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.

KEYWORDS

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

CCS CONCEPTS

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

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. 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. 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.

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.
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 bootstrap 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 multivariiate 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:

\[ Q = (Q_1, Q_2, \cdots, Q_k)^T = \text{ENCODER}_Q(q) \]
\[ D = (D_1, D_2, \cdots, D_k)^T = \text{ENCODER}_D(d) \]

where \( \text{ENCODER}_Q \) and \( \text{ENCODER}_D \) respectively denote query and document encoders. Each \( Q_i \) and \( D_i \) is a random variable; thus \( Q \) and \( D \) are k-variate distributions representing the query and the document. The superscript \( T \) denotes the transpose of the vector.

In this paper, we assume that \( Q \) and \( D \) 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 \( \mu = (\mu_1, \mu_2, \cdots, \mu_k)^T \) and a \( k \times k \) covariance matrix \( \Sigma \) as follows: \( \mathcal{N}_k(\mu, \Sigma) \). We compute the representations as independent normal distributions, thus the covariance matrix is diagonal. Therefore, our representations are modeled as follows:

\[
\begin{pmatrix}
\mu_1 & 0 & \cdots & 0 \\
0 & \sigma_1^2 & & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \sigma_k^2
\end{pmatrix}
\]

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

\[
Q \sim \mathcal{N}_k(M_Q, \Sigma_Q), \quad \Sigma_Q = \text{ENCODER}_Q(q)
\]
\[
D \sim \mathcal{N}_k(M_D, \Sigma_D), \quad \Sigma_D = \text{ENCODER}_D(d)
\]

where \( M_Q = (\mu_1, \mu_2, \cdots, \mu_k)^T \), \( \Sigma_Q = (\sigma_1^2, \sigma_2^2, \cdots, \sigma_k^2)^T \times I_k \), \( M_D = (\mu_d_1, \mu_d_2, \cdots, \mu_d_k)^T \), and \( \Sigma_D = (\sigma_{d_1}^2, \sigma_{d_2}^2, \cdots, \sigma_{d_k}^2)^T \times I_k \). 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 \( x = (x_1, x_2, \cdots, x_k)^T \) being generated from the k-variate normal distribution in Equation (2) is equal to:

\[
p(x) = \frac{1}{(2\pi)^{k/2} \det(\Sigma)^{1/2}} \exp\left(-\frac{1}{2} (x - M)^T \Sigma^{-1} (x - M)\right)
\]

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: \( -\text{KL}_{D_k}(Q \parallel D) \). The KL divergence can be computed as follows:

\[
\text{KL}_{D_k}(Q \parallel D) = \mathbb{E}_Q \left[ \log \frac{Q}{D} \right] = \mathbb{E}_Q \left[ \log Q - \log D \right]
\]
\[
= \frac{1}{2} \mathbb{E}_Q \left[ -\log \det(\Sigma_Q) - (x - M_Q)^T \Sigma_Q^{-1} (x - M_Q) + \log \det(\Sigma_D) + (x - M_D)^T \Sigma_D^{-1} (x - M_D) \right]
\]
\[
= \frac{1}{2} \mathbb{E}_Q \left[ \log \det(\Sigma_Q) \right] - \frac{1}{2} \mathbb{E}_Q \left[ (x - M_D)^T \Sigma_D^{-1} (x - M_D) \right]
\]

(5)
Since \((x - M_Q)^\top \Sigma_Q^{-1} (x - M_Q)\) is a real scalar (i.e., \(\in \mathbb{R}\)), it is equivalent to \(\text{tr}((x - M_Q)^\top \Sigma_Q^{-1} (x - M_Q))\), where \(\text{tr}\) denotes the trace of the given matrix. Since \(\text{tr}(XY) = \text{tr}(YX)\) for any two matrices \(X \in \mathbb{R}^{m \times n}\) and \(Y \in \mathbb{R}^{n \times p}\), we have:

\[
\text{tr}((x - M_Q)^\top \Sigma_Q^{-1} (x - M_Q)) = \text{tr}((x - M_Q)(x - M_Q)^\top \Sigma_Q^{-1})
\]

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

\[
\text{KLD}_k(Q \parallel D) = \frac{1}{2} \log \frac{\det(\Sigma_D)}{\det(\Sigma_Q)} - \frac{1}{2} \text{tr} \left( \mathbb{E}_Q \left[ (x - M_Q)(x - M_Q)^\top \Sigma_D^{-1} \right] \right) + \frac{1}{2} \text{tr} \left( \mathbb{E}_Q \left[ (x - M_Q)^\top \Sigma_Q^{-1} (x - M_Q) \right] \right)
\]

(6)

Given the definition of the covariance matrix, we know that \(\Sigma_Q = \mathbb{E}_Q \left[ (x - M_Q)(x - M_Q)^\top \right]\). Therefore, we have:

\[
\text{tr} \left( \mathbb{E}_Q \left[ (x - M_Q)(x - M_Q)^\top \Sigma_D^{-1} \right] \right) = \text{tr} \left( \mathbb{E}_Q \left[ \Sigma_Q \Sigma_Q^{-1} \right] \right) = \text{tr}(I_k) = k
\]

(7)

In addition, since \(Q\) is a multivariate normal distribution, for any matrix \(A\) we have \(\mathbb{E}_Q[x^\top Ax] = \text{tr}(A \Sigma_Q) + M_Q^\top A M_Q\). This results in:

\[
\text{tr} \left( \mathbb{E}_Q \left[ (x - M_Q)^\top \Sigma_D^{-1} (x - M_Q) \right] \right) = \text{tr}(\Sigma_Q^2) + (M_Q - M_D)^\top \Sigma_Q^{-1} (M_Q - M_D)
\]

(8)

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

\[
\frac{1}{2} \log \frac{\det(\Sigma_D)}{\det(\Sigma_Q)} - k + \text{tr}(\Sigma_Q^2) + (M_Q - M_D)^\top \Sigma_Q^{-1} (M_Q - M_D)
\]

(9)

This equation can be further simplified. Based on our earlier assumption that the covariance matrices are diagonal, then \(\text{det}(\Sigma_D) = \prod_{i=1}^k \sigma_d^2\). In addition, since we are using KL divergence to rank documents, constant values (e.g., \(k\) or document independent values (e.g., \(\log \det(\Sigma_D)\)) 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:

\[
\text{score}(q,d) = -\text{KLD}_k(Q \parallel D) = \text{rank} \left[ -\frac{1}{2} \sum_{i=1}^k \log \frac{\sigma_d^2}{\sigma_d^2} + \frac{
abla_{i=1}^k \sigma_d^2}{\nabla_{i=1}^k \sigma_d^2} + \frac{(\mu_d - \mu_d)^2}{\sigma_d^2} \right]
\]

(10)

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 \(\text{ENCODER}_Q\) and \(\text{ENCODER}_D\) (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 adopt 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 \(\tilde{q}_{[CLS]} \in \mathbb{R}^{1 \times 768}\) and \(\tilde{q}_{[VAR]} \in \mathbb{R}^{1 \times 768}\) respectively denote the representations produced by BERT for the first two tokens [CLS] and [VAR]. We obtain the mean and variance vectors for query \(q\) and two different semantic projection layers on \(\tilde{q}_{[CLS]}\) and \(\tilde{q}_{[VAR]}\), as follows:

\[
M_Q = \tilde{q}_{[CLS]} W_M
\]

\[
\Sigma_Q = \frac{1}{\beta} \log(1 + \exp(\beta \tilde{q}_{[VAR]} W_S)) I_k
\]

(11)

where \(W_M \in \mathbb{R}^{768 \times k}\) and \(W_S \in \mathbb{R}^{768 \times k}\) 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 (\(\lim_{x \rightarrow -\infty} \frac{1}{\beta} \log(1 + \exp(\beta x)) = 0\)), 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., \(\lim_{x \rightarrow \infty} \frac{1}{\beta} \log(1 + \exp(\beta x)) = x\), ensuring numerical stability for large input values. To better demonstrate its properties, Figure 2 in our experiments plots softplus for various values of \(\beta\) – 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., \(\text{ENCODER}_Q\) and \(\text{ENCODER}_D\)).

#### 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:

$$
\sum_{d,d' \in D_q} \mathbb{I}(y_q^d(d) > y_q^d(d')) \frac{1}{\pi_q(d)} - \frac{1}{\pi_q(d')} \left| \log(1+e^{y_q^d(d')-y_q^d(d)}) \right|
$$

where $\pi_q(d)$ denotes the rank of document $d$ in the result list produced by the student dense retrieval model, and $y_q^d(d)$ and $y_q^d(d')$ 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., queries).
- $D_q$ includes $m_{BM25} \in \mathbb{R}$ documents from the top 100 documents retrieved by BM25.
- $D_q$ includes $m_{hard} \in \mathbb{R}$ 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

$$
- \frac{1}{\sum_{i=1}^{k} \frac{\sigma_{di}^2}{\pi_i \sigma_{di}^2} + \frac{k \mu_q^2}{\sum_{i=1}^{k} \sigma_{di}^2} + \frac{k \mu_d^2}{\sum_{i=1}^{k} \sigma_{di}^2} - \frac{k}{\sum_{i=1}^{k} \frac{2\mu_{di}\mu_{qi}}{\sigma_{di}^2}}}
$$

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 $\mathbb{R}^{1x(2k+2)}$ and $\vec{d}$ in $\mathbb{R}^{1x(2k+2)}$ 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 $M_d$ and $M_{d'}$ and identical covariance matrix $\Sigma_D = \Sigma_{D'}$. Therefore, given Equation (10) we have:

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

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 $Q \sim N_k(M_Q, \Sigma_Q)$ and the document distribution $D \sim N_k(M_D, \Sigma_D)$ are independent, the expected value of their product is:

$$
\mathbb{E}[Q \cdot D] = \mathbb{E}[Q] \cdot \mathbb{E}[D] = M_Q \cdot M_D
$$

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.

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

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 show 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 task, 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 $1 \times 10^{-6}$. The batch size was set to 512. The parameter $\beta$ 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 $768 \cdot \frac{1}{T} = 1 \approx 381$ (see Section 4.3 for more information). In our experiments, we used 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 metric for TREC DL tasks.
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,
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_corrected 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 sample 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 $\beta$, we change $\beta$ 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 $\beta$ unless it is smaller than or equal to 0.25. Therefore, for a $\beta$ 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 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 $\rho$ and Kendall’s $\tau$ correlations. The superscript $^\dagger$ denotes that the obtained correlations by MRL $|\Sigma_Q|$ are significant.

<table>
<thead>
<tr>
<th>QPP Model</th>
<th>TREC-DL’19 $P_\rho$</th>
<th>TREC-DL’20 $P_\rho$</th>
<th>TREC-DL’19 $K_\tau$</th>
<th>TREC-DL’20 $K_\tau$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max VAR [5]</td>
<td>0.138</td>
<td>0.148</td>
<td>0.230</td>
<td>0.266</td>
</tr>
<tr>
<td>Max SCQ [60]</td>
<td>0.119</td>
<td>0.109</td>
<td>0.182</td>
<td>0.237</td>
</tr>
<tr>
<td>Avg IDF [5]</td>
<td>0.172</td>
<td>0.166</td>
<td>0.246</td>
<td>0.240</td>
</tr>
<tr>
<td>SCS [16]</td>
<td>0.160</td>
<td>0.174</td>
<td>0.231</td>
<td>0.275</td>
</tr>
<tr>
<td>Max PMI [15]</td>
<td>0.098</td>
<td>0.116</td>
<td>0.155</td>
<td>0.194</td>
</tr>
<tr>
<td>$P_{\text{clarity}}$ [40]</td>
<td>0.167</td>
<td>0.174</td>
<td>0.191</td>
<td>0.217</td>
</tr>
<tr>
<td>Max DC [2]</td>
<td>0.341†</td>
<td>0.294†</td>
<td>0.234†</td>
<td>0.244†</td>
</tr>
<tr>
<td>MRL [$</td>
<td>\Sigma_Q</td>
<td>$]</td>
<td>0.271†</td>
<td>0.259†</td>
</tr>
</tbody>
</table>

Following the QPP literature [5, 11, 15, 54], we use the following two evaluation metrics: Pearson’s $\rho$ correlation (a linear correlation metric) and Kendall’s $\tau$ 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 formulation 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.


