be $\frac{768}{2} - 1 = 383$. This configuration in the results table is referred to as DDR$_{k=383}$. Following previous work(Zamani and Bendersky 2023), we used the pre-trained checkpoints provided by TAS-B (Hofstätter et al. 2021) for initialization and used DistilBERT (Sanh et al. 2019) as our initial model. We employed a model based on the ELECTRA (Clark et al. 2020) architecture as the re-ranking teacher model[1].

Baseline

We conducted a comprehensive comparison between our proposed approach and the previous state-of-the-art models that utilize single-vector and multi-vector representations. Specifically, the single-vector models we compared include DPR (Karpukhin et al. 2020), ANCE (Xiong et al. 2021), ADORE (Zhan et al. 2021), RocketQA (Qu et al. 2021), RocketQAv2 (Ren et al. 2021), Margin-MSE (Hofstätter et al. 2020), TAS-B (Hofstätter et al. 2021), Contriever-FT (Izacard et al. 2022), CLDRD (Zeng, Zamani, and Vinay 2022), and MRL (Zamani and Bendersky 2023).

The multi-vector models for comparison include MVR (Zhang et al. 2022), COIL (Gao, Dai, and Callan 2021), ColBERTv2 (Santhanam et al. 2022), and CITADEL (Li et al. 2023). Among these, CITADEL stands out for its exceptional performance, having achieved state-of-the-art results across a variety of datasets.

We also compared with some sparse retrieval models such as BM25 (Robertson et al. 1995), DeepCT (Dai and Callan 2019), docT5query (Nogueira, Lin, and Epistemic 2019) and SPLADE (Formal, Piwowarski, and Clinchant 2021).

Main Results

We compared our model with previous state-of-the-art models. As shown in table 2, all dense retrieval models significantly outperform BM25, DeepCT and SPLADE, highlighting the superiority of these methods for in-domain passage retrieval tasks. For single-vector models, DDR$_{k=383}$ not only matches the index storage space requirements of standard single-vector models but also exceeds all baseline models in terms of MRR@10 and R@50 metrics for the MS MARCO DEV dataset. Among multi-vector models, DDR utilizes the least storage space (with MVR requiring four times the storage of DDR) yet exceeds all other multi-vector models. This demonstrates that our DDR model can address the multi-view problem while avoiding the shortcomings of multi-vector approaches.

In the TREC DL dataset, there are instances where our model exhibits slightly lower performance. It's important to note that this dataset includes only a limited number of test queries: 43 for TREC DL '19 and 54 for TREC DL '20. We observed that using different random seeds results in approximately 2% variation on this test set. Consequently, we be-

[1]https://huggingface.co/intfloat/simlm-msmarco-reranker

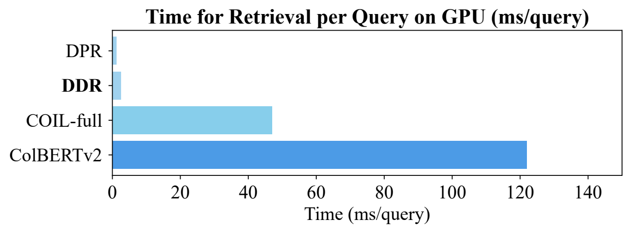

Figure 3: Retrieval latency per query on GPU as measured on the MS MARCO Passages development dataset.

Model	NQ	FiQA	Touché-2020	Climate-FEVER
Sparse Retrieval Models (For Reference)				
BM25	0.329	0.236	0.367	0.213
DeepCT	0.188	0.191	0.156	0.066
DocT5query	0.399	0.291	0.347	0.201
Dense Retrieval Models				
DPR	0.474	0.112	0.131	0.148
ANCE	0.446	0.295	0.284	0.198
TAS-B	0.463	0.300	0.173	**0.228**
RocketQAv2	0.505	0.302	0.247	0.180
CITADEL	0.510	0.298	0.294	0.191
DDR	**0.514**	**0.307**	**0.315**	0.193

Table 3: The zero-shot retrieval results achieved by DDR and the baseline models, evaluated using NDCG@10 metrics. The highest value is highlighted in bold.

lieve that these occasional discrepancies in performance are not statistically significant and do not undermine the overall effectiveness of our model.

Efficiency Analysis

In online search scenarios, achieving high retrieval efficiency is crucial. As shown in Figure 3, single vector models, such as DPR, are the most efficient in terms of inference speed among dense retrieval models and are also the most popular in practical applications. Despite the fact that multi-vector methods (e.g., COIL-full, ColBERTv2) achieve better results compared to single-vector methods, their retrieval latency is significantly higher. The retrieval latency of our proposed DDR is nearly as low as that of single vector models, and it is significantly faster than multi-vector models.

Zero-Shot Retrieval Results

We conducted a zero-shot retrieval evaluation on four diverse datasets from the BEIR benchmark (Thakur et al. 2021): NQ, FiQA, Touché-2020, and Climate-FEVER. In this experiment, we employed the model checkpoint trained on the MS MARCO dataset for evaluation. As shown in Table 3, our model outperforms all single-vector models and CITADEL on NQ, FiQA and Touché-2020 datasets. However, the overall improvements achieved by the best-performing models over traditional sparse retrieval models

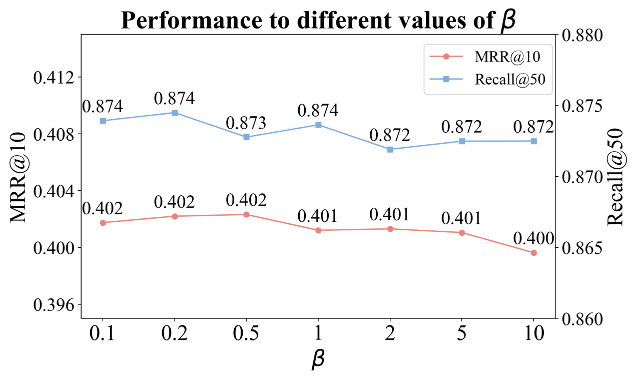

Figure 4: Sensitivity analysis of the parameter β when smoothing the variance vector $\boldsymbol{\sigma}^2$ on MS MARCO dataset.

like BM25 are less significant than those noted in Table 2. This highlights the challenges that neural retrieval models encounter when adapting to different domains.

Parameter Sensitivity Analysis

To assess the sensitivity of DDR performance to the value of β, we conducted a series of experiments where β was systematically varied within the range of 0.1 to 10. As illustrated in Figure 4, the results indicate that the model is not sensitive to the value of β, as its performance remains consistently stable. Even when β is set to 0.1 or 10, the model still maintains good performance. Therefore, the model demonstrates robust and strong performance with β ranging from 0.1 to 10.

Conclusion

In this paper, we proposed modeling queries and documents using vectors and multivariate Gaussian distributions, respectively. The relevance score of a document to a query was then determined by evaluating the likelihood of the query vector within the probability density function representing the document distribution. Through theoretical analysis, we demonstrated that this method can seamlessly integrate with existing efficient ANN algorithms. Extensive experiments validated the effectiveness of our proposed method. We believe this modeling approach is worthy of further exploration in the field of information retrieval.

Acknowledgments

This research was partially supported by grants from the National Natural Science Foundation of China (Grants No.62337001), the Key Technologies R & D Program of Anhui Province (No. 202423k09020039), the Key Projects ofthe National Natural Science Foundation of Universities in Anhui Province (No. 2023AH051302), the University Synergy Innovation Program of Anhui Province (GXXT-2022-042) and the Fundamental Research Funds for the Central Universities.

References

Athiwaratkun, B.; and Wilson, A. 2017. Multimodal Word Distributions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1645–1656.

Clark, K.; Luong, M.-T.; Le, Q. V.; and Manning, C. D. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations*.

Craswell, N.; Mitra, B.; Yilmaz, E.; Campos, D.; and Voorhees, E. M. 2020. Overview of the TREC 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Dai, Z.; and Callan, J. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. *arXiv preprint arXiv:1910.10687*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 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, Volume 1 (Long and Short Papers)*, 4171–4186.

Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; and Hon, H.-W. 2019. Unified language model pre-training for natural language understanding and generation. *Advances in neural information processing systems*, 32.

Erk, K. 2009. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 57–65.

Formal, T.; Piwowarski, B.; and Clinchant, S. 2021. 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*, 2288–2292.

Gao, L.; and Callan, J. 2021. Condenser: a Pre-training Architecture for Dense Retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 981–993.

Gao, L.; Dai, Z.; and Callan, J. 2021. COIL: Revisit Exact Lexical Match in Information Retrieval with Contextualized Inverted List. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 3030–3042.

Geffet, M.; and Dagan, I. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, 107–114.

He, L.; Huang, Z.; Chen, E.; Liu, Q.; Tong, S.; Wang, H.; Lian, D.; and Wang, S. 2023. An efficient and robust semantic hashing framework for similar text search. *ACM Transactions on Information Systems*, 41(4): 1–31.

Hofstätter, S.; Althammer, S.; Schröder, M.; Sertkan, M.; and Hanbury, A. 2020. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv preprint arXiv:2010.02666*.

Hofstätter, S.; Lin, S.-C.; Yang, J.-H.; Lin, J.; and Hanbury, A. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 113–122.

Izacard, G.; Caron, M.; Hosseini, L.; Riedel, S.; Bojanowski, P.; Joulin, A.; and Grave, E. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Trans. Mach. Learn. Res.*

Jegou, H.; Douze, M.; and Schmid, C. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1): 117–128.

Johnson, J.; Douze, M.; and Jégou, H. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3): 535–547.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6769–6781.

Khattab, O.; and Zaharia, M. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 39–48.

Lee, K.; Chang, M.-W.; and Toutanova, K. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 6086–6096.

Li, M.; Lin, S.-C.; Oguz, B.; Ghoshal, A.; Lin, J.; Mehdad, Y.; Yih, W.-t.; and Chen, X. 2023. CITADEL: Conditional Token Interaction via Dynamic Lexical Routing for Efficient and Effective Multi-Vector Retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 11891–11907.

Li, R.; He, L.; Liu, Q.; Zhao, Y.; Zhang, Z.; Huang, Z.; Su, Y.; and Wang, S. 2024a. CONSIDER: Commonalities and Specialties Driven Multilingual Code Retrieval Framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 8679–8687.

Li, R.; Liu, Q.; He, L.; Zhang, Z.; Zhang, H.; Ye, S.; Lu, J.; and Huang, Z. 2024b. Optimizing Code Retrieval: High-Quality and Scalable Dataset Annotation through Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2053–2065.

Liao, Z.; Hou, X.; Lou, D.; Chen, Y.-M.; and Zhang, N. 2023. CAMVR: Context-Adaptive Multi-View Representation Learning for Dense Retrieval. In *2023 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Lin, S.-C.; Yang, J.-H.; and Lin, J. 2021. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, 163–173.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lu, S.; He, D.; Xiong, C.; Ke, G.; Malik, W.; Dou, Z.; Bennett, P.; Liu, T.-Y.; and Overwijk, A. 2021. Less is More: Pretrain a Strong Siamese Encoder for Dense Text Retrieval Using a Weak Decoder. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2780–2791.

Luan, Y.; Eisenstein, J.; Toutanova, K.; and Collins, M. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9: 329–345.

Malkov, Y. A.; and Yashunin, D. A. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4): 824–836.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. Ms marco: A human-generated machine reading comprehension dataset.

Nogueira, R.; Lin, J.; and Epistemic, A. 2019. From doc2query to docTTTTTquery. *Online preprint*, 6(2).

Qu, Y.; Ding, Y.; Liu, J.; Liu, K.; Ren, R.; Zhao, W. X.; Dong, D.; Wu, H.; and Wang, H. 2021. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 5835–5847.

Ren, R.; Qu, Y.; Liu, J.; Zhao, W. X.; She, Q.; Wu, H.; Wang, H.; and Wen, J.-R. 2021. RocketQAv2: A Joint Training Method for Dense Passage Retrieval and Passage Reranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2825–2835.

Robertson, S. E.; Walker, S.; Jones, S.; Hancock-Beaulieu, M. M.; Gatford, M.; et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp*, 109: 109.

Sanh, V.; Debut, L.; Chaumond, J.; and Wolf, T. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Santhanam, K.; Khattab, O.; Saad-Falcon, J.; Potts, C.; and Zaharia, M. 2022. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 3715–3734.

Sun, Y.; Zhu, H.; Wang, L.; Zhang, L.; and Xiong, H. 2024. Large-scale online job search behaviors reveal labor market shifts amid COVID-19. *Nature Cities*, 1(2): 150–163.

Tang, H.; Sun, X.; Jin, B.; Wang, J.; Zhang, F.; and Wu, W. 2021. Improving Document Representations by Generating Pseudo Query Embeddings for Dense Retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 5054–5064.

Thakur, N.; Reimers, N.; Rücklé, A.; Srivastava, A.; and Gurevych, I. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Vilnis, L.; and McCallum, A. 2014. Word representations via gaussian embedding. *arXiv preprint arXiv:1412.6623*.

Wang, L.; Yang, N.; Huang, X.; Jiao, B.; Yang, L.; Jiang, D.; Majumder, R.; and Wei, F. 2023. SimLM: Pre-training with Representation Bottleneck for Dense Passage Retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2244–2258.

Wang, Y.; Wang, J.; Chen, B.; Zeng, Z.; and Xia, S.-T. 2024a. GMMFormer: Gaussian-Mixture-Model Based Transformer for Efficient Partially Relevant Video Retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 5767–5775.

Wang, Z.; Wang, P.; Liu, K.; Wang, P.; Fu, Y.; Lu, C.-T.; Aggarwal, C. C.; Pei, J.; and Zhou, Y. 2024b. A Comprehensive Survey on Data Augmentation. *arXiv preprint arXiv:2405.09591*.

Xiao, S.; Liu, Z.; Shao, Y.; and Cao, Z. 2022. Retro-MAE: Pre-Training Retrieval-oriented Language Models Via Masked Auto-Encoder. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 538–548.

Xiong, L.; Xiong, C.; Li, Y.; Tang, K.-F.; Liu, J.; Bennett, P. N.; Ahmed, J.; and Overwijk, A. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*.

Zamani, H.; and Bendersky, M. 2023. Multivariate representation learning for information retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 163–173.

Zeng, H.; Zamani, H.; and Vinay, V. 2022. Curriculum learning for dense retrieval distillation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1979–1983.

Zhan, J.; Mao, J.; Liu, Y.; Guo, J.; Zhang, M.; and Ma, S. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1503–1512.

Zhang, S.; Liang, Y.; Gong, M.; Jiang, D.; and Duan, N. 2022. Multi-View Document Representation Learning for Open-Domain Dense Retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5990–6000.

Zhao, H.; Zhao, C.; Zhang, X.; Liu, N.; Zhu, H.; Liu, Q.; and Xiong, H. 2023. An ensemble learning approach with gradient resampling for class-imbalance problems. *INFORMS Journal on Computing*, 35(4): 747–763.