Multivariate Representation Learning for Information Retrieval

Hamed Zamani
University of Massachusetts Amherst
United States
zamani@cs.umass.edu

Michael Bendersky
Google Research
United States
bemike@google.com

ABSTRACT

Dense retrieval models use bi-encoder network architectures for learning query and document representations. These representations are often in the form of a vector representation and their similarities are often computed using the dot product function. In this paper, we propose a new representation learning framework for dense retrieval. Instead of learning a vector for each query and document, our framework learns a multivariate distribution and uses negative multivariate KL divergence to compute the similarity between distributions. For simplicity and efficiency reasons, we assume that the distributions are multivariate normals and then train large language models to produce mean and variance vectors for these distributions. We provide a theoretical foundation for the proposed framework and show that it can be seamlessly integrated into the existing approximate nearest neighbor algorithms to perform retrieval efficiently. We conduct an extensive suite of experiments on a wide range of datasets, and demonstrate significant improvements compared to competitive dense retrieval models.

CCS CONCEPTS

• **Information systems** → **Document representation**; **Query representation**; **Retrieval models and ranking**.

KEYWORDS

Neural information retrieval; dense retrieval; learning to rank; approximate nearest neighbor search

ACM Reference Format:
Hamed Zamani and Michael Bendersky. 2023. Multivariate Representation Learning for Information Retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3539618.3591740

1 INTRODUCTION

Utilizing implicit or explicit relevance labels to learn retrieval models, also called learning-to-rank models, is at the core of information retrieval research. Due to efficiency and even sometimes effectiveness reservations, learning-to-rank models have been mostly used for *reranking* documents retrieved by an efficient retrieval model, such as BM25 [39]. Therefore, the performance of learning-to-rank models was bounded by the quality of candidate documents selected for reranking. In 2018, the SNRM model [55] has revolutionized the way we look at learning-to-rank models by arguing that bi-encoder neural networks can be used for representing queries and documents, and document representations can be then indexed for efficient retrieval at query time. The model applied learned latent sparse representations for queries and documents, and indexed the document representations using an inverted index. In 2020, the DPR model [23] demonstrated that even bi-encoder networks with dense representations can be used for efficient retrieval. They took advantage of approximate nearest neighbor algorithms for indexing dense document representations. This category of models, often called *dense retrieval* models, has attracted much attention and led to state-of-the-art performance on a wide range of retrieval tasks [18, 24, 37, 53, 57].

Existing sparse and dense representation learning models can be seen as instantiations of Salton et al.'s vector space models [41], i.e., queries and documents are represented using vectors and relevance is defined using vector similarity functions, such as inner product or cosine similarity. Such approaches suffer from a major shortcoming: they do not represent the model's *confidence* on the learned representations. Inspired by prior work on modeling uncertainty in information retrieval (e.g., [7, 8, 52]), this paper builds upon the following hypothesis:

> *Neural retrieval models would benefit from modeling uncertainty (or confidence) in the learned query and document representations.*

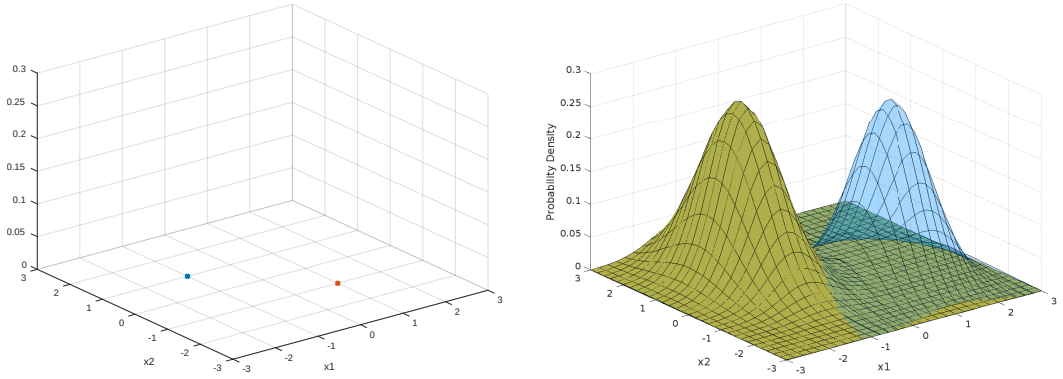
Therefore, we propose a generic framework that represents each query and document using a multivariate distribution, called the MRL framework. In other words, instead of representing queries and documents using k-dimensional vectors, we can assign a probability to each point in this k-dimensional space; the higher the probability, the higher the confidence that the model assigns to each point. For $k = 2$, Figure 1(a) depicts the representation of a query and a document in existing single-vector dense retrieval models.[1] On the other hand, Figure 1(b) demonstrates the representations that we envision for queries and documents.

To reduce the complexity of the model, we assume that the representations are multivariate normal distributions with a diagonal covariance matrix; meaning that the representation dimensions are orthogonal and independent. With this assumption, we learn two k-dimensional vectors for each query or document: a mean vector and a variance vector. In addition to uncertainty, such probabilistic modeling can implicitly represent breadth of information in queries and documents. For instance, a document that covers multiple topics and potentially satisfies a diverse set of information needs may be represented by a multivariate distribution with large variance values.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07.
https://doi.org/10.1145/3539618.3591740

[1] The third dimension is only used for consistent presentation. One can consider the probability of 1 for one point in the two-dimensional space and zero elsewhere.



(a) Representation learning in existing single-vector dense retrieval models

(b) Representation learning in MRL

Figure 1: Existing dense retrieval methods use a vector to represent any input. Figure 1(a) demonstrates example representations they learn for two inputs (e.g., a query and a document). The proposed framework learns multivariate distributions to represent each input, which is depicted in Figure 1(b).

MRL uses negative multivariate Kullback-Leibler (KL) divergence between query and document representations to compute the relevance scores. We prove that the relevance scores can be computed efficiently by proposing solutions that can be implemented using existing approximate nearest neighbor search algorithms. We also demonstrate that one can simply implement the MRL framework using existing pre-trained large language models, such as BERT [13].

We show that an implementation of MRL that uses a single vector with 768 dimensions to represent multivariate representations for each query and document significantly outperforms existing single vector dense retrieval models on several standard text retrieval benchmarks. MRL also often outperforms ColBERTv2 [43], a state-of-the-art multi vector dense retrieval model, while using significantly less storage and having significantly lower query latency. We further demonstrate that MRL also performs effectively in zero-shot settings when applied to unseen domains. Besides, we also demonstrate that the norm of variance vectors learned by MRL are a strong indicator of the retrieval effectiveness and can be used as a pre-retrieval query performance predictor.

We believe that MRL smooths the path towards developing more advanced probabilistic dense retrieval models and its applications can be extended to recommender systems, conversational systems, and a wide range of retrieval-enhanced machine learning models.

2 RELATED WORK

Variance of retrieval performance among different topics has been a long-standing research theme in the information retrieval community. For instance, TREC 2004 Robust Track organizers noted that solely optimizing the average metric aggregates (e.g., MAP) "further improves the effectiveness of the already-effective topics, sometimes at the expense of the poor performers" [49]. Moreover, identifying poorly performing topics is hard, and failure to do so leads to degraded user perception of the retrieval system as "an individual user does not see the average performance of the system, but only the effectiveness of the system on his or her requests" [49]. These insights led the information retrieval community to consider *query performance prediction* [5] – a notion that certain signals can predict the performance of a search query. Such predictions can be helpful in guiding the retrieval system in taking further actions as needed for more difficult queries, e.g., suggesting alternative query reformulations [1].

A degree of query ambiguity with respect to the underlying corpus has been shown to be a valuable predictor of poor performance of search queries [11]. Therefore, dealing with retrieval uncertainty has been proposed as a remedy. For instance, Collins-Thompson and Callan [8] propose estimating query uncertainty by repeatedly fitting a Dirichlet distribution over bootstrap samples from the top-k retrieved documents. They show that a Bayesian combination of multiple boostrap samples (which takes into account sample variance) leads to both significantly better retrieval metrics, and better retrieval robustness (less queries hurt by the query expansion methods). In a related vein, Zhu et al. [62] develop a risk-aware language model based on the Dirichlet distribution (as a conjugate prior to the multinomial distribution). They use the variance of the Dirichlet distribution for adjusting the risk in the final ranking score (i.e., revising the relevance estimates downwards in face of high variance).

The idea of risk adjustment inspired by the financial investment literature was further developed by Wang and Zhu into the portfolio theory for information retrieval [52]. Portfolio theory generalizes the probability ranking principle (PRP) by considering both the uncertainty of relevance predictions and correlations between retrieved documents. It also demonstrates that one way to address uncertainty is via diversification [6]. The portfolio theory-based approach to retrieval has since been applied in several domains including recommendation [44], quantum-based information retrieval [63], and computational advertising [59], among others.

While, as this prior research shows, there has been an extensive exploration of risk and mean-variance trade-offs in the statistical language models for information retrieval, there has been so far much less discussion of these topics in the context of neural (*aka*

dense) models for retrieval. As a notable exception to this, Cohen et al. [7] recently proposed a Bayesian neural relevance model, where a posterior is approximated using Monte Carlo sampling based on drop-out [14]. A similar approach was proposed by Penha and Hauff [34] in the context of conversational search. These approaches, which employ variational inference at training time, can only be applied for *reranking*. In contrast, in this work we model uncertainty at the level of query and document representations, and demonstrate how such representations can be efficiently and effectively used for *retrieval* using any of the existing approximate nearest neighbor methods.

Outside the realm of information retrieval research, various forms of representations that go beyond Euclidean vectors have been explored, including order embeddings [46], hyperbolic embeddings [31], and probabilistic box embeddings [47], Such representations have been shown to be effective for various NLP tasks that involve modeling complex relationship or structures. Similar to our work, Vilnis and McCallum [48] used Gaussian distributions for representation learning by proposing Gaussian embeddings for words. In this work, we focus on query and document representations in the retrieval setting.

Some prior work, as a way to achieve semantically richer representations, model queries and documents using a combination of multiple vectors [26, 43, 61]. While such representations were shown to lead to better retrieval effectiveness, they do come at significant computational and storage costs. We demonstrate that our multivariate distribution representations are significantly more efficient than multi-vector ones, while attaining comparable or better performance on a wide range of collections.

3 THE MRL FRAMEWORK

Existing single vector dense retrieval models uses a k-dimensional latent vector to represent each query or each query token [17, 23, 53, 57]. We argue that these dense retrieval models can benefit from modeling uncertainty in representation learning. That means the model may produce a representation for a clear navigational query with high confidence, while it may have lower confidence in representing an ambiguous query. Same argument applies to the documents. However, the existing frameworks for dense retrieval do not model such confidence or uncertainty in representations. In this paper, we present MRL– a generic framework for modeling uncertainty in representation learning for information retrieval. MRL models each query (or document) using a *k-variate distribution* – a group of k continuous random variables using which we can compute the probability of any given vector in a k-dimensional space being a representation of the input query (or document). Formally, MRL encodes each query q and each document d as follows:

$$\begin{aligned}\mathbf{Q} &= (Q_1, Q_2, \cdots, Q_k)^\intercal = \textsc{Encoder}_\text{Q}(q)\\ \mathbf{D} &= (D_1, D_2, \cdots, D_k)^\intercal = \textsc{Encoder}_\text{D}(d)\end{aligned} \tag{1}$$

where $\textsc{Encoder}_\text{Q}$ and $\textsc{Encoder}_\text{D}$ respectively denote query and document encoders. Each Q_i and D_i is a random variable; thus $\mathbf{Q}$ and $\mathbf{D}$ are k-variate distributions representing the query and the document. The superscript $^\intercal$ denotes the transpose of the vector.

In this paper, we assume that $\mathbf{Q}$ and $\mathbf{D}$ both are k-variate normal distributions. The reasons for this assumption are: (1) we can define each k-variate normal distribution using a mean vector and a covariance matrix, (2) lower order distributions (i.e., any combination of the k dimensions) and conditional distributions are also normal, which makes it easily extensible, (3) linear functions of multivariate normal distributions are also multivariate normal, leading to simple aggregation approaches. A k-variate normal distribution can be represented using a $k \times 1$ mean vector $\boldsymbol{M} = (\mu_1, \mu_2, \cdots, \mu_k)^\intercal$ and a $k \times k$ covariance matrix $\boldsymbol{\Sigma}$ as follows: $\mathcal{N}_k(\boldsymbol{M}, \boldsymbol{\Sigma})$. We compute the representations as k *independent* normal distributions, thus **the covariance matrix is diagonal**. Therefore, our representations are modeled as follows:

$$\mathcal{N}_k\left[\begin{pmatrix}\mu_1\\ \mu_2\\ \vdots\\ \mu_k\end{pmatrix}, \begin{pmatrix}\sigma_1^2 & 0 & \dots & 0\\ 0 & \sigma_2^2 & \dots & 0\\ \vdots & & \ddots & \\ 0 & 0 & \dots & \sigma_k^2\end{pmatrix}\right] \tag{2}$$

With this formulation, we can re-write Equation (1) as follows:

$$\begin{aligned}\mathbf{Q} &\sim \mathcal{N}_k(\boldsymbol{M}_Q, \boldsymbol{\Sigma}_Q), \qquad \boldsymbol{M}_Q, \boldsymbol{\Sigma}_Q = \textsc{Encoder}_\text{Q}(q)\\ \mathbf{D} &\sim \mathcal{N}_k(\boldsymbol{M}_D, \boldsymbol{\Sigma}_D), \qquad \boldsymbol{M}_D, \boldsymbol{\Sigma}_D = \textsc{Encoder}_\text{D}(d)\end{aligned} \tag{3}$$

where $\boldsymbol{M}_Q = (\mu_{q1}, \mu_{q2}, \cdots, \mu_{qk})^\intercal$, $\boldsymbol{\Sigma}_Q = (\sigma_{q1}^2, \sigma_{q2}^2, \cdots, \sigma_{qk}^2)^\intercal \times I_k$, $\boldsymbol{M}_D = (\mu_{d1}, \mu_{d2}, \cdots, \mu_{dk})^\intercal$, and $\boldsymbol{\Sigma}_D = (\sigma_{d1}^2, \sigma_{d2}^2, \cdots, \sigma_{dk}^2)^\intercal \times I_k$. Therefore, it is safe to claim that MRL uses large language models to learn a k-dimensional mean vector and a k-dimensional variance vector for representing each input query and document. This representation for a query and a document is plotted in Figure 1(b) ($k = 2$ in the plot).

Using the flexible modeling offered by the MRL framework, we can compute the probability of any k dimensional vector representing each query or document. In more detail, the probability of vector $\mathbf{x} = (x_1, x_2, \cdots, x_k)^\intercal$ being generated from the k-variate normal distribution in Equation (2) is equal to:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{k}{2}} \det(\boldsymbol{\Sigma})^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{M})^\intercal \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{M})\right) \tag{4}$$

where $\det(\cdot)$ denotes the determinant of the given matrix. This formulation enables us to compute the probability of any k-dimensional vector being a representation for each query and document.

Once the queries and documents are represented, MRL computes the relevance score for a pair of query and document using the negative Kullback-Leibler divergence (negative KL divergence) between two k-variate distributions: $-\text{KLD}_k(\mathbf{Q} \parallel \mathbf{D})$. The KL divergence can be computed as follows:

$$\begin{aligned}\text{KLD}_k(\mathbf{Q} \parallel \mathbf{D}) &= \mathbb{E}_\mathbf{Q}\left[\log \frac{\mathbf{Q}}{\mathbf{D}}\right] = \mathbb{E}_\mathbf{Q}\left[\log \mathbf{Q} - \log \mathbf{D}\right]\\ &= \frac{1}{2}\mathbb{E}_\mathbf{Q}\Big[-\log\det(\boldsymbol{\Sigma}_Q) - (\mathbf{x} - \boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}(\mathbf{x} - \boldsymbol{M}_Q)\\ &\qquad + \log\det(\boldsymbol{\Sigma}_D) + (\mathbf{x} - \boldsymbol{M}_D)^\intercal \boldsymbol{\Sigma}_D^{-1}(\mathbf{x} - \boldsymbol{M}_D)\Big]\\ &= \frac{1}{2}\log\frac{\det(\boldsymbol{\Sigma}_D)}{\det(\boldsymbol{\Sigma}_Q)} - \frac{1}{2}\mathbb{E}_\mathbf{Q}\left[(\mathbf{x} - \boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}(\mathbf{x} - \boldsymbol{M}_Q)\right]\\ &\qquad + \frac{1}{2}\mathbb{E}_\mathbf{Q}\left[(\mathbf{x} - \boldsymbol{M}_D)^\intercal \boldsymbol{\Sigma}_D^{-1}(\mathbf{x} - \boldsymbol{M}_D)\right]\end{aligned} \tag{5}$$

Since $(\mathbf{x}-\boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}(\mathbf{x}-\boldsymbol{M}_Q)$ is a real scalar (i.e., $\in \mathbb{R}$, it is equivalent to $\text{tr}\{(\mathbf{x}-\boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}(\mathbf{x}-\boldsymbol{M}_Q)\}$, where $\text{tr}\{\cdot\}$ denotes the trace of the given matrix. Since $\text{tr}\{XY\} = \text{tr}\{YX\}$ for any two matrices $X \in \mathbb{R}^{a\times b}$ and $Y \in \mathbb{R}^{b\times a}$, we have:

$$\text{tr}\{(\mathbf{x}-\boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}(\mathbf{x}-\boldsymbol{M}_Q)\} = \text{tr}\{(\mathbf{x}-\boldsymbol{M}_Q)(\mathbf{x}-\boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}\}$$

Therefore, since $\mathbb{E}[\text{tr}\{X\}] = \text{tr}\{\mathbb{E}[X]\}$ for any square matrix X, we can rewrite Equation (5) as follows:

$$\begin{aligned}\text{KLD}_k(\mathbf{Q} \parallel \mathbf{D}) = \frac{1}{2}\log\frac{\det(\boldsymbol{\Sigma}_D)}{\det(\boldsymbol{\Sigma}_Q)} \\ - \frac{1}{2}\text{tr}\left\{\mathbb{E}_\mathbf{Q}\left[(\mathbf{x}-\boldsymbol{M}_Q)(\mathbf{x}-\boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}\right]\right\} \\ + \frac{1}{2}\text{tr}\left\{\mathbb{E}_\mathbf{Q}\left[(\mathbf{x}-\boldsymbol{M}_D)^\intercal \boldsymbol{\Sigma}_D^{-1}(\mathbf{x}-\boldsymbol{M}_D)\right]\right\} \quad (6)\end{aligned}$$

Given the definition of the covariance matrix, we know that $\boldsymbol{\Sigma}_Q = \mathbb{E}_\mathbf{Q}\left[(\mathbf{x}-\boldsymbol{M}_Q)(\mathbf{x}-\boldsymbol{M}_Q)^\intercal\right]$. Therefore, we have:

$$\begin{aligned}\text{tr}\{\mathbb{E}_\mathbf{Q}\left[(\mathbf{x}-\boldsymbol{M}_Q)(\mathbf{x}-\boldsymbol{M}_Q)^\intercal \boldsymbol{\Sigma}_Q^{-1}\right]\} \\ = \text{tr}\left\{\mathbb{E}_\mathbf{Q}\left[\boldsymbol{\Sigma}_Q\boldsymbol{\Sigma}_Q^{-1}\right]\right\} = \text{tr}\{I_k\} = k \quad (7)\end{aligned}$$

In addition, since Q is a multivariate normal distribution, for any matrix A we have $\mathbb{E}_Q[\mathbf{x}^\intercal A\mathbf{x}] = \text{tr}\{A\boldsymbol{\Sigma}_Q\} + \boldsymbol{M}_Q^\intercal A\boldsymbol{M}_Q$. This results in:

$$\begin{aligned}\text{tr}\left\{\mathbb{E}_\mathbf{Q}\left[(\mathbf{x}-\boldsymbol{M}_D)^\intercal \boldsymbol{\Sigma}_D^{-1}(\mathbf{x}-\boldsymbol{M}_D)\right]\right\} = \\ \text{tr}\{\boldsymbol{\Sigma}_D^{-1}\boldsymbol{\Sigma}_Q\} + (\boldsymbol{M}_Q-\boldsymbol{M}_D)^\intercal \boldsymbol{\Sigma}_D^{-1}(\boldsymbol{M}_Q-\boldsymbol{M}_D) \quad (8)\end{aligned}$$

Using Equations (7) and (8), we can rewrite Equation (6) as follows:

$$\frac{1}{2}\left[\log\frac{\det(\boldsymbol{\Sigma}_D)}{\det(\boldsymbol{\Sigma}_Q)} - k + \text{tr}\{\boldsymbol{\Sigma}_D^{-1}\boldsymbol{\Sigma}_Q\} + (\boldsymbol{M}_Q-\boldsymbol{M}_D)^\intercal \boldsymbol{\Sigma}_D^{-1}(\boldsymbol{M}_Q-\boldsymbol{M}_D)\right] \quad (9)$$

This equation can be further simplified. Based on our earlier assumption that the covariance matrices are diagonal, then $\det(\boldsymbol{\Sigma}_D) = \prod_{i=1}^k \sigma_{di}^2$. In addition, since we are using KL divergence to rank documents, constant values (e.g., k) or document independent values (e.g., $\log\det(\boldsymbol{\Sigma}_Q)$) do not impact document ordering. Therefore, there can be omitted and we can use the following equation to rank the documents using negative multivariate KL-divergence:

$$\begin{aligned}\text{score}(q,d) = -\text{KLD}_k(\mathbf{Q} \parallel \mathbf{D}) \\ \stackrel{\text{rank}}{=} -\frac{1}{2}\left[\sum_{i=1}^k \log\sigma_{di}^2 + \frac{\prod_{i=1}^k \sigma_{qi}^2}{\prod_{i=1}^k \sigma_{di}^2} + \sum_{i=1}^k \frac{(\mu_{qi}-\mu_{di})^2}{\sigma_{di}^2}\right] \quad (10)\end{aligned}$$

In Section 4.3, we explain how to efficiently compute this scoring function using approximate nearest neighbor methods.

4 MRL IMPLEMENTATION

In this section, we first describe our network architecture for implementing the the query and document encoders Encoder_Q and Encoder_D (see Equation (3)). Next, we explain our optimization approach for training the models.

4.1 Encoder Architecture

Pretrained large language models (LLMs) have demonstrated promising results in various information retrieval tasks [10, 17, 33, 57]. Therefore, we decide to adapt existing pretrained LLMs to learn a k-variate normal distribution for each given input. As described above, each k-variate normal distribution can be modeled using a k-dimensional mean vector and a k-dimensional variance vector. We use two special tokens as the input of pretrained LLMs to obtain these two vectors. For example, we convert an input query '`neural information retrieval`' to '`[CLS] [VAR] neural information retrieval [SEP]`' and feed it to BERT-base [13]. Let $\vec{q}_{\texttt{[CLS]}} \in \mathbb{R}^{1\times 768}$ and $\vec{q}_{\texttt{[VAR]}} \in \mathbb{R}^{1\times 768}$ respectively denote the representations produced by BERT for the first two tokens `[CLS]` and `[VAR]`. We obtain the mean and variance vectors for query q using two separate dense projection layers on $\vec{q}_{\texttt{[CLS]}}$ and $\vec{q}_{\texttt{[VAR]}}$, as follows:

$$\begin{aligned}\boldsymbol{M}_Q &= \vec{q}_{\texttt{[CLS]}} W_M \\ \boldsymbol{\Sigma}_Q &= \frac{1}{\beta}\log(1+\exp(\beta.\vec{q}_{\texttt{[VAR]}} W_\Sigma)).I_k \quad (11)\end{aligned}$$

where $W_M \in \mathbb{R}^{768\times k}$ and $W_\Sigma \in \mathbb{R}^{768\times k}$ are the projection layer parameters. To compute the diagonal covariance matrix, we use the softplus function (i.e., $\frac{1}{\beta}\log(1+\exp(\beta.x))$) for the following reasons: (1) it is continuous and differentiable, thus it can be used in gradient descent-based optimization, (2) softplus ensures that variance values are always positive, (3) zero is its lower bound ($\lim_{x\to-\infty}\frac{1}{\beta}\log(1+\exp(\beta.x)) = 0$), yet it is never equal to zero, thus it does not cause numeric instability in KL-divergence calculation (see Equation (10)), and (4) for large x values, it can be approximated using a linear function, i.e., $\lim_{x\to\infty}\frac{1}{\beta}\log(1+\exp(\beta.x)) = x$, ensuring numerical stability for large input values. To better demonstrate its properties, Figure 2 in our experiments plots softplus for various values of β – a hyper-parameter that specifies the softplus formation. The $k\times k$ identity matrix I_k in Equation (11) is used to convert the variance vector to a diagonal covariance matrix.

Note that MRL does not explicitly compute variance, instead learns representations for the `[VAR]` token such that it minimizes the loss function based on negative multivariate KL divergence scoring. Therefore, the model implicitly learns how to represent latent variance vectors.

The mean vector and covariance matrices for document representations are also computed similarly. In our experiments, all parameters (including parameters in BERT and the dense projection layers) are updated and shared between the query and document encoders (i.e., Encoder_Q and Encoder_D).

4.2 Model Training

Recent research has suggested that dense retrieval models can significantly benefit from knowledge distillation [18, 37, 43]. Following these models, we use a BERT-based cross-encoder re-ranking model as the teacher model. Let D_q be a set of documents selected for query q for knowledge distillation. We use the following listwise

loss function for each query q as follows:

$$\sum_{d,d' \in D_q} \mathbb{1}\{y_q^T(d) > y_q^T(d')\} |\frac{1}{\pi_q(d)} - \frac{1}{\pi_q(d')}| \log(1+e^{y_q^S(d') - y_q^S(d)}) \tag{12}$$

where $\pi_q(d)$ denotes the rank of document d in the result list produced by the student dense retrieval model, and $y_q^T(d)$ and $y_q^S(d)$ respectively denote the scores produced by the teacher and the student models for the pair of query q and document d. This knowledge distillation listwise loss function is inspired by LambdaRank [3] and is also used by Zeng et al. [57] for dense retrieval distillation.

For each query q, the document set D_q is constructed based on the following steps:

- D_q includes all positive documents from the relevance judgments (i.e., qrel).
- D_q includes $m_{\text{BM25}} \in \mathbb{R}$ documents from the top 100 documents retrieved by BM25.
- D_q includes $m_{\text{hard}} \in \mathbb{R}$ documents from the top 100 documents retrieved by student model (i.e., negative sampling using the model itself every 5000 steps).

In addition, we take advantage of the other passages in the batch as in-batch negatives. Although in-batch negatives resemble randomly sampled negatives that can be distinguished easily from other documents, it is efficient since passage representations can be reused within the batch [23].

4.3 Efficient Retrieval

Existing dense retrieval models use approximate nearest neighbor (ANN) approaches for efficient retrieval. However, using ANN algorithms in the proposed MRL framework is not trivial. The reason is that MRL uses the negative k-variate KL divergence formulation presented in Equation (10) to compute relevance scores. This is while existing ANN algorithms only support simple similarity functions such as dot product, cosine similarity, or negative Euclidean distance. To address this issue, we convert Equation (10) to a dot product formulation. Let us expand the last term in Equation (10):[2]

$$-\left[\underbrace{\sum_{i=1}^{k} \log \sigma_{di}^2}_{\text{doc prior}} + \frac{\prod_{i=1}^{k} \sigma_{qi}^2}{\prod_{i=1}^{k} \sigma_{di}^2} + \sum_{i=1}^{k} \frac{\mu_{qi}^2}{\sigma_{di}^2} + \underbrace{\sum_{i=1}^{k} \frac{\mu_{di}^2}{\sigma_{di}^2}}_{\text{doc prior}} - \sum_{i=1}^{k} \frac{2\mu_{di}\mu_{qi}}{\sigma_{di}^2}\right] \tag{13}$$

The first and the fourth terms in Equation (13) are document priors, thus they are query independent and can be pre-computed. Therefore, let $\gamma_d = -\sum_{i=1}^{k} (\log \sigma_{di}^2 + \frac{\mu_{di}^2}{\sigma_{di}^2})$ denote the document prior score. Therefore, the scoring function in Equation (10) can be formulated as the dot product of the following two vectors:

$$\vec{q} = \left[1, \Pi_q, \mu_{q1}^2, \mu_{q2}^2, \cdots, \mu_{qk}^2, \mu_{q1}, \mu_{q2}, \cdots, \mu_{qk}\right]$$
$$\vec{d} = \left[\gamma_d, \frac{-1}{\Pi_d}, \frac{-1}{\sigma_{d1}^2}, \frac{-1}{\sigma_{d2}^2}, \cdots, \frac{-1}{\sigma_{dk}^2}, \frac{2\mu_{d1}}{\sigma_{d1}^2}, \frac{2\mu_{d2}}{\sigma_{d2}^2}, \cdots, \frac{2\mu_{dk}}{\sigma_{dk}^2}\right] \tag{14}$$

[2]We drop multiplication to $\frac{1}{2}$ as it does not impact document ordering.

where $\Pi_q = \prod_{i=1}^{k} \sigma_{qi}^2$ and $\Pi_d = \prod_{i=1}^{k} \sigma_{di}^2$ are pre-computed scalars. The dot product of $\vec{q} \in \mathbb{R}^{1\times(2k+2)}$ and $\vec{d} \in \mathbb{R}^{1\times(2k+2)}$ is equal to the retrieval score formulated in Equation (10). More importantly, $\vec{q}$ is document independent and $\vec{d}$ is query independent. Therefore, we can use existing approximate nearest neighbor algorithms, such as HNSW [30], and existing tools, such as FAISS [22], to index all $\vec{d}$ vectors and conduct efficient retrieval for any query vector $\vec{q}$.

5 DISCUSSION

In this section, we attempt to shed some light on the behavior of retrieval using MRL, by providing theoretical answers to the following questions.

> Q1. How does MRL rank two documents with identical covariance matrices?

Let d and d' be two documents, represented by the mean vectors $\boldsymbol{M}_D$ and $\boldsymbol{M}_{D'}$ and identical covariance matrix $\boldsymbol{\Sigma}_D = \boldsymbol{\Sigma}_{D'}$. Therefore, given Equation (10) we have:

$$\text{score}(q,d) - \text{score}(q,d') \overset{\text{rank}}{=} \sum_{i=1}^{k} \left[(\mu_{qi} - \mu_{d'i})^2 - (\mu_{qi} - \mu_{di})^2\right]$$

This shows that in case of identical covariance matrices, MRL assigns a higher relevance score to the document whose mean vector is closest to the query mean vector with respect to *Euclidean distance*.

A remark of this finding is that if the covariance matrix is constant for all documents (i.e., if we ignore uncertainty), MRL can be reduced to existing dense retrieval formulation, where negative Euclidean distance is used to measure vector similarity. Therefore, MRL is a generalized form of this dense retrieval formulation.

> Q2. Popular dense retrieval models use inner product to compute the similarity between query and document vectors. What happens if we use inner product in MRL?

Inner product or dot product cannot be defined for multivariate distributions, however, one can take several samples from the query and document distributions and compute their dot product similarity. Since the query distribution $\mathbf{Q} \sim \mathcal{N}_k(\boldsymbol{M}_Q, \boldsymbol{\Sigma}_Q)$ and the document distribution $\mathbf{D} \sim \mathcal{N}_k(\boldsymbol{M}_D, \boldsymbol{\Sigma}_D)$ are independent, the expected value of their product is:

$$\mathbb{E}[\mathbf{Q} \cdot \mathbf{D}] = \mathbb{E}[\mathbf{Q}] \cdot \mathbb{E}[\mathbf{D}] = \boldsymbol{M}_Q \cdot \boldsymbol{M}_D$$

That means, in expectation, the dot product of samples from multivariate distributions will be equivalent to the dot product of their mean vectors. Therefore, with this formulation (i.e., using expected dot product instead of negative KL divergence) the results produced by MRL will be equivalent to the existing dense retrieval models and representation uncertainties are not considered.

> Q3. Negative KL divergence has been used in the language modeling framework of information retrieval [27]. How is it connected with the proposed MRL framework?

Lafferty and Zhai [27] extended the query likelihood retrieval model of Ponte and Croft [35] by computing negative KL divergence between unigram query and document language models. Similarly,

Table 1: Characteristics and statistics of the datasets in our experiments.

Dataset	Domain	# queries	# documents	avg doc length
MS MARCO DEV	Miscellaneous	6,980	8,841,823	56
TREC DL '19	Miscellaneous	43	8,841,823	56
TREC DL '20	Miscellaneous	54	8,841,823	56
SciFact	Scientific fact retrieval	300	5,183	214
FiQA	Financial answer retrieval	648	57,638	132
TREC COVID	Bio-medical retrieval for Covid-19	50	171,332	161
CQADupStack	Duplicate question retrieval	13,145	457,199	129

MRL uses negative KL divergence to compute relevance scores, however, there are several fundamental differences. First, Lafferty and Zhai [27] compute the distributions based on term occurrences in queries and documents through maximum likelihood estimation, while MRL learns *latent* distributions based on the contextual representations learned from queries and documents. Second, Lafferty and Zhai [27] use univariate distributions for queries and documents, while MRL uses high-dimensional multivariate distributions.

6 EXPERIMENTS

To evaluate the impact of multivariate representation learning, we first run experiments on standard passage retrieval collections from MS MARCO and TREC Deep Learning Tracks. We also study the parameter sensitivity of the model in this task. We further demonstrate the ability of multivariate representations to better model distribution shift when applied to zero-shot retrieval settings, i.e., retrieval on a target collection that is significantly different from the training set. Our experiments also shows that the norm of learned variance vectors is correlated with the retrieval performance of the model.

6.1 Datasets

In this section, we introduce our training set and evaluation sets whose characteristics and statistics are reported in Table 1.

Training Set. We train our ranking model on the MS MARCO passage retrieval training set. The MS MARCO collection [4] contains approximately 8.8M passages and its training set includes 503K unique queries. The MS MARCO training set was originally constructed for a machine reading comprehension tasks, thus it did not follow the standard IR annotation guidelines (e.g., pooling). The training set contains an average of 1.1 relevant passage per query, even though there exist several relevant documents that are left adjudged. This is one of the reasons that knowledge distillation help dense retrieval models learn more robust representations.

Passage Retrieval Evaluation Sets. We evaluate our models on three query sets for the passage retrieval task. They all use the MS MARCO passage collection. These evaluation query sets are: (1) *MS MARCO DEV*: the standard development set of MS MARCO passage retrieval task that consists of 6980 queries with incomplete relevance annotations (similar to the training set), (2) *TREC-DL'19*: passage retrieval query set used in the first iteration of TREC Deep Learning Track in 2019 [9] which includes 43 queries, and (3) *TREC-DL'20*: the passage retrieval query set of TREC Deep Learning Track 2020 [10] with 54 queries. Relevance annotation for TREC DL tracks was curated using standard pooling techniques. Therefore, we can consider them as datasets with complete relevance annotations.

Zero-Shot Passage Retrieval Evaluation Sets. To demonstrate the generalization of retrieval models to different domains, we perform a zero-shot passage retrieval experiment (i.e., the models are trained on the MS MARCO training set). To do so, we use four domains which diverse properties. (1) *SciFact* [51]: a dataset for scientific fact retrieval with 300 queries, (2) *FiQA* [29]: a passage retrieval dataset for natural language questions in the financial domain with 648 queries, (3) *TREC COVID* [50]: a task of retrieving abstracts of bio-medical articles in response to 50 queries related to the Covid-19 pandemic, and (4) *CQADupStack* [19]: the task of duplicated question retrieval on 12 diverse StackExchange websites with 13,145 test queries. To be consistent with the literature, we used the BEIR [45] version of all these collections.

6.2 Experimental Setup

We implemented and trained our models using TensorFlow. The network parameters were optimized with Adam [25] with linear scheduling with the warmup of 4000 steps. In our experiments, the learning rate was selected from $[1 \times 10^{-6}, 1 \times 10^{-5}]$ with a step size of 1×10^{-6}. The batch size was set to 512. The parameter β was selected from $[0.5, 1, 2.5, 5, 7.5, 10]$. To have a fair comparison with the baselines that often use 768 dimensions for representing queries and documents using BERT, we set the parameter k (i.e., the number of random variables in our multivariate normal distributions) to $\frac{768}{2} - 1 = 381$ (see Section 4.3 for more information). In our experiments, we use the DistilBERT [42] with the pre-trained checkpoint made available from TAS-B [18] as the initialization. As the re-ranking teacher model, we use a BERT cross-encoder, similar to that of Nogueira and Cho [33]. Hyper-parameter selection and early stopping was conducted based on the performance in terms of MRR on the MS MARCO validation set.

6.3 Evaluation Metrics

We use appropriate metrics for each evaluation set based on their properties. For MS MARCO Dev, we use MRR@10 which is the standard metric for this dataset, and we followed TREC Deep Learning Track's recommendation on using NDCG@10 [21] as the evaluation

Table 2: The passage retrieval results obtained by the proposed approach and the baselines. The highest value in each column is bold-faced. The superscript * denotes statistically significant improvements compared to all the baselines based on two-tailed paired t-test with Bonferroni correction at the 95% confidence level. "-" denotes the results that are not applicable or available.

Model	Encoder	#params	MS MARCO DEV		TREC-DL'19		TREC-DL'20	
			MRR@10	MAP	NDCG@10	MAP	NDCG@10	MAP
Single Vector Dense Retrieval Models								
ANCE [53]	BERT-Base	110M	0.330	0.336	0.648	0.371	0.646	0.408
ADORE [58]	BERT-Base	110M	0.347	0.352	0.683	0.419	0.666	0.442
RocketQA [37]	ERNIE-Base	110M	0.370	-	-	-	-	-
Contriever-FT [20]	BERT-Base	110M	-	-	0.621	-	0.632	-
TCT-ColBERT [28]	BERT-Base	110M	0.335	0.342	0.670	0.391	0.668	0.430
Margin-MSE [17]	DistilBERT	66M	0.325	0.331	0.699	0.405	0.645	0.416
TAS-B [18]	DistilBERT	66M	0.344	0.351	0.717	0.447	0.685	0.455
CLDRD [57]	DistilBERT	66M	0.382	0.386	0.725	0.453	0.687	0.465
MRL (ours)	DistilBERT	66M	**0.393***	**0.402***	**0.738**	**0.472***	0.701*	**0.479***
Some Sparse Retrieval Models (For Reference)								
BM25 [39]	-	-	0.187	0.196	0.497	0.290	0.487	0.288
DeepCT [12]	BERT-Base	110M	0.243	0.250	0.550	0.341	0.556	0.343
docT5query [32]	T5-Base	220M	0.272	0.281	0.642	0.403	0.619	0.407
Multi Vector Dense Retrieval Model (For Reference)								
ColBERTv2 [43]	DistilBERT	66M	0.384	0.389	0.733	0.464	**0.712**	0.473

metrics. To complement our result analysis, we also use mean average precision of the top 1000 retrieved documents (MAP), which is a common recall-oriented metric. For zero-shot evaluation, we follow BEIR's recommendation and use NDCG@10 to be consistent with the literature [45]. The two-tailed paired t-test with Bonferroni correction is used to identify statistically significant performance differences ($p\_value < 0.05$).

6.4 Experimental Results

Baselines. We also compare against the following state-of-the-art dense retrieval models with single vector representations:

- ANCE [53] and ADORE [58]: two effective dense retrieval models based on BERT-Base [13] that use the model itself to mine hard negative documents.
- RocketQA [37], Margin-MSE [17], and TAS-B [18]: effective dense retrieval models that use knowledge distillation from a BERT reranking model (a cross-encoder) in addition to various techniques for negative sampling.
- Contriever-FT [20]: a single vector dense retrieval model that is pre-trained for retrieval tasks and then fine-tuned on MS MARCO. This model has shown effective performance on out-of-distribution target domain datasets.
- TCT-ColBERT [28]: a single vector dense retrieval model that is trained through knowledge distillation where a multi vector dense retrieval model (i.e., ColBERT [24]) is used as the teacher model.
- CLDRD [57]: the state-of-the-art single vector dense retrieval model that uses knowledge distillation from a reranking teacher model through gradual increase of training data difficulty (curriculum learning).

Even though MRL is a single vector dense retrieval model, as a point of reference, we use a state-of-the-art dense retrieval model with multiple vectors (i.e., ColBERTv2 [43]). For demonstrating a fair comparison, all baselines are trained and tuned in the same way as the proposed approach.

We also compare our model against the following baselines that use inverted index for computing relevance scores (sometimes called sparse retrieval models):

- BM25 [39]: a simple yet strong term matching model for document retrieval that computes relevance scores based on term frequency in each document, document length, and inverse document frequency in the collection. We use the Galago search engine [36] to compute BM25 scores and tuned BM25 parameters using the training set.
- DeepCT [12]: an approach that uses BERT to compute a weight for each word in the vocabulary for each document and query. Then words with highest weights are then selected and added to the inverted index with their weights. This approach can be seen as a contextual bag-of-words query and document expansion approach.
- docT5query [32]: a sequence-to-sequence model based on T5 [38] that is trained on MS MARCO to generate queries from any relevance passage. The documents are then expanded using the generated queries.

The Passage Retrieval Results. The passage retrieval results are presented in Table 2. According to the table, all dense retrieval models perform substantially better than BM25 and DeepCT, demonstrating the effectiveness of such approaches for in-domain passage retrieval tasks. We observe that the approaches that use knowledge distillation (i.e., every dense retrieval model, except for ANCE,

Table 3: A comparison of storage requirement and query latency between single vector and multi vector dense retrieval models with DistilBERT encoders on MS MARCO collection with 8.8 million passages. We ran this experiment on a machine with 16 Core i7-4790 CPU @ 3.60GHz.

	storage requirement	query latency
Single vector DR	26GB	89 ms / query
Multi vector DR	192GB	438 ms / query

Table 4: Sensitivity of MRL's retrieval performance to different values of β.

	MS MARCO DEV		TREC-DL'19		TREC-DL'20	
	MRR@10	MAP	NDCG@10	MAP	NDCG@10	MAP
$\beta = 0.1$	0.385	0.384	0.723	0.448	0.693	0.466
$\beta = 0.25$	0.399	0.415	0.743	0.468	0.704	0.478
$\beta = 0.5$	0.403	0.408	0.742	0.481	0.703	0.486
$\beta = 1$	0.403	0.412	0.748	0.480	0.711	0.489
$\beta = 5$	0.405	0.421	0.749	0.484	0.716	0.489
$\beta = 10$	0.402	0.421	0.758	0.489	0.701	0.483

ADORE, and Contriever-FT) generally perform better than others. The recent CLDRD model shows the strongest retrieval results among all single vector dense retrieval models. The multi vector dense retrieval approach (ColBERTv2) outperforms all single vector dense retrieval baselines. Note that ColBERTv2 stores a vector for each token in the documents and thus it requires significantly larger storage for storing the ANN index and also suffers from substantially higher query latency (see Table 3 for more information). We show that MRL outperforms all baselines in terms of all the evaluation metrics used in the study. The improvements compared to all baselines are statistically significant, except for NDCG@10 in TREC-DL'19; the *p_value* (corrected using Bonferroni correction) for MRL versus CLDRD in this case was 0.07381. Note that this dataset only contains 43 queries and significance tests are impacted by sampled size. MRL performs significantly better than any other baseline in this case.

Parameter Sensitivity Analysis. To measure the sensitivity of MRL's performance to the value of β, we change β from 0.1 to 10 and report the results in Table 4. To get a sense of the impact of these values, please see Figure 2. The results show that the model is not sensitive to the value of β unless it is smaller than or equal to ≤ 0.25. Therefore, for a β value of around 1 or larger, the model shows a robust and strong performance.

The Zero-Shot Retrieval Results. All datasets used in Table 2 are based on the MS MARCO passage collection and their queries are similar to that of our training set. To evaluate the model's performance under distribution shift, we conduct a zero-shot retrieval experiment on four diverse datasets: SciFact, FiQA, TREC COVID, and CQADupStack (see Section 6.1). In this experiment, we do not re-train any model and the ones trained on MS MARCO training set and used in Table 2 are used for zero-shot evaluation on these datasets. The results are reported in Table 5. We observe that many neural retrieval models struggle with outperforming BM25

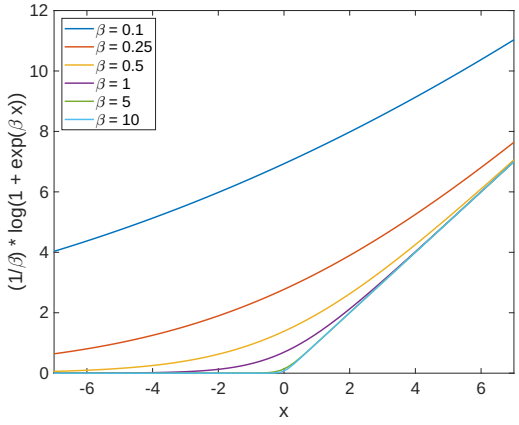


Figure 2: The softplus curve that is used to compute the variance vector for different values of β. Softplus is a monotonic and increasing function with a lower bound of zero. It's value for large x values can be approximated using the linear function $y = x$ for numeric stability.

Table 5: The zero-shot retrieval results obtained by the proposed approach and the baselines, in terms of NDCG@10. The highest value in each column is bold-faced. The superscript * denotes statistically significant improvements compared to all the baselines based on two-tailed paired t-test with Bonferroni correction at the 95% confidence level.

Model	SciFact	FiQA	TREC COVID	CQA DupStack
Single Vector DR Models				
ANCE [53]	0.507	0.295	0.654	0.296
ADORE [58]	0.514	0.255	0.590	0.273
RocketQA [37]	0.606	0.319	0.658	0.316
Contriever-FT [20]	0.677	0.329	0.596	0.321
TCT-ColBERT [28]	0.614	0.316	0.661	0.309
Margin-MSE [17]	0.608	0.298	0.673	0.297
TAS-B [18]	0.643	0.300	0.481	0.314
CLDRD [57]	0.637	0.348	0.571	0.327
MRL (ours)	**0.683***	**0.371***	0.668	0.341*
Some Sparse Retrieval Models (For Reference)				
BM25 [39]	0.665	0.236	0.656	0.299
DeepCT [12]	0.630	0.191	0.406	0.268
docT5query [32]	0.675	0.291	0.713	0.325
Multi Vector DR Models (For Reference)				
ColBERTv2 [43] (DistilBERT)	0.682	0.359	**0.696**	**0.357**

on SciFact and TREC COVID datasets. In general, the improvements observed compared to BM25 by the best performing models are not as large as the ones we observe in Table 2. This highlights the difficulty of handling domain shift by neural retrieval models. Generally speaking, the multi vector dense retrieval model (ColBERTv2)

Table 6: Pre-retrieval query performance prediction results in terms of Pearson's ρ and Kendall's τ correlations. The superscript † denotes that the obtained correlations by MRL $|\Sigma_Q|$ are significant.

QPP Model	TREC-DL'19		TREC-DL'20	
	P-ρ	K-τ	P-ρ	K-τ
Max VAR [5]	0.138	0.148	0.230	0.266
Max SCQ [60]	0.119	0.109	0.182	0.237
Avg IDF [5]	0.172	0.166	0.246	0.240
SCS [16]	0.160	0.174	0.231	0.275
Max PMI [15]	0.098	0.116	0.155	0.194
P_{clarity} [40]	0.167	0.174	0.191	0.217
Max DC [2]	**0.341**	**0.294**	0.234	0.244
MRL $\lvert\Sigma_Q\rvert$	$0.271^\dagger$	$0.259^\dagger$	$\mathbf{0.272}^\dagger$	$\mathbf{0.298}^\dagger$

shows a more robust performance in zero-shot settings. It outperforms all single vector dense retrieval models on TREC COVID and CQADupStack. MRL performs better on the other two datasets: SciFact and FiQA. Again, we highlight that MRL has substantially lower storage requirements compared to ColBERTv2 and it also has significantly faster query processing time. Refer to Table 3 for more information.

Exploring the Learned Variance Vectors. In our exploration towards understanding the representations learned by MRL, we realize that the norm of our covariance matrix for each query is correlated with the ranking performance of our retrieval model for that query. This observation motivated us to use the learned $|\Sigma_Q|$ for each query as a pre-retrieval query performance predictor (QPP). Some other well known pre-retrieval (i.e., based solely on the query and collection content, not any retrieved results) performance predictors include distribution of the query term IDF weights, the similarity between a query and the underlying collection; and the variability with which query terms occur in documents [60].

We compare our prediction against some of these commonly used unsupervised pre-retrieval QPP methods in Table 6. They include:

- Max VAR [5]: VAR uses the maximum variance of query term weight in the collection.
- Max SCQ [60]: It computes a TF-IDF formulation for each query term and returns the maximum value.
- Avg IDF [5]: This baseline uses an inverse document frequency formulation for each query term and return the average score.
- SCS [16]: It computes the KL divergence between the unigram query language model and the collection language model.
- Max PMI [15]: It uses the point-wise mutual information of query terms in the collection and returns the maximum value.
- P_{clarity} [40]: This baseline uses Gaussian mixture models in the embedding space as soft clustering and uses term similarity to compute the probability of each query term being generated by each cluster.
- Max DC [2]: This approach uses pre-trained embeddings to construct an ego network and computes Degree Centrality (DC) as the number of links incident upon the ego.

Following the QPP literature [5, 11, 15, 54], we use the following two evaluation metrics: Pearson's ρ correlation (a linear correlation metric) and Kendall's τ correlation (a rank-based correlation metric). We only report the results on the TREC DL datasets, since MS MARCO DEV only contains one relevant document per query and may not be suitable for performance prediction tasks. We observe that relative to existing pre-retrieval QPP approaches, MRL $|\Sigma_Q|$ has a high correlation with the actual retrieval performance. All of these correlations are significant ($p\_value < 0.05$). Note that MRL is not optimized for performance prediction and its goal is not QPP and these results just provide insights into what a model with multivariate representation may learn.

7 CONCLUSIONS AND FUTURE WORK

This paper introduced MRL– a novel representation learning paradigm for neural information retrieval. It uses multivariate normal distributions for representing queries and documents, where the mean and variance vectors for each input query or document are learned using large language models. We suggested a theoretically sound and empirically strong retrieval model based on multivariate Kullback-Leibler (KL) divergence between the learned representations. We showed that the proposed formulated can be approximated and used in existing approximate nearest neighbor search algorithms for efficient retrieval. Experiments on a wide range of datasets showed that MRL advances state-of-the-art in single vector dense retrieval and sometimes even outperforms the state-of-the-art multi vector dense retrieval model, while being more efficient and requiring orders of magnitude less storage. We showed that the norm of variance vectors learned for each query is correlated with the model's retrieval performance, and thus it can be used as a pre-retrieval query performance predictor.

Multivariate representation learning opens up many exciting directions for future exploration. Given the flexibility of multivariate normal distributions, the representations learned by the model can be easily extended. For instance, linear interpolation of multivariate normals is a multivariate normal distribution. Therefore, one can easily extend this formulation to many settings, such as (pseudo-) relevance feedback, context-aware retrieval, session search, personalized search, and conversational retrieval. Furthermore, such a representation learning approach can be extended to applications beyond standard IR problems. They can be used in representation learning for users and items in collaborative filtering models, graph embedding for link prediction and knowledge graph construction, and information extraction. Another promising research direction is enhancing retrieval-enhanced machine learning (REML) models [56] using multivariate representations. MRL enables REML models to be aware of the retrieval confidence and data distribution for making final predictions.

ACKNOWLEDGMENTS

This work was supported in part by the Google Visiting Scholar program and in part by the Center for Intelligent Information Retrieval. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Negar Arabzadeh, Bhaskar Mitra, and Ebrahim Bagheri. 2021. MS MARCO Chameleons: Challenging the MS MARCO Leaderboard with Extremely Obstinate Queries. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4426–4435.

[2] Negar Arabzadeh, Fattane Zarrinkalam, Jelena Jovanovic, Feras N. Al-Obeidat, and Ebrahim Bagheri. 2020. Neural embedding-based specificity metrics for pre-retrieval query performance prediction. *Inf. Process. Manag.* 57, 4 (2020), 102248.

[3] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research.

[4] Daniel Fernando Campos, Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, Li Deng, and Bhaskar Mitra. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *30th Conference on Neural Information Processing Systems, NIPS* (2016).

[5] David Carmel and Elad Yom-Tov. 2010. *Estimating the Query Difficulty for Information Retrieval* (1st ed.). Morgan and Claypool Publishers.

[6] Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and Diversity in Information Retrieval Evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08)*. Association for Computing Machinery, New York, NY, USA, 659–666. https://doi.org/10.1145/1390334.1390446

[7] Daniel Cohen, Bhaskar Mitra, Oleg Lesota, Navid Rekabsaz, and Carsten Eickhoff. 2021. Not All Relevance Scores Are Equal: Efficient Uncertainty and Calibration Modeling for Deep Retrieval Models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 654–664. https://doi.org/10.1145/3404835.3462951

[8] Kevyn Collins-Thompson and Jamie Callan. 2007. Estimation and use of uncertainty in pseudo-relevance feedback. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 303–310.

[9] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2019. Overview of the TREC 2019 Deep Learning Track. In *TREC*.

[10] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2020. Overview of the TREC 2020 Deep Learning Track. In *TREC*.

[11] Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. 2002. Predicting Query Performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02)*. Association for Computing Machinery, New York, NY, USA, 299–306. https://doi.org/10.1145/564376.564429

[12] Zhuyun Dai and Jamie Callan. 2020. Context-Aware Sentence/Passage Term Importance Estimation For First Stage Retrieval. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* abs/1910.10687 (2020).

[13] 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, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[14] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*. PMLR, 1050–1059.

[15] Claudia Hauff. 2010. Predicting the Effectiveness of Queries and Retrieval Systems. *SIGIR Forum* 44, 1 (aug 2010), 88. https://doi.org/10.1145/1842890.1842906

[16] Ben He and Iadh Ounis. 2004. Inferring Query Performance Using Pre-retrieval Predictors. In *String Processing and Information Retrieval*, Alberto Apostolico and Massimo Melucci (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 43–54.

[17] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation. *ArXiv* abs/2010.02666 (2020).

[18] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy J. Lin, and Allan Hanbury. 2021. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2021).

[19] Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. 2015. CQADupStack: A Benchmark Data Set for Community Question-Answering Research. In *Proceedings of the 20th Australasian Document Computing Symposium (ADCS '15)*. Association for Computing Machinery, New York, NY, USA, Article 3, 8 pages. https://doi.org/10.1145/2838931.2838934

[20] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Transactions on Machine Learning Research* (2022).

[21] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.* 20, 4 (oct 2002), 422–446. https://doi.org/10.1145/582415.582418

[22] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.

[23] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick 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 '20)*. Association for Computational Linguistics, Online, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[24] O. Khattab and Matei A. Zaharia. 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 (SIGIR '20)*.

[25] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference for Learning Representations (ICLR '15)*.

[26] Weize Kong, Swaraj Khadanga, Cheng Li, Shaleen Kumar Gupta, Mingyang Zhang, Wensong Xu, and Michael Bendersky. 2022. Multi-Aspect Dense Retrieval. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*. Association for Computing Machinery, New York, NY, USA, 3178–3186. https://doi.org/10.1145/3534678.3539137

[27] John Lafferty and Chengxiang Zhai. 2001. Document Language Models, Query Models, and Risk Minimization for Information Retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '01)*. Association for Computing Machinery, New York, NY, USA, 111–119. https://doi.org/10.1145/383952.383970

[28] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy J. Lin. 2021. Distilling Dense Representations for Ranking using Tightly-Coupled Teachers. *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)* (2021).

[29] Macedo Maia, Siegfried Handschuh, Andre Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. WWW'18 Open Challenge: Financial Opinion Mining and Question Answering. *WWW '18: Companion Proceedings of the The Web Conference 2018*, 1941–1942.

[30] Yu A. Malkov and D. A. Yashunin. 2020. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Trans. Pattern Anal. Mach. Intell.* 42, 4 (apr 2020), 824–836. https://doi.org/10.1109/TPAMI.2018.2889473

[31] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems* 30 (2017).

[32] Rodrigo Nogueira. 2019. From doc2query to docTTTTTquery.

[33] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *ArXiv* abs/1901.04085 (2019).

[34] Gustavo Penha and Claudia Hauff. 2021. On the calibration and uncertainty of neural learning to rank models for conversational search. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 160–170.

[35] Jay M. Ponte and W. Bruce Croft. 1998. A Language Modeling Approach to Information Retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*. Association for Computing Machinery, New York, NY, USA, 275–281. https://doi.org/10.1145/290941.291008

[36] The Lemur Project. [n.d.]. *Galago*. https://www.lemurproject.org/galago.php

[37] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 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, NAACL*.

[38] 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. *Journal of Machine Learning Research* 21, 140 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[39] Stephen Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference (TREC-3)*. Gaithersburg, MD: NIST, 109–126.

[40] Dwaipayan Roy, Debasis Ganguly, Mandar Mitra, and Gareth J.F. Jones. 2019. Estimating Gaussian mixture models in the local neighbourhood of embedded word vectors for query performance prediction. *Information Processing & Management* 56, 3 (2019), 1026–1045. https://doi.org/10.1016/j.ipm.2018.10.009

[41] G. Salton, A. Wong, and C. S. Yang. 1975. A Vector Space Model for Automatic Indexing. *Commun. ACM* 18, 11 (nov 1975), 613–620. https://doi.org/10.1145/361219.361220

[42] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv* abs/1910.01108 (2019).

[43] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 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. Association for Computational Linguistics, Seattle, United States, 3715–3734.

[44] Yue Shi, Xiaoxue Zhao, Jun Wang, Martha Larson, and Alan Hanjalic. 2012. Adaptive Diversification of Recommendation Results via Latent Factor Portfolio. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*. Association for Computing Machinery, New York, NY, USA, 175–184. https://doi.org/10.1145/2348283.2348310

[45] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

[46] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-Embeddings of Images and Language. In *Proceedings of the 2016 International Conference on Learning Representations (ICLR '16)*.

[47] Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic Embedding of Knowledge Graphs with Box Lattice Measures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 263–272. https://doi.org/10.18653/v1/P18-1025

[48] Luke Vilnis and Andrew McCallum. 2015. Word Representations via Gaussian Embedding. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR '15)*, Yoshua Bengio and Yann LeCun (Eds.).

[49] Ellen M. Voorhees. 2004. Overview of the TREC 2004 Robust Retrieval Track. In *The Thirteenth Text REtrieval Conference, TREC 2004*. 70–80.

[50] Ellen M. Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2020. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. *CoRR* (2020).

[51] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or Fiction: Verifying Scientific Claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 7534–7550. https://doi.org/10.18653/v1/2020.emnlp-main.609

[52] Jun Wang and Jianhan Zhu. 2009. Portfolio Theory of Information Retrieval. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '09)*. Association for Computing Machinery, New York, NY, USA, 115–122. https://doi.org/10.1145/1571941.1571963

[53] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *Proceedings of the 9th International Conference on Learning Representations (ICLR '21)*.

[54] Hamed Zamani, W. Bruce Croft, and J. Shane Culpepper. 2018. Neural Query Performance Prediction Using Weak Supervision from Multiple Signals. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 105–114. https://doi.org/10.1145/3209978.3210041

[55] Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 497–506. https://doi.org/10.1145/3269206.3271800

[56] Hamed Zamani, Fernando Diaz, Mostafa Dehghani, Donald Metzler, and Michael Bendersky. 2022. Retrieval-Enhanced Machine Learning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2875–2886. https://doi.org/10.1145/3477495.3531722

[57] Hansi Zeng, Hamed Zamani, and Vishwa Vinay. 2022. Curriculum Learning for Dense Retrieval Distillation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 1979–1983. https://doi.org/10.1145/3477495.3531791

[58] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing Dense Retrieval Model Training with Hard Negatives. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2021).

[59] Dell Zhang and Jinsong Lu. 2009. Batch-Mode Computational Advertising Based on Modern Portfolio Theory. In *Advances in Information Retrieval Theory*, Leif Azzopardi, Gabriella Kazai, Stephen Robertson, Stefan Rüger, Milad Shokouhi, Dawei Song, and Emine Yilmaz (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 380–383.

[60] Ying Zhao, Falk Scholer, and Yohannes Tsegay. 2008. Effective Pre-Retrieval Query Performance Prediction Using Similarity and Variability Evidence. In *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval (ECIR'08)*. Springer-Verlag, Berlin, Heidelberg, 52–64.

[61] Giulio Zhou and Jacob Devlin. 2021. Multi-vector attention models for deep re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 5452–5456.

[62] Jianhan Zhu, Jun Wang, Michael Taylor, and Ingemar J Cox. 2009. Risk-aware information retrieval. In *Advances in Information Retrieval: 31th European Conference on IR Research, ECIR 2009, Toulouse, France, April 6-9, 2009. Proceedings 31*. Springer, 17–28.

[63] Guido Zuccon, Leif Azzopardi, and C.J. "Keith" van Rijsbergen. 2010. Has Portfolio Theory Got Any Principles?. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*. Association for Computing Machinery, New York, NY, USA, 755–756. https://doi.org/10.1145/1835449.1835600