Learning to Rank Under Uncertainty: A Robust Neural Approach

Maryam Khodabakhsh[1*] and Ebrahim Bagheri[2]

[1*]Faculty of Computer Engineering, Shahrood University of Technology, Shahrood, Iran.
[2]University of Toronto, Toronto, Canada.

*Corresponding author(s). E-mail(s): m_khodabakhsh@shahroodut.ac.ir;
Contributing authors: ebrahim.bagheri@utoronto.ca;

Abstract

Dense retrieval models provide representations in the form of embeddings in latent space and output a single deterministic score for a document based on the estimation of its relevance to the input query. While remarkable progress has been achieved in dense retrieval methods, they are limited by the fact that they consider queries and documents as deterministic points in latent space that encode the most likely features of the given query or document, and hence do not explicitly encode any degrees of noise, ambiguity or uncertainty. In this paper, we build on existing strong transformer-based dense retrievers by enabling them to capture uncertainty in latent space. In our proposed approach, embeddings in latent space are no longer a deterministic point, but rather a probabilistic distribution. With such probabilistic embeddings, the dense retrievers can be trained to achieve competitive performance on in-distribution queries and higher generalizability on out-of-distribution queries. Based on extensive experiments, we demonstrate that our proposed model consistently improves retrieval effectiveness in comparison to the state-of-the-art dense retrieval methods.

Keywords: Probabilistic Embeddings, Data Uncertainty, robustness, Learning to Rank

1 Introduction

Within the context of Information Retrieval, the *ad hoc retrieval* task is aimed at finding relevant documents or passages that can fulfill users' information needs expressed

as queries. Researchers have studied this task with the aim of devising methods that can efficiently and effectively map the query and document spaces onto each other in order to be able to seamlessly identify and retrieve relevant information. Most traditional methods have adopted a vector space model [1, 2], which represents queries and documents as sparse term vectors [3, 4]. These approaches are quite effective when retrieving documents that have close lexical resemblance to user queries. As such, they otherwise show less effective performance when so called *vocabulary mismatch* exists [5].

More recent neural-based retrieval models have moved beyond *sparse* vector space models into *dense* embedding representations for queries and documents that are efficient to use in terms of their memory usage and also effective as they can capture deeper semantic information beyond lexical similarities [6]. The idea behind such models is that dense embedding representations are learnt for queries and documents such that queries and documents that are relevant to each other are placed in close proximity to each other in the representation space, while those that are not relevant to each other are placed further apart. Furthermore, neural retrieval models experienced an increased effectiveness with the introduction of Transformer-based Pre-trained Language Models (PLM) that can be finetuned for the ad hoc retrieval task [7–11].

Modern neural information retrieval systems have shown remarkable performance improvements, particularly with the advent of pre-trained transformer-based models. Among them, Cross Encoder architectures [12, 13] have become the dominant choice for learning accurate matching functions between query-document pairs by jointly encoding the concatenated input into a single contextual representation as a *deterministic point embedding* in the latent space. Unlike existing sparse and dense-based retrieval models that compute independent embeddings for queries and documents, Cross Encoders allow rich token-level interactions, enabling finer-grained relevance estimation. This deterministic perspective is effective if the user's query is expressed clearly or if there are no ambiguities in the document corpus. However, in practice, it is difficult to estimate an accurate point embedding for *ambiguous* [14], *hard* [15–17], or *out-of-distribution* (out-of-domain) [18] queries, which can impose some degree of noise or uncertainty in the embedding space [19, 20]. Most Cross Encoders compute a deterministic score based on a fixed embedding (typically from the [CLS] token) of the joint input. This approach assumes a confident, point-estimate representation of relevance, which can lead to unreliable results under uncertainty, for example, with ambiguous queries, vague documents, or domain shifts. Ideally, a well-designed ranking system should be capable of capturing its own uncertainty when interpreting ambiguous or out-of-domain content and gracefully indicating when it is no longer effective or confident in its rankings. In order for the ranking system to be able to capture such uncertainty, it will require explicit mechanisms that are able to encode uncertainty when learning representations for queries and documents. Current standard approaches for learning representations for queries and documents that adopt a *deterministic point embedding* are not capable to capturing such uncertainty.

To address this issue, we propose a Variational Cross-Encoder Model, referred to as `VarCrossEncoder`, that adopts a *distributional estimation* instead of a deterministic point embedding in the latent space for each input query-document pair. The

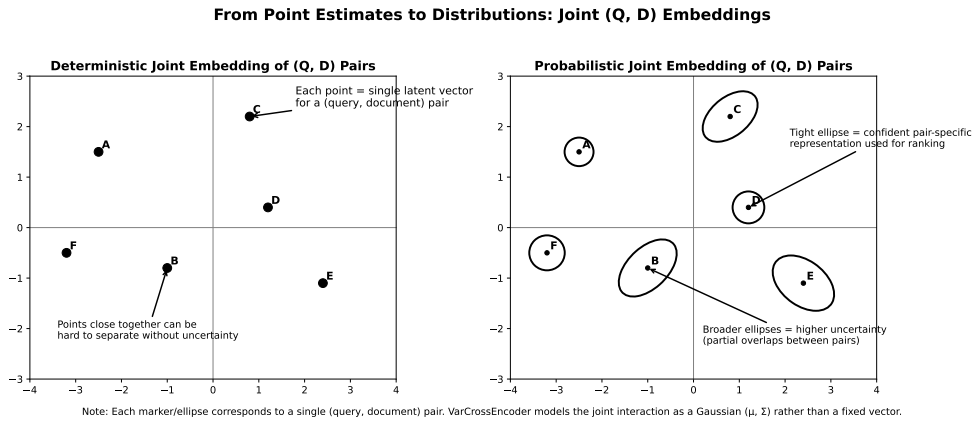

Fig. 1 Comparison of deterministic and probabilistic embeddings. In deterministic approaches, each query–document pair is mapped to a single point in the latent space, ignoring uncertainty. In contrast, probabilistic embeddings model each pair as a distribution, capturing variance across latent dimensions (illustrated here in a 2D space).

distributional estimation allows for capturing degrees of noise or uncertainty in the representation adopted for each point in latent space (see Figure 1). We encode the interaction between the query and document as a Multivariate Gaussian distribution, which assigns a probability to each point in latent space. The representations learnt by `VarCrossEncoder` follow Gaussian distributions with a diagonal covariance matrix. We demonstrate that such multivariate Gaussian representations for each query-document pair has the ability to effectively capture uncertainty in query and document representations, and hence exhibit retrieval effectiveness on in-distribution queries and increased generalizability and more effective performance on out-of-distribution queries.

While uncertainty has been studied in information retrieval, existing approaches leave several important gaps. Risk-aware retrieval methods [21] focus on variance at the score level, Bayesian sampling techniques primarily [22] address model-level uncertainty, and multivariate representation learning [23] treats queries and documents as independent probabilistic entities. These approaches do not explicitly capture data-level uncertainty in the joint query–document representation, where ambiguity in queries or noise in documents directly affects ranking effectiveness. This gap limits the robustness and generalizability of current neural rankers, particularly for ambiguous, noisy, or out-of-distribution queries.`VarCrossEncoder` addresses this gap by learning joint probabilistic embeddings of query–document pairs within a cross-encoder framework, thus modeling both the most likely semantic interactions and their inherent uncertainty.

The main contributions of this work can be categorized into three parts, conceptual, methodological, and empirical, as outlined below:

- **Conceptual contribution**: We propose the concept of uncertainty-aware probabilistic ranking, which encodes each query-document pair as a probabilistic

embedding rather than adopting an existing deterministic point embeddings. This allows our model to capture not only the most likely semantic representation of the query-document interaction but also the inherent uncertainty or ambiguity present in the data. By modeling the joint representation as a multivariate Gaussian distribution, our approach can express confidence over different dimensions of the latent space, facilitating more robust and adaptive ranking decisions, especially in the presence of noisy, ambiguous, or out-of-distribution queries.

- **Methodological contribution**: We propose a neural ranking architecture, `VarCrossEncoder`, that integrates probabilistic embeddings within a cross-encoder framework to effectively model the interaction between query and document representations. To optimize this architecture, we employ the Information Bottleneck principle, which encourages the model to preserve only the most relevant features of the joint representation while penalizing uncertainty through variational inference.
- **Empirical contribution**: We perform extensive experiments to show the impact of probabilistic embeddings on the ad hoc retrieval task. Our experimental results show the effectiveness of `VarCrossEncoder` compared to the state-of-the-art dense retrievers, which operate based on deterministic embeddings.

The rest of this paper is structured as follows: In Section 2, we cover relevant literature that has addressed uncertainty or noise in the area of information retrieval. Subsequently, in Section 3, we offer a clear problem statement and the desirable characteristics for a *robust* neural ranking model that intends to capture uncertainty. We discuss that such a model would need to show effective in-distribution retrieval, strong generalizability for out-of-distribution retrieval, and improved effectiveness on query subspaces that are difficult for neural rankers that operate based on deterministic embeddings. Based on this problem statement, we then introduce our proposed `VarCrossEncoder` approach. Section 4 then introduces our research questions, experimental setup, and the datasets used for evaluation purposes. This is followed by a detailed analysis of the findings from our experiments. In Sections 5, 6, and 7, we explore how the proposed model impacts hard queries, analyze its sensitivity to hyperparameters, and examine how alternative loss functions exhibit different performance. The paper is then concluded in Section 8.

2 Related Work

There has been a long interest in capturing various degrees of uncertainty in observations, data points, and decision boundaries in areas beyond information retrieval such as computer vision [24–26]. For instance, Ji et al. [25] leverage uncertainty-aware pre-training to improve robustness in multimodal tasks, enabling better handling of ambiguous or noisy inputs in vision-language understanding. Similarly, Upadhyay et al. [26] introduce ProbVLM, which integrates probabilistic adapters into frozen vision-language models, enhancing adaptability and reliability in downstream applications such as image captioning and visual question answering. Considering the popularity of cross-modal retrieval within the IR research community [27–29], Chun et al. [30] propose Probabilistic Embeddings for Cross-Modal Retrieval, introducing a framework where image-text pairs are represented as probability distributions instead of

fixed vectors. This approach captures uncertainty in feature representations, improving retrieval robustness, especially for ambiguous or noisy data. The model leverages probabilistic similarity metrics to enhance retrieval accuracy across different modalities. Their work demonstrates state-of-the-art performance in vision-language tasks by effectively modeling inherent uncertainty in multimodal embeddings. More recently, there has been increasing focus on the issue of deep uncertainty learning to improve the *robustness* and *interpretability* of discriminant Deep Neural Networks (DNNs) [31, 32]. Uncertainty can often be due to noise in the parameters of the deep neural networks (model uncertainty, also known as *epistemic uncertainty*) or the noise inherent in the training data (data uncertainty, also known as *aleatoric uncertainty*). Model uncertainty can be reduced by using ensemble models or additional training data [33–36]. In contrast, however, data uncertainty cannot be addressed with more training data and will require the explicit capturing of noise (uncertainty) in the model and its training process [37]. The uncertainty studied in our work in this paper can be categorized as *data uncertainty*.

In the context of information retrieval, Zhu et al. [21] may have been among the first researchers to discuss the concept of uncertainty, as an intrinsic part of document ranking, which has not generally been considered in other IR models. These researchers believed that variance or uncertainty could introduce a level of volatility in the retrieved results and proposed a risk-aware information retrieval model that allows for controlling such volatility. In their research, they approach the variance of a probabilistic language model [38] as a risk factor aimed at optimizing retrieval effectiveness. Generative models due to their probabilistic nature can also be regarded as a method for estimating data uncertainty in which relevance estimation is viewed as the probability of generating a query given document. As an example, dos Santos et al. [39] have employed large-scale sequence-to-sequence Transformer-based models to rank answers based on their *generation probability*, which implicitly captures uncertainty.

Recently, as neural ranking models have gained increased popularity in IR systems, it has become necessary to incorporate uncertainty estimation techniques into neural IR. For this reason, Penha et al. [40] captured uncertainty in conversational retrieval by first integrating dropout into a transformer architecture during inference and then modifying the ranking score through an uncertainty measure to improve the final re-ranking. The objective of this method was to reduce uncertainty in ranking as document scores from neural retrieval models often exhibit significant uncertainty, a factor often overlooked due to the widespread use of deterministic ranking models. Cohen et al. [22] also focused on this issue by modifying BERT-based rankers with a Bayesian approximation method of stochastic dropout sampling. This method captures predictive relevance distributions to measure uncertainty and subsequently incorporate uncertainty in the ranking process. Yang et al. [41] also adopted an uncertainty estimation approach to improve exploration in an online learning-to-rank model. Instead of using uncertainty-aware re-ranking, they used uncertainty estimates to identify candidate relevant documents. This helps reduce the *exploitation bias* commonly found in an online learning-to-rank setting.

Another important focus area in information retrieval is the *Query Performance Prediction (QPP)* task [15–17], which focuses on determining if the retrieved document

list meets the user's needs. In other words, it determines how difficult a query is and estimates its performance. One of the reasons for the poor performance of a query is query ambiguity [42] which means that the users may struggle to articulate their needs through the correct terminology, or there are no documents in the corpus that satisfy the users' needs. Query ambiguity could be seen as *noise in the query* or *uncertainty in the document collection*. On this basis, researchers have proposed effective QPP methods such as Normalized Query Commitment (NQC) [43] and Score Magnitude and Variance (SMV) [44] to estimate query performance by using the variance of the scores of the retrieved documents, capturing potential uncertainty. Another approach to address uncertainty is to intentionally generate uncertainty through *query perturbations* [15]. These approaches add noise to the initially ranked documents, hence generating perturbations, in order to assess the robustness of the ranked list in light of the injected noise and the introduced uncertainty.

There are also other methods for accounting for uncertainty. For instance, Chun et al. [45] approximate uncertainty using probabilistic distance for image-text matching. Wei et al. [46] introduced DVSSE, which captures uncertainty in visual retrieval through fine-grained sub-embedding tuning. Additionally, Li et al. [47] proposed a framework for cross-modal retrieval that first constructs a set of learnable prototypes for each modality to represent the entire semantic subspace. Then, Dempster-Shafer Theory and Subjective Logic Theory are utilized to develop an evidential theoretical framework by associating evidence with Dirichlet Distribution parameters.

In contrast to capturing uncertainty at the method level, some researchers have incorporated uncertainty at the *representation level* [23, 48, 49]. For instance, the Probabilistic Face Embedding (PFE) method [48] proposed to capture data uncertainty for each data sample in such a way that a Gaussian distribution is estimated, instead of a fixed point, in the latent space. In the context of IR, Zamani et al. [23] 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. They proposed a new representation learning framework, namely Multivariate Representation Learning (MRL), for dense retrieval where instead of learning a vector for each query and document, MRL learns a multivariate distribution and uses negative multivariate KL divergence to compute the similarity between distributions. Unlike MRL, EASE-DR [50] used VAE for correcting sentence representations with anisotropy by sampling them from the latent distribution of VAE. Other prior work [51–53] have also aimed to achieve semantically richer representations by modeling queries and documents using a combination of multiple vectors. While such representations were shown to lead to better retrieval effectiveness, they do come at significant computational and memory costs.

Our work differentiates itself from prior probabilistic ranking models in the following key ways:

- *Data-Level Uncertainty*: Unlike prior approaches such as [22, 40], which focus on capturing uncertainty at the model level, our method accounts for uncertainty at the data level to perform the ranking task. For example, [22] adopts a Bayesian perspective to estimate a model's uncertainty regarding its own document scoring, leveraging dropout as a form of variational inference.

- *Text-Only Retrieval Focus*: Among methods that model uncertainty at the representation level [30, 50, 54], our work is distinct in that it does not target cross-modal retrieval. Instead, we focus on matching text documents to text queries.
- *Joint Probabilistic Embeddings for Query-Document Pairs*: Unlike Multivariate Representation Learning (MRL) [23], which models uncertainty for text queries and documents separately at the representation level, our approach learns a joint probabilistic embedding for query-document pairs in a shared latent space. This design has been shown to be more efficient while achieving comparable or superior ranking performance across diverse retrieval benchmarks. We further incorporate an Information Bottleneck loss, which is absent in MRL. This framework enables us to balance informativeness and compression, guiding the model to retain only the most relevant aspects of the joint representation while penalizing uncertainty through variational inference. This leads to improved robustness on out-of-distribution queries and better handling of ambiguous cases, as shown in our experiments.

To clarify the degree of innovation in our work, it is important to situate it within existing research on uncertainty in information retrieval. Prior studies have addressed uncertainty through risk-aware retrieval [21], Bayesian dropout sampling applied to BERT-based rankers [22], and multivariate representation learning of queries and documents [23]. While these methods highlight the importance of uncertainty, they primarily capture model-level uncertainty or represent queries and documents as independent probabilistic entities. In contrast, our approach focuses on data uncertainty by introducing joint probabilistic embeddings of query–document pairs within a cross-encoder framework. This design enables us to capture both the most likely semantic interactions (means) and the inherent ambiguity (variances), thereby enhancing robustness in out-of-distribution settings and improving performance on difficult queries.

3 Proposed Approach

The objective of our work is to incorporate data uncertainty into query-document representations in order to maximize ranking performance in out-of-distribution scenarios. To this end, we propose a neural ranking model that relies on learning the probabilistic embeddings for each query-document pair to give a distributional estimation instead of a point estimate in the latent space.

In this section, we first offer a clear problem statement and then propose a neural ranking architecture and its associated loss function used for learning and optimizing the model with probabilistic embeddings.

3.1 Problem Statement

Given a user information need expressed as a query Q, the ranking task aims to retrieve a ranked list of documents from a corpus in order to maximize an evaluation metric of interest, e.g., nDCG or MAP. As a result of the ranking, every document D in the corpus will receive a score that can be used to measure the relevance of that

document to the query, $s(Q, D)$. Neural ranking models estimate query-document relevance scores by learning degrees of query-document association in the latent space based on lexical matching or learning embedding representations for semantic matching [55]. The main objective of our work is to establish a ***robust*** neural ranking model with the following Characteristics (**C**):

C1. Neural ranking models often rely on a significant number of training samples that consist of query-document pairs that help the model learn the association between queries and their relevant documents. However, the training data often comes from historical queries submitted to a search engine and relevant documents are assessed based on documents available in the corpus. In reality, users' information needs can shift and the distribution of content in the corpus can also change over time. As such, while neural models will be effective on query and document collections that have a similar distribution to the training data (referred to as *in-distribution*), they may suffer when adopted for *out-of-distribution* settings. The objective of our work is to allow the model to generalize effectively to *out-of-distribution* settings by effectively capturing uncertainty in query-document representations;
C2. While generalizing to out-of-distribution queries is important for robustness, it is also important for the ranking model to show strong stable performance on in-distribution queries. In other words, capturing uncertainty in query-document representations should not lead to a decreased performance on in-distribution queries;
C3. While most neural ranking models focus on improving the average effectiveness of retrieval results, recent research has shown that improving mean retrieval effectiveness does not necessarily lead to an improved stable performance across different query subsets [56, 57]. Accordingly, a robust neural ranking model is expected to achieve competitive average effectiveness by improving the performance of difficulty queries rather than only further improving the performance of queries that are already quite easy for other existing ranking methods.

Simply put, the above characteristics intend to ensure that our proposed model is generalizable to out-of-distributions settings (C1), shows stable and competitive performance on in-domain scenarios (C2), and shows improved performance on query subsets that are difficult for other neural ranking methods to satisfy (C3). In order to propose a neural ranking model that satisfies these three characteristics, we propose to explicitly capture the degree of ***uncertainty*** when learning and estimating the relevance score of a document for a given query. To this end, instead of treating the embeddings as a deterministic point in latent space, we propose probabilistic embeddings, which give a distributional estimation in the latent space for each input data. we hypothesize that the distribution of each embedding, after considering potential uncertainties, follows a multivariate Gaussian distribution where the mean of the distribution can be interpreted as the most likely latent feature values while the span of the distribution represents the uncertainty of these estimations. Our proposed probabilistic embeddings penalize uncertain features (dimensions) and pay more attention to more confident features.

Given the popularity of neural rankers that exploit pre-trained language models [58–61], and most specifically the superior performance of `Cross Encoders` [62]

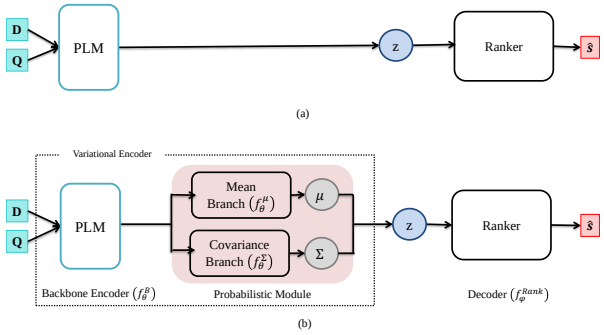

Fig. 2 Architecture of a typical `Cross Encoder` (a) against our Proposed `VarCrossEncoder` approach (b).

for document ranking, we propose `VarCrossEncoder`, which is based on the `Cross Encoder` ranking model and captures *data uncertainty* in the form of probability distributions (e.g., Gaussian) while learning query-document representations. Figure 2 depicts an overview of the proposed `VarCrossEncoder` approach. A typical `Cross Encoder` encodes the concatenation of the query and candidate document into a single deterministic representation, which is fed into a fully connected network (classifier) that gives a matching score. However, in contrast, our proposed `VarCrossEncoder` adopts a probabilistic embedding for the input data (the concatenation of query and document). In our approach, the input data is mapped onto a Gaussian distribution $\mathcal{N}(z; \mu, \Sigma)$ using a variational encoder that consists of a backbone feature extractor, f_θ^B, followed by two separate branches f_θ^μ and f_θ^Σ, each of which predicts μ and Σ. Embeddings sampled from this Gaussian are then consumed by a decoder (i.e., classifier or regressor in the context of learning to rank), f_φ^{Rank} to predict the relevance score.

3.2 Model Architecture

Given our objective is to integrate uncertainty information into the ranking task, `VarCrossEncoder` needs to have an extra component along with the ranking module that would be responsible for modeling uncertainty. To this end, the component of the variational encoder has been incorporated in the model to capture uncertainty. As shown in Figure 2, `VarCrossEncoder` consists of two components:

1. *The Variational Encoder*: To capture uncertainty, the variational encoder models the interaction between the queries and documents as a point in the latent probabilistic space and provides the probabilistic embeddings for a given input. A well-known practice is to model latent space as a Gaussian distribution where the mean can be regarded as the most likely embedding value while the diagonal covariance can be interpreted as the data uncertainty. Naturally, a larger variance

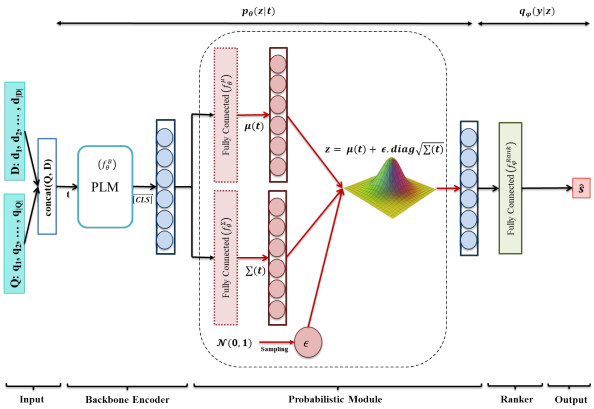

Fig. 3 The Architecture of our Proposed `VarCrossEncoder` approach.

means higher uncertainty. The motivation behind the probabilistic embeddings is that the learned variance acts like a predictor indicating how well an embedding represents the input samples as a place in the latent space based on their identity;

2. *Ranker*: As a common component in neural ranking models, the ranker decodes the probabilistic embeddings with the aim of estimating the relevance scores between a given query and the documents in the corpus.

In the following, we describe each component in detail.

3.2.1 The Variational Encoder

The variational encoder component in `VarCrossEncoder` aims to leverage uncertainty in the ranking task by encoding the input (both the query and the document) into a latent variable that models the distribution of diverse documents for a given query. The variational encoder models the distribution $p(z|t)$ with the parameters of distribution θ that are the weights of encoder layers in `VarCrossEncoder`. The input and output of the variational encoder are $t \in R^d$ and $z \in R^K$, respectively. The dimensionality d represents the size of the input embedding, typically matching the hidden size of the transformer model. In contrast, K denotes the dimension of the latent space where the compressed representation z is defined. As depicted in Figure 2, the variational encoder is formed as a backbone encoder followed by a probabilistic module.

The Backbone Encoder. The main objective of the backbone encoder is to provide a deterministic point representation for the input query and document in latent space. Given language models such as BERT are trained for tasks such as Next Sentence Prediction (NSP), they include additional meta-tokens that often capture the semantics of a sequence of terms including the [SEP] and [CLS] meta-tokens [63]. Earlier work has shown that the meta-tokens could be fine-tuned for performing downstream tasks such as document ranking [12, 13], among others. For instance, MacAvaney et al. [13] and Nogueira et al. [12] introduced *BERT-based ranking* where the query-document pairs are considered as two text sentences, which are separated

by the [SEP] meta-token and whose relevance relation can be captured through a leading [CLS] meta-token. Given `Cross Encoders` allow full token-level cross-interaction between query and document pairs and thus provide more accurate performance, we adopt a similar strategy to capture the interaction between query and document pairs.

We let Q be a query consisting of tokens $\{q_1, q_2, ..., q_{|Q|}\}$ and further let D be a document consisting of tokens $\{d_1, d_2, ..., d_{|D|}\}$. Adopted from [12, 13], we first concatenate the query and document pair using two meta-tokens, namely [SEP] and [CLS], and then feed them through transformer-based language models. The contextual representation of the leading [CLS] meta-token is used as the deterministic point embedding.

$$\overrightarrow{[\boldsymbol{CLS}]} = f_\theta^B(concat(Q, D)) \tag{1}$$

$$t = concat(Q, D) = [CLS] \quad q_1, q_2, ..., q_{|Q|} \quad [SEP] \quad d_1, d_2, ..., d_{|D|} \quad [SEP] \tag{2}$$

where $\overrightarrow{[CLS]}$ is the embeddings for the [CLS] token that is a deterministic point in the embedding space. f_θ^B is a transformer-based language model that encodes the query-document pair (Q, D) to the contextualized representation vector of $[CLS]$. The transformer-based language models not only extract the semantic information for the input but also model the attention between the query tokens and the document tokens.

The Probabilistic Module. Since it is difficult to give an accurate point embedding for noisy data, the probabilistic module models the input uncertainty in the embedding space by representing each embedding as a random variable: $z \sim p(z|t)$. As depicted in Figure 3, the output neurons of the probabilistic encoding are supposed to determine the parameters of the conditional distribution $p(z|t)$. Since the distribution $p(z|t)$ is a multivariate Gaussian distribution, we have two outputs for the variational encoder: one for the mean of this distribution $\mu(t)$ and the other for the covariance matrix of this distribution $\Sigma(t)$ which is usually diagonal. Any distribution with any number of parameters can be chosen for $p(z|t)$ but the multivariate Gaussian with diagonal covariance has already been widely used in the literature:

$$p(\mathbf{z}|t) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu(t)}, \Sigma(t)) = \mathcal{N}(\mathbf{z}|f_\theta^\mu(t), f_\theta^\Sigma(t)) \tag{3}$$

where the mean $\mu(t)$ and the diagonal covariance $\Sigma(t)$ are estimated through a shared backbone transformer-based encoder and *probabilistic branch* with separate output layers. Depending on whether the shared backbone encoder is followed by which of the mean or covariance prediction layers, or both of them, we propose three types of probabilistic modules: (1) VAR-based Module, (2) MEANVAR-based Module, and (3) MEAN-based Module.

1. VAR*-based Module (*VARM*).* The VAR-based Module uses a feature (mean of the Gaussian) from an existing deterministic representation model and the input uncertainty is modeled by learning only the variance. The hypothesis behind the VAR-based module is that the deterministic embedding [CLS], if properly optimized, represents the most likely features of the given input Q and D in the latent space. As such, given a pre-trained language model, we take $\mu(t) = [CLS]$ and optimize the uncertainty module, f_θ^Σ, to estimate $diag(\Sigma(t))$. We consider a network with fully connected layers with linear activating functions as the corresponding probabilistic module to encode the model's confidence along each feature dimension. The mean μ can be seen as the

most probable embedding value while the diagonal covariance Σ can be interpreted as the data uncertainty. The larger the variance is, the higher the uncertainty would be. This module is well-suited for applications where query-document pairs contain inherent noise or ambiguity, such as user-generated content, social media retrieval, or medical document search. By learning the variance while keeping the mean fixed, it helps mitigate the effects of uncertain or unreliable textual inputs in ranking models.

2) MeanVar*-based Module (*MeanVarM*).* The Var-based module is limited in that it adopts the deterministic representation as the embedded feature (mean) and learns only the uncertainty. As a result, it is unclear how uncertainty affects feature learning. In the MeanVar-based Module, as shown in figure 3, the deterministic representation is followed by two separate fully connected networks to predict both the parameters of the Gaussian distribution, $\mu(t)$ as well as $diag(\Sigma(t))$. Here we recall that $\mu(t)$ can be the identity feature of the query-document pair and $diag(\Sigma(t))$ would refer to the uncertainty of the predicted $\mu(t)$.

3) Mean*-based Module (*MeanM*).* In the Mean-based variation, for each query-document pair, the variance of the Gaussian is the embedding produced by the pre-trained language model, $diag(\Sigma(t)) = [CLS]$. An extra branch is appended to the pre-trained language model and trained to estimate the mean of $\mu(t)$. In MeanM, the [CLS] token is treated as variance rather than mean. The motivation is that while [CLS] may not always capture the central semantic content of a query–document pair, its variability across training samples can still serve as a signal of uncertainty. This allows the model to use [CLS] as an indicator of dispersion or noisiness in the representation rather than as the identity feature. This module is most effective in structured information retrieval scenarios where query ambiguity is minimal. By focusing only on the mean representation, it prioritizes efficient ranking and high retrieval precision in cases where uncertainty is less of a concern.

Sampling the Latent Variable. When the data point $[CLS]$ is fed as input to the probabilistic module, the parameters of the conditional distribution are obtained; hence, the distribution of latent space is determined corresponding to the data point t. Now, in the latent space, we sample the corresponding latent variable from the distribution of latent space:

$$\mathbf{z}_i \sim p(\mathbf{z}_i|t) \tag{4}$$

We adopt the re-parameterization trick [64] during training for easy backpropagation:

$$\mathbf{z}_i = \boldsymbol{\mu(t)} + \boldsymbol{diag}(\sqrt{\boldsymbol{\Sigma(t)}}) \cdot \epsilon^{(i)} \tag{5}$$

$$\epsilon^{(i)} \sim \mathcal{N}(0, 1) \tag{6}$$

Therefore, we first sample noise from $\mathcal{N}(0, 1)$ and then obtain z following Equation 5 instead of directly sampling from $\mathcal{N}(\mu(t), \Sigma(t))$. The latent variable z_i is fed as input to the classifier which is explained in the following.

While z is a probabilistic variable, the sampling during inference is performed from a well-defined distribution learned during training for each query-document pair. Although sampling introduces some stochasticity, the distribution is centered around the most likely embedding (the mean), and the variance typically reflects controlled uncertainty. In practice, we use the mean of the distribution (i.e., the expected value of z) to obtain stable and consistent ranking scores.

3.2.2 The Ranker

The last component of the `VarCrossEncoder` plays the role of a decoder in such a way that it consumes the probabilistic embeddings of the previous component to produce the final output of the model and is therefore highly dependent on the downstream task. A simple and straightforward formulation of the ranking task is to convert it into a regression or classification problem and then sort the documents based on the relevance scores or the probability that each document belongs to the relevant class, respectively. In our work, we consider the decoder as a regression function that is trained to map the probabilistic embeddings z for the given pair (Q, D) into a real-valued relevance score such that the most relevant documents to a given query are scored higher to maximize a rank-based metric:

$$\widehat{s} = f_{\varphi}^{Rank}(\mathbf{z}) \tag{7}$$

As shown in Figure 3, we add a fully connected layer with a linear activation function to establish f_{φ}^{Rank}:

$$\widehat{s} = f_{\varphi}^{Rank}(\mathbf{z}) = \mathbf{z} \cdot W \tag{8}$$

The input and output of the decoder are $z \in R^K$ and $s \in R$, respectively and $W \in R^K$ is a weight matrix. This component models the conditional distribution $q(y|z)$ with parameters of φ that are the weights of regression layers (W), in `VarCrossEncoder`.

3.3 Model Learning

To specifically optimize for the removal of irrelevant and redundant information from the input representations, we adopt the Information Bottleneck (IB) principle [65]. Let (Q, D, y) be a training sample where y is the binary relevance label for the query-document pair (Q, D). Given t as the concatenation of the query Q and document D, the main objective of the supervised IB is to preserve the information about the target class in the latent while filtering out irrelevant information from the input [66]. In other words, supervised IB aims to find a compressed representation z of the input t such that the mutual information between t and z is as low as possible (*compression loss*) and preserves information about the output y as high as possible (prediction loss), by minimizing:

$$\boldsymbol{\ell} = \beta \underbrace{I(t; \mathbf{z})}_{\texttt{Compression Loss}} - \underbrace{I(\mathbf{z}; y)}_{\texttt{Prediction Loss}} \tag{9}$$

where $\beta >= 0$ regulates the trade-off between compression and prediction, and $I(.,.)$ denotes mutual information. Specifically, $I(t; z)$ quantifies the amount of information retained by the latent representation z from t, which we aim to minimize to remove redundancy. Conversely, $I(z; y)$ represents the extent to which z captures useful information for predicting y, which we seek to maximize. In the Information Bottleneck (IB) framework, our goal is to learn a latent representation z that captures only the

information in t relevant for predicting the label y, while discarding noise and irrelevant details. This is achieved by minimizing $I(t;z)$, encouraging z to be a compressed, lossy version of t, and by maximizing $I(z;y)$, ensuring that z remains informative for predicting y. The term $I(t;z)$ acts as a compression loss, penalizing representations that retain unnecessary input information. Minimizing it helps prevent overfitting and enhances generalization—particularly important in the presence of uncertainty, which this work explicitly addresses. Alemi et al. [67] provide an efficient variational approximation for Equation 9 as follows:

$$\boldsymbol{\ell} = \beta \underbrace{\mathbb{E}_T[\quad KL[p_\theta(\mathbf{z}|t), r(\mathbf{z})]]}_{\texttt{Compression Loss}} + \underbrace{\mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|t)}[-log q_\varphi(y|\mathbf{z})]}_{\texttt{Prediction Loss}} \tag{10}$$

Here, $r(z)$ is an estimate of the prior probability $p(z)$ of z, and $p_\theta(z|x)$ is an estimate of the posterior probability of z. During training, the compressed sentence representation z is sampled from the distribution $p_\theta(z|x)$, meaning that a specific pattern of noise is added to the input of the output classifier $q_\varphi(y|z)$. Increasing this noise decreases the information conveyed by z. In this way, the variational encoder module can block the output classifier $q_\varphi(y|z)$ from learning to use specific information. At test time, the expected value of z is used for predicting labels with $q_\varphi(y|z)$.

We consider parametric Gaussian distributions for prior $r(z)$ and $p_\theta(z|t)$ to allow an analytic computation for the Kullback-Leibler divergence, namely $r(z) = \mathcal{N}(z|\mu_0, \Sigma_0)$ and $p_\theta(z|t) = \mathcal{N}(z|\mu(t), \Sigma(t))$, where μ and μ_0 are $K-$dimensional mean vectors and Σ and Σ_0 are diagonal covariance matrices. In other words, the first term in Equation 10 is a regularization term that explicitly constrains $\mathcal{N}(\mu(t), \Sigma(t))$ to be close to a multivariate normal distribution, $\mathcal{N}(0, 1)$:

$$\boldsymbol{\ell}_{Compression} = KL(p(\mathbf{z}|t)||r(\mathbf{z})) = \mathbb{E}_p[log(p(\mathbf{z}|t)) - log(r(\mathbf{z}))] \tag{11}$$

By considering the probability density function of multivariate normal distribution, we have:

$$\begin{aligned} KL(p(\mathbf{z}|t)||r(\mathbf{z})) &= \mathbb{E}_p[\frac{1}{2} log \frac{|\Sigma_0|}{|\Sigma(t)|} - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu(t)})^T \Sigma^{-1}(t)(\mathbf{x} - \boldsymbol{\mu(t)}) + \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_0})^T \Sigma_0^{-1}(\mathbf{x} - \boldsymbol{\mu_0})] \\ &= \frac{1}{2}\mathbb{E}_p[log \frac{|\Sigma_0|}{|\Sigma(t)|}] - \frac{1}{2}\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu(t)})^T \Sigma^{-1}(t)(\mathbf{x} - \boldsymbol{\mu(t)})] + \frac{1}{2}\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu_0})^T \Sigma_0^{-1}(\mathbf{x} - \boldsymbol{\mu_0})] \\ &= \frac{1}{2} log \frac{|\Sigma_0|}{|\Sigma(t)|} - \frac{1}{2}\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu(t)})^T \Sigma^{-1}(t)(\mathbf{x} - \boldsymbol{\mu(t)})] + \frac{1}{2}\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu_0})^T \Sigma_0^{-1}(\mathbf{x} - \boldsymbol{\mu_0})] \end{aligned} \tag{12}$$

Now, since $(x - \mu(t))^T \Sigma^{-1}(t)(x - \mu(t)) \in R$, we can write it as $tr\{(x - \mu(t))^T \Sigma^{-1}(t)(x - \mu(t))\}$, where $tr\{.\}$ is the trace operator. Using the trace trick, we can write it as $tr\{(x - \mu(t))(x - \mu(t))^T \Sigma^{-1}(t)\}$. The expectation and trace can be interchanged to obtain:

$$
\begin{aligned}
&= \frac{1}{2}\mathbb{E}_p[tr\{(\mathbf{x} - \boldsymbol{\mu(t)})(\mathbf{x} - \boldsymbol{\mu(t)})^T\Sigma^{-1}(t)\}] \\
&= \frac{1}{2}tr\{\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu(t)})(\mathbf{x} - \boldsymbol{\mu(t)})^T\Sigma^{-1}(t)]\} \\
&= \frac{1}{2}tr\{\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu(t)})(\mathbf{x} - \boldsymbol{\mu(t)})^T]\Sigma^{-1}(t)\}
\end{aligned} \tag{13}
$$

We know $\mathbb{E}_p[(x - \mu(t))(x - \mu(t))^T] = \Sigma(t)$. Hence, simplifying it to:

$$
\begin{aligned}
&= \frac{1}{2}tr\{\Sigma(t)\Sigma^{-1}(t)\} \\
&= \frac{1}{2}tr\{I_K\} \\
&= \frac{K}{2}
\end{aligned} \tag{14}
$$

We can now simplify the third term:

$$
\mathbb{E}_p[(\mathbf{x} - \boldsymbol{\mu_0})^T\Sigma_0^{-1}(\mathbf{x} - \boldsymbol{\mu_0})] = (\boldsymbol{\mu(t)} - \boldsymbol{\mu_0})^T\Sigma_0^{-1}(\boldsymbol{\mu(t)} - \boldsymbol{\mu_0}) + tr\{\Sigma_0^{-1}\Sigma(t)\} \tag{15}
$$

Combining all this, we obtain:

$$
KL(p(\mathbf{z}|t)||r(\mathbf{z})) = \frac{1}{2}\left[log\frac{|\Sigma_0|}{|\Sigma(t)|} - K + (\boldsymbol{\mu(t)} - \boldsymbol{\mu_0})^T\Sigma_0^{-1}(\boldsymbol{\mu(t)} - \boldsymbol{\mu_0}) + tr\{\Sigma_0^{-1}\Sigma(t)\}\right] \tag{16}
$$

When r is $\mathcal{N}(0, I)$, we will have:

$$
\ell_{Compression} = KL(p(\mathbf{z}|t)||r(\mathbf{z})) = \frac{1}{2}\left[\boldsymbol{\mu(t)}^T\boldsymbol{\mu(t)} + tr\{\Sigma(t)\} - K - log|\Sigma(t)|\right] \tag{17}
$$

The second term in Equation 10 measures how far the predicted labels of `VarCrossEncoder` are from true labels. We adapt cross-entropy loss incorporated with a Sigmoid function as follows:

$$
\ell_{Prediction} = -\left(ylog(q(y = 1|\mathbf{z})) + (1 - y)log(1 - q(y = 1|\mathbf{z}))\right) \tag{18}
$$

where $q(y = 1|z)$ demonstrates the predicted probability that D is relevant to Q. Given $\widehat{s}$ as the ranking score for pair (Q, D), we use the Sigmoid function to calculate the probability of $q(y = 1|z)$ by substituting the value of the Sigmoid function $\sigma(\widehat{s})$ in place of $q(y = 1|z)$ in the binary cross-entropy loss function:

$$
\begin{aligned}
\ell_{Prediction} &= -\left(ylog(\frac{1}{1+e^{-\widehat{s}}}) + (1-y)log(1-\frac{1}{1+e^{-\widehat{s}}})\right) \\
&= -\left(y(log(1) - log(1+e^{-\widehat{s}})) + (1-y)log(\frac{e^{-\widehat{s}}}{1+e^{-\widehat{s}}})\right) \\
&= -\left(-ylog(1+e^{-\widehat{s}}) - ylog(\frac{e^{-\widehat{s}}}{1+e^{-\widehat{s}}}) + log(\frac{e^{-\widehat{s}}}{1+e^{-\widehat{s}}})\right) \\
&= -\left(-ylog(1+e^{-\widehat{s}}) - yloge^{-\widehat{s}} + ylog(1+e^{-\widehat{s}}) + loge^{-\widehat{s}} - log(1+e^{-\widehat{s}})\right) \\
&= -\left(y\widehat{s} + -\widehat{s} - log(1+e^{-\widehat{s}})\right) \\
&= log(1+e^{-\widehat{s}}) + \widehat{s}(1-y)
\end{aligned}
\tag{19}
$$

Note that Equation 10 can be approximated using Monte Carlo sampling [68] with a sample size of N:

$$
\boldsymbol{\ell} = \frac{1}{N}\Sigma_{i=1}^{N}\left[\frac{\beta}{2}\left[\boldsymbol{\mu(t_i)}^T\boldsymbol{\mu(t_i)} + tr\{\Sigma(t_i)\} - K - log|\Sigma(t_i)|\right] - [log(1+e^{-\widehat{y}_i}) + \widehat{y}_i(1-y_i)]\right] \tag{20}
$$

$$
\boldsymbol{\ell} = \frac{1}{N}\Sigma_{i=1}^{N}\left[\frac{\beta}{2}\left[\boldsymbol{\mu(t_i)}^T\boldsymbol{\mu(t_i)} + tr\{\Sigma(t_i)\} - K - log|\Sigma(t_i)|\right] - [log(1+e^{-\widehat{y}_i}) + \widehat{y}_i(1-y_i)]\right] \tag{21}
$$

Algorithm 1 Training the `VarCrossEncoder`

Inputs: Dataset :$\{Q_i, D_i, y\}_{i=1}^{N}$
Outputs: Trained `VarCrossEncoder` $p_\theta(z|t), q_\varphi(y|z)$
for i = 1 to N **do**
  $t \leftarrow concat(Q_i, D_i)$
  $z \leftarrow f^B(t)$
  $\mu \leftarrow f^\mu(z)$
  $\Sigma \leftarrow f^\mu(z)$
  $\epsilon \leftarrow$ sample from $\mathcal{N}(0, I)$
  $z \leftarrow \mu + diag(\sqrt{\Sigma}).\epsilon$
  $\widehat{s} \leftarrow f^{Rank}(z)$
  $\ell_{Compression} \leftarrow \frac{1}{2}\left[\mu^T\mu + tr\{\Sigma\} - K - log|\Sigma|\right]$
  $\ell_{Prediction} \leftarrow log(1+e^{-\widehat{s}}) + \widehat{s}(1-y)$
  min $\ell \leftarrow \Sigma_{i=1}^{N}[\beta.\ell_{Compression} - \ell_{Prediction}]$ w.r.t. θ, φ by SGD
end for

Algorithm 1 summarizes the computational steps required to optimize Equation 10. To compute the compressed sentence representations $p_\theta(z|t)$, as shown in Algorithm 1, we first concatenate the input pair (Q, D) and the feed it $f^B(t)$ through a transformer based language model (Lines 4-5). It is then followed by fully connected linear layers, each with K hidden units to compute $\mu(t)$ and $\Sigma(t)$ (Lines 6-7). The re-parameterization trick is adapted to estimate the gradients, namely $z = \mu + diag(\sqrt{\Sigma}).\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$ (Lines 8-9). We also use another fully connected linear layer to approximate $q_\varphi(y|z)$ (Line 10). Finally, the loss function used for model training is computed based on Equation 10 (Lines 11-13).

3.4 Theoretical Properties

We now formalize several desirable theoretical properties of the `VarCrossEncoder` model. These properties help explain the observed improvements in empirical performance, particularly under ambiguous, noisy, or out-of-distribution conditions. Each property is motivated by a principle of robust representation learning and then proved based on the model's probabilistic structure.

The **first** property concerns the *stability of ranking scores under latent perturbations*. Intuitively, a desirable ranking system should produce stable and reliable relevance scores even when the inputs are subject to minor ambiguities or variations. In our model, each query-document pair is mapped to a multivariate Gaussian latent distribution $z \sim \mathcal{N}(\mu(t), \Sigma(t))$, and the final relevance score is computed by applying a linear decoder $s = z^\top W$. During inference, the expected value of this score is used: $\mathbb{E}[s] = \mu(t)^\top W$. Since the expected score depends solely on the mean of the latent distribution, it is deterministic and unaffected by the stochasticity introduced during training. Moreover, the score variance is given by $\mathrm{Var}[s] = W^\top \Sigma(t) W$, allowing the model to express its confidence in each prediction.

To formally justify this, recall that for a linear transformation of a Gaussian random vector $z \sim \mathcal{N}(\mu, \Sigma)$, the scalar random variable $s = z^\top W$ is also Gaussian, with mean and variance:

$$\mathbb{E}[s] = W^\top \boldsymbol{\mu} \tag{22}$$

$$\mathrm{Var}[s] = W^\top \Sigma W \tag{23}$$

Since the expected score is used at test time, the ranking output is stable with respect to the stochasticity of the latent space. The variance term $\mathrm{Var}[s]$ additionally enables a direct measure of uncertainty in the predicted score, making this formulation particularly valuable for retrieval under ambiguous or noisy conditions.

The **second** property addresses the *robustness of the latent space* induced by the variational regularization term in the model's training objective. `VarCrossEncoder` employs the Information Bottleneck principle [67] by minimizing the Kullback-Leibler (KL) divergence between the learned posterior $p(z|t) = \mathcal{N}(\mu(t), \Sigma(t))$ and a fixed isotropic prior $r(z) = \mathcal{N}(0, I)$. The KL divergence has a closed-form expression given by:

$$\mathcal{L}_{\mathrm{KL}} = \frac{1}{2}\left(\|\boldsymbol{\mu(t)}\|^2 + \mathrm{tr}(\Sigma(t)) - K - \log|\Sigma(t)|\right) \tag{24}$$

where K is the latent dimension. Each term in this expression serves to regularize the latent encoding. The norm $\|\mu(t)\|^2$ penalizes embeddings far from the origin, the trace $\mathrm{tr}(\Sigma(t))$ discourages overly diffuse representations, and the log-determinant $\log|\Sigma(t)|$ penalizes collapsed variances that may overfit to deterministic noise. Together, minimizing $\mathcal{L}_{\mathrm{KL}}$ prevents both overconfident and overly uncertain representations [69]. This leads to a form of norm control and scale normalization that is critical for generalization, especially when test queries deviate from the training distribution.

To see why this regularization enhances robustness, consider the role of each term in the loss landscape. A large norm $\|\mu(t)\|$ leads to higher KL penalty, forcing embeddings to remain near the prior's center unless strongly supported by data. Similarly, if $\Sigma(t)$ becomes too large (i.e., high uncertainty across all dimensions), the trace term increases the loss. On the other hand, if $\Sigma(t)$ collapses to zero (i.e., fully deterministic encoding), the log-determinant becomes negatively infinite. Thus, the KL loss creates a saddle-like geometry where moderate, well-supported embeddings are favored, thereby enhancing robustness across both in-distribution and out-of-distribution regimes.

The **third** property pertains to *feature selectivity*. In a high-dimensional latent space, not all dimensions are equally informative or reliable. The model should therefore learn to emphasize latent features that are confident and predictive, while downweighting dimensions that are uncertain or noisy. In `VarCrossEncoder`, this arises naturally from the structure of the latent distribution and the reparameterization trick used during training.

Let $\Sigma(t) = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_K^2)$ be the diagonal covariance matrix of the latent Gaussian for input t. Using the reparameterization trick, we write:

$$z = \boldsymbol{\mu(t)} + \boldsymbol{\Sigma^{1/2}(t)} \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{25}$$

Then the decoder score becomes:

$$s = \mathbf{z}^\top W = \boldsymbol{\mu(t)}^\top W + \epsilon^\top \boldsymbol{\Sigma^{1/2}(t)} W \tag{26}$$

The second term, $\epsilon^\top \Sigma^{1/2}(t) W$, represents the stochastic deviation in the score due to latent uncertainty. Each dimension k contributes a noise term $\epsilon_k \sigma_k W_k$. During training, if the noise induced by a particular dimension increases the prediction loss (e.g., because it leads to inconsistent outputs), the model can respond in two ways: either it reduces the uncertainty σ_k^2 via backpropagation through the variance head, or it reduces the decoder weight W_k. This leads to an emergent behavior where the model selectively relies on confident and predictive dimensions, while suppressing or ignoring those with high uncertainty [70].

The probabilistic structure and variational training of `VarCrossEncoder` ensure *three desirable theoretical properties*. **First**, relevance scores are stable in expectation and expose their uncertainty through analytically tractable variance. **Second**, KL-based regularization leads to bounded, normalized embeddings that generalize robustly. **Finally**, the model naturally exhibits feature selectivity, relying more heavily on informative dimensions while suppressing noisy ones. These properties are not only mathematically grounded but also closely aligned with the empirical strengths observed in our experiments.

4 Experiments

We have structured our experiments in such a way to assess whether three characteristics (C1-C3) that were introduced early in the paper have been satisfied in our proposed approach or not. To this end, we address the following three research questions:

RQ1. Would capturing uncertainty in the form of probabilistic embeddings as proposed in `VarCrossEncoder` maintain the same degree of effectiveness within *in-distribution* settings?
RQ2. Given the fact that deterministic embeddings often exhibit weaker performance in *out-of-distribution* settings, would the proposed `VarCrossEncoder` approach show any notable improvements for out-of-distribution queries?
RQ3. To what extent can the proposed `VarCrossEncoder` improve the performance of the state-of-the-art baseline or more specifically are any improvements over SOTA baselines due to the more effective performance of `VarCrossEncoder` on more difficult query subspaces?

In the following, we first introduced the datasets used for performing the experiments, and the metrics for evaluating the results. Then, we describe the implementation details for `VarCrossEncoder`. Finally, we present the results of the experiments and their analysis.

4.1 Dataset

We employ the well-known ranking dataset of MS MARCO[1] passage collection [71] for training `VarCrossEncoder`. MS MARCO is comprised 8.8M passages extracted from Web documents. The MS MARCO training set includes over half a million search queries sampled from the Bing search engine logs and also consists of over 532k relevant judgments where at least one relevant passage per query is marked by human assessors.

For evaluation purposes, we selected two groups of datasets. The first group consists of the *in-distribution queries* and the other contains the *out-of-distribution queries*.

4.1.1 In-Distribution Datasets

To evaluate the proposed `VarCrossEncoder` within an in-distribution setting, we used the MS MARCO Development set (Dev set, for short). It consists of 6,980 queries and their relevant passage pairs. We also additionally used the TREC Deep Learning Track 2019 [72], 2020 [73], and Deep Learning Hard (DL-Hard) [74] sets with 43, 54, and 50 queries, respectively in our evaluations. In contrast to the MS MARCO Dev set, which has around one relevant document per query, these three query sets provide multiple relevant documents per document judged on a 4-level relevance scale. Queries in Deep Learning tracks of 2021 [75] which have been annotated on MS MARCO V2 edition is another evaluation set with 53 topics. Summary statistics for the datasets used in our experiments are shown in Table 1.

[1] https://microsoft.github.io/msmarco

Table 1 Statistics of the datasets used in our experiments.

	Dataset	Task	Domain	Relevancy	#Queries	#Qrel
Train	MS MARCO Train set	Passage Retrieval	Misc.	Binary	502,939	532,761
In-Distribution	MS MARCO Dev set	Passage Retrieval	Misc.	Binary	6,980	7,437
	TREC DL 2019	Passage Retrieval	Misc.	4-level	47	9,260
	TREC DL 2020	Passage Retrieval	Misc.	4-level	54	11,386
	TREC DL 2021	Passage Retrieval	Misc.	4-level	53	10,8286
	DL-Hard	Passage Retrieval	Misc.	4-level	50	4256
Out-of-Distribution	FiQA-2018	Question Answering	Finance	Binary	648	1,706
	ELI5	Question Answering	Reddit	Binary	1,507	18,037
	COUGH	FAQ	Bio-Medical	Binary	1,201	39,760
	TREC-COVID	Document retrieval	Bio-Medical	3-Level	50	66,336
	Robust04	Document retrieval	News	3-Level	250	311,410

4.1.2 Out-of-Distribution Datasets

For out-of-distribution datasets, we adopted collections that were not related to the MS MARCO passage retrieval collection and hence could be considered separate and hence out-of-distribution. The following five datasets were chosen for this purpose:

FiQA-2018 [76]: This dataset contains question-answer pairs in the financial domain that were extracted from StackExchange posts discussing investment topics between 2009 and 2017. It includes 57,640 answer posts and 648 questions for testing. Each question is associated with an average of 2.6 posts with binary labels (relevant or irrelevant), indicating whether a post is the answer to its corresponding question or not.

ELI5 [77]: This is an English-language dataset of questions and answers gathered from the Reddit forum "Explain Like I'm Five"(ELI5). We use the KILT [78] version of the dataset which has 1,507 development examples. We employ all answers in the collection to build a corpus with 3,270 answers. Also, for a given query, we consider its corresponding answers as relevant and other answers in the corpus as irrelevant, resulting in a set with 18,037 binary relevance judgments.

COUGH [79]: This is an FAQ English dataset constructed by scraping data from 55 websites (e.g., CDC and WHO) containing 1,201 user queries and 7,117 FAQs about COVID-19. The relevance judgments set includes $\sim$ 32 human-annotated FAQ items per query with binary judgements.

TREC-COVID [80]: This is an ad-hoc search challenge for scientific articles related to COVID-19 based on the CORD-19 dataset. It contains 50 queries and 171K documents. The labels in TREC-COVID are 3-level (i.e. 0, 1, and 2) and there are 430.8 passages on average labeled as 1 or 2 in this version.

Robust04[2] is a dataset for news retrieval focusing on poorly performing topics. It has 250 queries and around half a million documents. The labels are 3-level and there are on average 69.9 passages labeled as relevant for each query.

The statistics of the out-of-distribution datasets are shown in Table 1.

[2] https://trec.nist.gov/data/robust/04.guidelines.html

4.2 Evaluation Metrics

To evaluate `VarCrossEncoder`, we employed the official metrics for each dataset, namely MRR@10 for the Dev set and NDCG@10 for TREC DL 2019, 2020, and DL-HARD. In addition to MRR@10 and NDCG@10, we evaluated our proposed approach for out-of-distribution datasets using precision and average precision at cut-off 10.

4.3 Implementation Details

We used distilbert-base-uncased [81], as the pre-trained version of BERT, with 66 million parameters pre-trained on the Toronto Book Corpus and English Wikipedia. We trained our model with a batch size of 32 for one epoch. A positive-to-negative ratio was adopted in the experiments in such a way that for each positive sample (label 1) four negative samples (label 0) were included. For the purpose of establishing negative samples, we benefited from the triples that are provided by Microsoft[3]. The parameters of the model are optimized via AdamW with the learning rate of $2e-5$.

In order to assess to what extent the probabilistic embedding can affect the model, we compared `VarCrossEncoder` against `Cross Encoder` with the same settings.

4.4 First-stage Retriever: BM25

Similar to MS MARCO and TREC DL sets, we used BM25 as the first-stage retriever to identify a list of the top-1000 documents per query on the out-of-distribution datasets. BM25 calculates a score for a query–document pair based on the statistics of the words that overlap between them. In order to evaluate our proposed `VarCrossEncoder` against `Cross Encoder`, we re-rank the top-1000 passages retrieved by BM25.

4.5 Results and Findings

In order to address three research questions, we performed numerous experiments. In the following, we will present the findings and provide an in-depth analysis.

4.5.1 Findings on Research Question 1 (RQ1)

As mentioned in Section 3, the probabilistic module of our proposed `VarCrossEncoder` approach injects uncertainty through a Gaussian multivariate distribution into the ranking by learning the probabilistic embeddings. In this first research question, we study the impact of this probabilistic module on the performance of the ranking task with particular attention to in-distribution datasets. To this end, we compared the performance of `VarCrossEncoder` against `Cross Encoder` for each three types of probabilistic modules separately. Table 2 shows the comparative results of the impact of data uncertainty over the overall performance of `VarCrossEncoder` on the in-distribution datasets in terms of their official metrics. The best performance for each dataset is shown in bold.

Since the evaluation datasets are the same domain as the training sets, the goal of these experiments is to investigate the robustness of `VarCrossEncoder`

[3] https://msmarco.z22.web.core.windows.net/msmarcoranking/qidpidtriples.train.full.2.tsv.gz

Table 2 The performance of `VarCrossEncoder` on in-distribution datasets.

	MS MARCO Dev set MRR@10	DL-Hard NDCG@10	TREC DL 2019 NDCG@10	TREC DL 2020 NDCG@10	TREC DL 2021 NDCG@10
`Cross Encoder`	0.356	0.386	0.698	0.678	0.602
`VarCrossEncoder`					
-VarM	**0.358**	**0.403**	0.699	**0.69**	**0.606**
-MeanVarM	0.355	0.391	**0.712**	0.682	0.587
-MeanM	0.345	0.363	0.687	0.686	0.571

in terms of the second characteristic (**C2**). As shown in Table 2, our proposed `VarCrossEncoder` always shows better performance compared to the `Cross Encoder` with deterministic embeddings. In particular, we point out that on the DL-Hard dataset, `VarCrossEncoder` with the Var-based module outperforms the baseline by 4.4% in terms of NDCG@10. Also, it is possible to see an improvement of 2.0% on NDCG@10 over TREC DL 2019 when the MeanVar-based module is applied in `VarCrossEncoder` compared to the baseline.

An important observation that can be made based on the results in Table 2 is that `VarCrossEncoder` with the Var-based module improves the baseline consistently over all in-distribution datasets while there is no consistent behavior for the two other types of probabilistic modules. For example, `VarCrossEncoder` with the Mean-based module outperforms the baseline over only TREC DL 2020. Similarly, `VarCrossEncoder` with the MeanVar-based module shows no performance improvement on either the MS MARCO Dev set or TREC DL 2021. These observations can be explained by the fact that the mean of the distribution estimates the most likely feature values while the *variance* shows the *uncertainty* in the feature values. So, to capture data uncertainty, it is important to learn variance that it minimizes the cross-entropy loss function.

To further investigate the effectiveness of `VarCrossEncoder` in ranking tasks, we analyzed its ability to distinguish relevant documents from irrelevant ones based on their predicted scores. For each query, we calculated the average predicted score of the relevant documents and compared it against the average score of the irrelevant documents. Since relevant documents are expected to be more semantically aligned with the query, we anticipated their scores to be consistently higher. Accordingly, we plotted the average scores of relevant and irrelevant documents across all in-distribution evaluation sets (excluding MS MARCO Dev) for the three probabilistic modules: VarM, MeanVarM, and MeanM. As illustrated in Figure 4, the green bars denote the average scores of relevant documents, while the red bars denote those of irrelevant documents. The results show that, across queries, the average scores of relevant documents are consistently higher than those of irrelevant ones in all evaluation sets and across all three modules, confirming that `VarCrossEncoder` effectively separates relevant from non-relevant items and thereby demonstrates strong regression ability in ranking. It is worth noting that, since no judgments for irrelevant documents are provided in MS MARCO Dev, the average score of irrelevant documents is set to zero, and no corresponding bars are plotted.

In summary and in response to RQ1, we find that moving from deterministic embeddings to our proposed probabilistic embeddings not only does not negatively impact the performance of the ranker on in-distribution data, but can also lead to increased performance if the Var-based module is adopted for the probabilistic

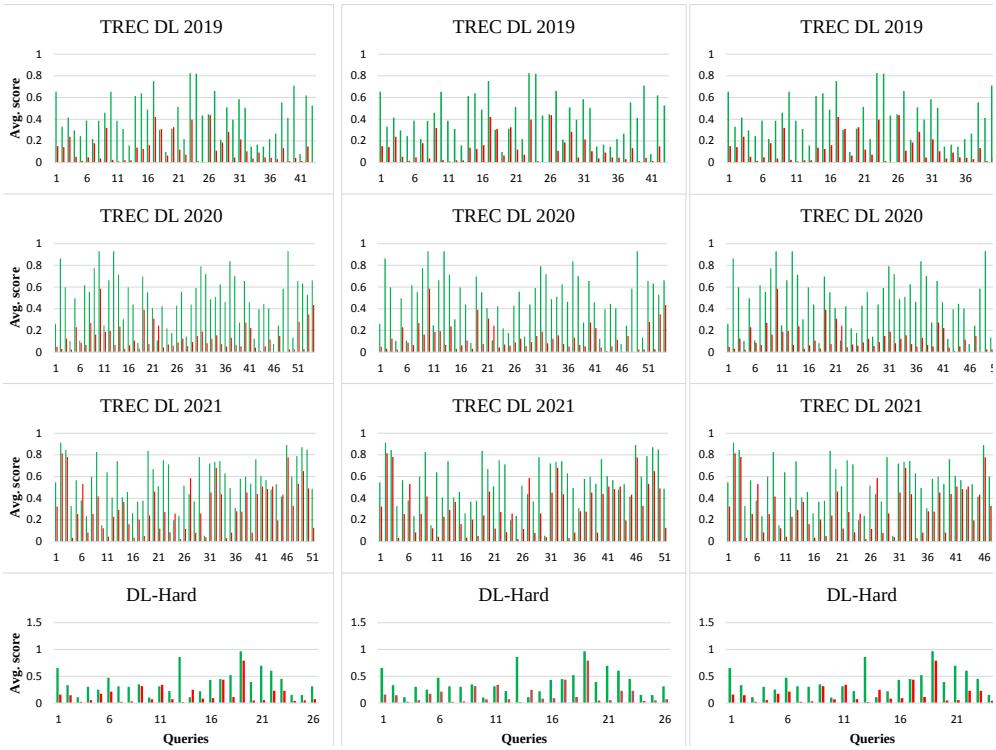

Fig. 4 Average predicted scores of relevant and irrelevant documents for each query using the proposed `VarCrossEncoder` with the VAR-based module (left), the MEANVAR-based module (middle), the MEAN-based module (right) across all in-domain evaluation sets. Green bars indicate relevant documents, and red bars indicate irrelevant documents. The differences observed between relevant and irrelevant document scores are statistically significant.

module. This is an important observation since one would expect that a deterministic embedding could learn in-distribution representations more accurately without requiring much flexibility afforded by capturing uncertainty. However, based on our observations, even within in-distribution datasets, capturing uncertainty can both maintain and increase the performance of the ranker.

4.5.2 Findings on Research Question 2 (RQ2)

Deep neural networks have shown impressive success in document ranking based on the fact that the training and testing domains follow an independent and identical distribution. The inevitable performance drop of deep neural ranking models can often be observed when tested in the out-of-distribution domains. Common neural ranking methods often neglect the representation discrepancy caused by out-of-distribution data during testing, as they only consider embeddings as deterministic values. We believe that the neural ranking model's ability is enhanced by incorporating the uncertainty of domain shifts during training. Specifically, we hypothesize that the embedding

Table 3 The performance of `VarCrossEncoder` and the baseline over the out-of-distributions datasets (left). Percentage improvement on the ranking task (right). * denotes statistically significant improvement over `Cross Encoder` at $p < 0.05$ based on a paired t-test across queries.

Domain	Ranking models	NDCG@10	P@10	MRR@10	AP@10	NDCG@10Δ%	P@10Δ%	MRR@10Δ%	AP@10Δ%
	`Cross Encoder`	0.277	0.075	0.35	0.212				
	`VarCrossEncoder`								
FiQA-2018	-VarM	**0.299***	**0.08***	**0.371***	**0.234***	**8.07**	**5.93**	**6.14**	**10.62**
	-MeanVarM	0.294*	0.078	0.366*	0.229*	6.12	3.07	4.68	7.84
	-MeanM	0.28	0.077	0.343	0.216	1.38	1.64	-1.87	2.16
	`Cross Encoder`	0.215	0.146	0.421	0.097				
	`VarCrossEncoder`								
ELI5	-VarM	0.229*	0.155*	**0.443***	0.106*	6.71	6.2	**5.26**	9
	-MeanVarM	**0.231***	**0.156***	0.442*	**0.107***	**7.46**	**7.11**	4.99	**10.39**
	-MeanM	0.219	0.148	0.423	0.101*	1.81	1.87	0.49	3.41
	`Cross Encoder`	0.177	0.104	0.311	0.102				
	`VarCrossEncoder`								
COUGH	-VarM	0.199*	0.117*	0.35*	0.116*	12.65	12.75	12.32	13.59
	-MeanVarM	**0.202***	**0.119***	**0.352***	**0.117***	**14.15**	**14.62**	**13.03**	**14.88**
	-MeanM	0.185*	0.109*	0.316	0.108*	4.56	4.35	1.51	5.83
	`Cross Encoder`	0.514	0.6	0.719	0.011				
	`VarCrossEncoder`								
TREC-COVID	-VarM	**0.539**	**0.614**	**0.799**	**0.012**	**4.74**	**2.33**	**11.08**	**6.87**
	-MeanVarM	0.534	0.592	0.771	0.011	3.91	-1.33	7.14	4.15
	-MeanM	0.516	0.584	0.765	0.011	0.29	-2.67	6.36	4.43
	`Cross Encoder`	0.42	0.402	0.655	0.096				
	`VarCrossEncoder`								
Robust04	-VarM	0.443*	0.425*	0.668	0.105*	5.62	5.79	2.09	8.76
	-MeanVarM	**0.455***	**0.432***	**0.69***	**0.106***	**8.34**	**7.39**	**5.44**	**9.76**
	-MeanM	0.45*	0.441*	0.665	0.105*	7.21	9.79	1.51	9.37

distribution follows a multivariate Gaussian distribution after accounting for potential uncertainties. Therefore, each embedding is now a probabilistic point with various distribution possibilities, rather than a deterministic value. The second research question (RQ2) aims to explore whether the neural ranking models can be trained with uncertain representations to mitigate the effects of domain variations and improve resilience against out-of-distribution data. To address this research question, we adopt two complementary perspectives. First, we evaluate `VarCrossEncoder`, trained on the MS MARCO dataset, across multiple datasets that differ in domain from MS MARCO in order to assess its effectiveness in out-of-distribution scenarios. Second, we examine the robustness of `VarCrossEncoder` under conditions of noise injection in queries drawn from these out-of-domain datasets.

Generalization of `VarCrossEncoder` Beyond the Training Domain. To answer RQ2 from the perspective of generalization, we evaluate our proposed `VarCrossEncoder` approach, trained on the MS MARCO set, over five datasets that are different in domain compared to the MS MARCO set. Table 3 shows the performance of `VarCrossEncoder` and the base `Cross Encoder` in terms of NDCG, Precision, Mean Reciprocal Rank, and Average Precision at cut-off 10 when trained on the MS MARCO train set and tested over out-of-distribution sets of FiQA-2018 [76], EII5 [77], COUGH [79], TREC-COVID [80], and Robust04. Also, Table 3 shows the improvement percentage in the ranking task in terms of four evaluation metrics. The best performance for each dataset is shown in bold.

This second research question explores the robustness of `VarCrossEncoder` in terms of the first characteristic **C1**. As observed in Table 3, the proposed `VarCrossEncoder` approach has effectively improved the performance of the baseline over the out-of-distribution datasets. We make several observations based on the results in Table 3. In terms of performance, `VarCrossEncoder` with the MEAN-based

module has the least improvements over the datasets in terms of four evaluation metrics, while the `VarCrossEncoder` with both VAR and MEANVAR probabilistic modules exhibit higher improvements. This observation is consistent with observations made in RQ1 such that the ranking performance is more considerable when the uncertainty is learned by the parameter of the variance (Var and MeanVar) instead of fixing it (Mean). The most significant degree of improvement is observed in the COUGH dataset (14.14% on NDCG@10) when both parameters of the Gaussian distribution (mean and variance) are learned simultaneously. The `VarCrossEncoder` approach with VARVar-based module shows the most impact at 8.07% and 4.74% on two datasets, namely FiQA-2018 and TREC-COVID, respectively in terms of NDCG@10 while for both ELI5 and Robust04, the highest improvement is observed when the MEANVAR module is adopted.

We observe that the improvements of `VarCrossEncoder` on out-of-distribution datasets vary by domain. Larger gains are seen in datasets such as COUGH and ELI5, where queries are short, ambiguous, or phrased in diverse ways, resulting in high levels of data uncertainty. In these cases, probabilistic embeddings capture the variability in query–document interactions more effectively than deterministic models. Notably, the MeanVar module, which models both the mean and variance of interactions, outperforms modules that focus on only one aspect. By contrast, in more structured or technical datasets such as TREC-COVID, where terminology is precise and query intent is clearer, improvements are smaller but still consistent. Here, the Var module, which focuses solely on modeling variance, performs better than modules that model both mean and variance simultaneously. Overall, `VarCrossEncoder` provides consistent improvements across all evaluated datasets, with particularly strong benefits in domains characterized by noisy or underspecified queries. This suggests that uncertainty modeling is especially effective in such high-uncertainty domains, while also maintaining robustness in datasets with more structured or specialized vocabulary.

For the out-of-distribution evaluation sets, we conducted the same statistical analysis by computing, for each query, the average predicted score of relevant documents and comparing it with the average score of irrelevant documents. Figure 5 shows these averages for three datasets of COUGH, TREC-COVID, and Robust04 across the three probabilistic modules: VARM, MEANVARM, and MEANM. The results demonstrate that, across queries, relevant documents consistently receive higher scores than irrelevant ones for all datasets and across all three modules, confirming that `VarCrossEncoder` effectively separates relevant from irrelevant items and exhibits strong regression ability in ranking. It is worth noting that this analysis could not be performed for the ELI5 and FiQA-2018 datasets, due to the same limitation encountered with MS MARCO Dev, namely, the absence of explicit judgments for irrelevant documents.

In summary and in response to RQ2, we find that capturing uncertainty in the neural ranker will lead to notable improvements on out-of-distribution datasets. This means that the model is able to generalize more effectively to domains that it has not necessarily seen in the past during the testing phase when uncertainty has been captured explicitly in the training phase.

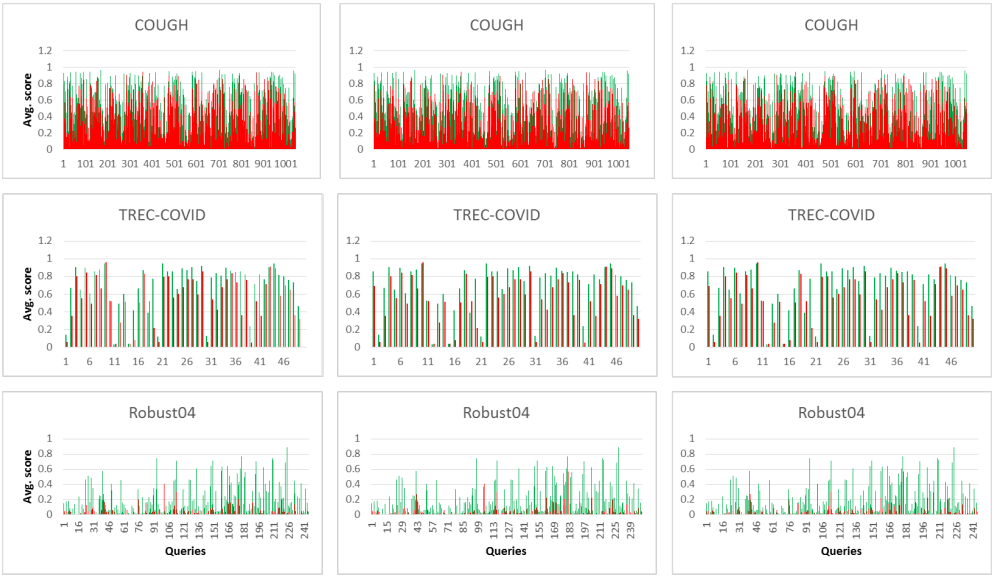

Fig. 5 Average predicted scores of relevant and irrelevant documents for each query using the proposed `VarCrossEncoder` with (a) the VAR-based module, (b) the MEANVAR-based module, (c) the MEAN-based module across all out-of-distribution evaluation sets. Green bars indicate relevant documents, and red bars indicate irrelevant documents. The differences observed between relevant and irrelevant document scores are statistically significant.

Robustness of `VarCrossEncoder` Under Noisy Query Conditions. Neural ranking model of `VarCrossEncoder` is designed to capture uncertainty when estimating document relevance for a given query. Capturing such uncertainty enhances robustness, which is particularly critical in real-world information retrieval systems where users' information needs evolve over time and corpus distributions shift, and noise frequently arises in queries and documents.

In practice, the effectiveness of ranking models can be significantly compromised by noise, which arises naturally in user queries and documents. For example, spelling errors, synonym mismatches, and incomplete queries frequently occur in real search environments. Previous work has shown that neural ranking models are especially sensitive to such perturbations [82]. To develop models that are more stabile in noisy conditions, it is essential to systematically evaluate the effects of different noise types on ranking performance.

To this end, we examine the robustness of `VarCrossEncoder` in comparison with Cross Encoder under six forms of word-level noise. These include: (1) spelling errors (SP), where a query word is substituted with a common misspelling from a predefined dictionary; (2) wordEmb-insert (WEI), which inserts additional words selected via similarity in word embeddings [83]; (3) wordEmb-substitute (WES), which replaces a query word with one of its closest neighbors in the word embedding space; (4) synonym

Table 4 Performance of Cross Encoder and `VarCrossEncoder` under different noise types across out-of-domain sets.

	Cross Encoder						`VarCrossEncoder`					
	RD	RS	SE	SR	WEI	WES	RD	RS	SE	SR	WEI	WES
FiQA-2018	0.165	0.27	0.138	0.168	0.146	0.73	0.182	0.283	0.173	0.19	0.179	0.92
ELI5	0.156	0.203	0.123	0.143	0.101	0.066	0.168	0.219	0.144	0.162	0.123	0.088
COUGH	0.125	0.172	0.089	0.109	0.081	0.051	0.156	0.198	0.122	0.138	0.111	0.078
TREC-COVID	0.387	0.509	0.446	0.429	0.419	0.303	0.434	0.52	0.444	0.481	0.456	0.36
Robust04	0.194	0.401	0.262	0.321	0.351	0.206	0.222	0.443	0.297	0.359	0.4	0.238

replacement (SR), where a word is substituted using WordNet synonyms; (5) random-swap (RS), which exchanges the positions of two randomly chosen words; and (6) random-delete (RD), which removes a randomly selected word from the query. Noise is quantified at the word level, and in all experiments, the level of noise was set to one, meaning a single word was modified per query.

The chosen noise types were implemented using the NLPaug library [84], a widely used open-source augmentation toolkit that offers flexible and reproducible methods for introducing perturbations in textual data. This ensured consistency across experiments and seamless integration into our evaluation pipeline.

Our evaluation involved injecting one of the six noise types into queries and ranking documents using both models of `VarCrossEncoder` and the baseline of Cross Encoder. Models' performance was assessed using NDCG@10 across all out-of-domain datasets, as reported in Table 4. It is important to note that the results for `VarCrossEncoder` were obtained using the MeanVarM module.

Two main conclusions can be drawn from the findings. First, the performance of both models degrades consistently across all datasets when noise is introduced, compared to their noise-free counterparts (based on Table 3). This observation confirms the sensitivity of neural ranking models to query perturbations. Second, `VarCrossEncoder` demonstrates consistently higher performance than the Cross Encoder across nearly all evaluation sets and noise types. The only exception occurs on the TREC-COVID dataset under spelling noise, where the Cross Encoder marginally outperforms `VarCrossEncoder`. Overall, the results indicate that `VarCrossEncoder` is more robust to noisy input, validating the effectiveness of incorporating uncertainty modeling into the ranking process.

4.5.3 Findings on Research Question 3 (RQ3)

In this research question, we are interested in comparing the performance of `VarCrossEncoder` against the state-of-the-art baselines to understand how our proposed approach for capturing uncertainty compares against existing work in the literature. For this purpose, we consider two groups of baselines: (i) ranking models based on probabilistic embeddings, and (ii) neural ranking models with multiple vector representations.

Ranking Models based on Probabilistic Embeddings: the most relevant state-of-the-art baseline to our work for capturing uncertainty in ranking models is the Multivariate Representation Learning (MRL) model [23] that learns separate probabilistic representations for queries and documents, using negative multivariate KL

Table 5 Comparative performance of the baselines with `VarCrossEncoder`.

	MRL				EASE-DR				PCME				VarCrossEncoder			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.081	0.029	0.092	0.051	0.012	0.005	0.016	0.007	0.003	0.002	0.003	0.001	0.294	0.078	0.366	0.229
ELI5	0.143	0.108	0.276	0.056	0.024	0.02	0.056	0.007	0.006	0.004	0.017	0.002	0.231	0.156	0.442	0.107
COUGH	0.096	0.065	0.178	0.048	0.021	0.015	0.043	0.009	0.005	0.004	0.007	0.186	0.202	0.119	0.352	0.117
TREC-COVID	0.448	0.538	0.617	0.011	0.208	0.232	0.438	0.003	0.165	0.186	0.368	0.002	0.534	0.592	0.771	0.0114
Robust04	0.254	0.286	0.371	0.035	0.079	0.078	0.175	0.013	0.048	0.047	0.116	0.008	0.455	0.432	0.69	0.106

Table 6 GPU memory usage and inference time comparison between VarCrossEncoder and baseline models (evaluated on an RTX 6000 Ada GPU).

	MRL	EASE-DR	PCMR	VarCrossEncoder
Memory gpu (MiB)	41617	57464	22355	6707
Inference time (Second)	0.101	5.94	0.006	0.767

divergence for similarity estimation. In contrast, our VarCrossEncoder employs a cross-encoding framework that models uncertainty at the interaction level between queries and documents, allowing for directly capturing uncertainty in query-document relevance estimation, rather than at the individual embedding level. EASE-DR [85] is another baseline that samples the latent space distribution of a variational autoencoder to generate isotropic sentence embeddings. It then applies supervised contrastive learning to enhance the uniformity of these embeddings in the representation space. As an additioanl baseline, we employed PCME [30], which models samples from paired data as probabilistic distributions within a shared embedding space. However, instead of image-caption pairs, we used query-document pairs. Table 5 shows the performance of `VarCrossEncoder` and the baselines over out-of-distribution datasets. For ease of comparison, we have compared our best-performing variation MeanVarM against the baselines; however, comparison against other variations of our work can easily be done according to performance values reported in Table 3. As shown in Table 5, for all datasets, the baselines show a consistently weaker performance in terms of all evaluation metrics compared to our method. In other words, our proposed `VarCrossEncoder` shows consistently better performance over the baselines on out-of-distribution datasets pointing to the fact that it has been able to generalize more effectively by explicitly capturing uncertainty in its training process. It is worth noting that while PCME also employs probabilistic embeddings, it was originally designed for cross-modal retrieval tasks such as image–text matching. As a result, it models queries and documents as independent distributions and measures similarity at the distribution level, without capturing fine-grained token-level interactions. This architectural mismatch limits its effectiveness for text-only retrieval, explaining the substantial performance gap we observe compared to `VarCrossEncoder`, which directly models joint probabilistic embeddings of query–document pairs.

We conducted additional experiments to evaluate the efficiency of `VarCrossEncoder` in terms of GPU memory consumption (MiB) and inference time (seconds), comparing it against representative baselines, namely MRL, EASE-DR, and PCMR. The results are summarized in Table 6. As shown, `VarCrossEncoder` requires substantially less GPU memory than the baselines while maintaining competitive inference efficiency.

Table 7 Comparative performance of the colBERTv2 [51] with `VarCrossEncoder`.

	colBERTv2				`VarCrossEncoder`			
	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.324	0.087	0.398	0.257	0.294	0.078	0.366	0.229
ELI5	0.274	0.187	0.499	0.132	0.231	0.156	0.442	0.107
COUGH	0.244	0.147	0.418	0.142	0.202	0.119	0.352	0.117
TREC-COVID	0.51	0.57	0.735	0.11	0.534	0.592	0.771	0.114
Robust04	0.428	0.408	0.656	0.095	0.455	0.432	0.69	0.106

It is important to consider that ranking methods must not only be effective but also fast in practice. We therefore analyze the inference time results as follows: (1) MRL achieves substantially lower inference latency—approximately 7× faster than `VarCrossEncoder`. This difference arises because `VarCrossEncoder`, being a cross-encoder architecture, jointly encodes query–document pairs at runtime. In contrast, MRL only computes the query representation online and then applies a Kullback–Leibler divergence operation against pre-computed document representations, making inference considerably faster. (2) Compared to EASE-DR, `VarCrossEncoder` is at least 8× faster, highlighting its advantage over other probabilistic baselines where high latency may hinder real-world deployment. (3) Although PCMR achieves the lowest inference time, this comes at the expense of substantially higher memory usage and extremely long training time.

Overall, these results indicate that `VarCrossEncoder` strikes a more practical balance between efficiency and retrieval effectiveness than existing probabilistic baselines.

It is important to note that `VarCrossEncoder`, similar to other cross-encoder architectures, is designed for use in a re-ranking setting and thus inherits the scalability limitations inherent to this family of models. Although the probabilistic module introduces only negligible computational overhead relative to a standard cross-encoder, the model is not intended to directly score entire corpora. Instead, it operates on a candidate set retrieved by a first-stage retriever (e.g., the top-100 or top-1000 documents returned by BM25 or a dense dual-encoder), within which `VarCrossEncoder` can effectively capture uncertainty and enhance ranking robustness.

Neural Ranking Models with Multiple Vector Representations: we include ColBERTv2 [51] as a representative multi-vector baseline because it is widely adopted and achieves strong performance in recent retrieval literature. ColBERTv2 encodes queries and documents into multiple contextualized vectors, which enables fine-grained lexical and semantic matching at the retrieval stage. It is therefore best understood as a high-capacity retriever. Our proposed `VarCrossEncoder`, in contrast, functions as a re-ranker that refines the candidate set returned by a first-stage retriever such as BM25. While the two approaches are not directly comparable, the contrast is instructive and points to possible synergies if combined within a hybrid pipeline. Table 7 reports results for both approaches. `VarCrossEncoder` achieves competitive performance, surpassing ColBERTv2 on TREC-COVID and Robust04 while trailing slightly on FiQA-2018, ELI5, and COUGH. These findings highlight the potential of uncertainty-aware re-ranking under distributional shift and position `VarCrossEncoder` as a complementary method to strong retrievers such as ColBERTv2.

Table 8 The performance of `VarCrossEncoder` and the baselines over the low-performance queries.

	Domain	`Cross Encoder`	`VarCrossEncoder`	MRL	EASE-DR	PCME
In-domain	MS MARCO Dev set	0.019	**0.03**	0.003	0.005	0.0004
	DL-Hard	0.055	**0.082**	0.011	0.012	0.004
	TREC DL 2019	0.253	**0.257**	0.065	0.029	0.007
	TREC DL 2020	0.261	**0.27**	0.084	0.052	0.007
	TREC DL 2021	0.208	**0.216**	0.063	0.036	0.02
Out-of-domain	FiQA-2018	0.003	**0.019**	0.013	0.003	0.001
	ELI5	0.015	0.025	**0.032**	0.008	0.002
	COUGH	0.002	0.018	**0.024**	0.005	0.002
	TREC-COVID	0.149	**0.183**	0.156	0.068	0.053
	Robust04	0.078	**0.095**	0.068	0.02	0.021

The comparison results are reported in Table 7 for out-of-domain datasets. We observe that `VarCrossEncoder` achieves competitive performance, outperforming ColBERTv2 on several datasets (e.g., TREC-COVID and Robust04), while remaining slightly behind on others (e.g., FiQA-2018, ELI5, and COUGH). These findings suggest that the proposed model provides a promising alternative for uncertainty-aware re-ranking, particularly in domains with distributional shifts.

5 Impact on Hard Queries

Zhang et al. [86] have argued that ideal ranking models are expected to be not only effective in terms of high mean retrieval performance over all queries (e.g., mean average precision (MAP)) but also stable in terms of low variance of retrieval performance across different queries. In the same direction, Wu et al. [87] introduced the concept of performance variance as a characteristic of robust ranking models (C3) and analyzed it by emphasizing the poorly performing queries. Therefore, in this section, we focus on poorly performing queries (also referred to as Hard queries) to study the robustness of `VarCrossEncoder`. For this purpose, We first introduce the evaluation metric for identifying poorly performing queries and then report our findings.

Voorhees [88] introduced the $\%no$ measure as the percentage of queries that are not able to retrieve any relevant documents in their top ten retrieved documents. The ranking model would be more robust with a lower $\%no$ value. Bigdeli et al. [89] defined the lowest-performing queries as the most difficult queries for a ranker to be those that fall in the lower half of retrieval effectiveness compared to other queries. We evaluate the robustness of `VarCrossEncoder` in terms of the poorly performing queries using the strategy proposed in [89].

Following [89], to identify poorly performing queries, we rank queries based on their performance values that are achieved by the baseline `Cross Encoder` and choose the bottom 50% of queries to constitute Hard queries. Table 8 shows the performance of state-of-the-art baseline MRL as well as the performance of our proposed model for all evaluation datasets. It is worth mentioning that the performance scores of all datasets are measured based on NDCG@10 except the Dev set which is based on the official metrics of MRR@10. The highest performance is shown in bold for each evaluation set.

As shown in Table 8, the performance of our proposed approach is higher than MRL in all datasets except ELI5 and COUGH, which means that `VarCrossEncoder` has been able to address a larger subset of the poorly performing queries compared to

the baselines. This shows that when the representations of the query-document pairs are mapped into the latent probabilistic space, a noticeable number of low-performing queries are then effectively, or at least partially, addressed by `VarCrossEncoder`.

6 Discussion

The primary objective of the discussion section is to conduct an in-depth examination of the factors that influence the effectiveness of `VarCrossEncoder`. In particular, we focus on two critical aspects: (1) the role of the Information Bottleneck's β parameter, which governs the trade-off between preserving task-relevant information and enforcing compression in the learned representations; and (2) the impact of different pre-trained language models used as the backbone encoders. By systematically analyzing these dimensions, we aim to provide insights into how hyperparameter choices and architectural variations affect both the efficiency and the overall retrieval performance of the proposed model.

Impact of the Information Bottleneck's β Parameter: Based on equation 9, the Information Bottleneck parameter, denoted as β, controls the trade-off between compression and prediction in probabilistic representation learning. Its value directly influences the performance of `VarCrossEncoder` by determining the extent to which irrelevant information is suppressed during encoding. To evaluate the sensitivity of the model to the β parameter, we first outline the strategy adopted for its selection. We employed a cross-set validation approach, whereby β was tuned on a dataset distinct from the one used for evaluation. This strategy reduces the risk of overfitting the hyperparameter to a specific benchmark and provides a more reliable estimate of the model's generalization ability. For example, when assessing performance on TREC DL 2022, the parameter was optimized using TREC DL 2021, and vice versa. Following [67], we conducted experiments across multiple β values, ranging from 10^{-9} to 10^{-5}. For these experiments, we adopted MeanVarM as the probabilistic module within the model.

The Figure 6 illustrates the performance trends of VarCrossEncoder across different datasets as the information bottleneck's β parameter varies from 10^{-9} to 10^{-5}. Several conclusions can be drawn from observing the results presented in the figure: (1) Overall, the model demonstrates stable performance within a range of β values, with noticeable variation depending on the evaluation set. (2) Most datasets, such as TREC LD 2019, TREC LD 2020, and FiQA-2018, show stable performance across the full range of β values, suggesting that VarCrossEncoder is not highly sensitive to this hyperparameter. (3) Datasets like TREC LD 2021 and DL-hard benefit slightly from β values around 10^{-7} to 10^{-6}, where scores reach their highest levels. This indicates that a moderate degree of compression may help filter out irrelevant information without overly constraining the predictive capacity of the model. (4) At 10^{-5}, several datasets, such as TREC DL 2020, TREC-COVID, and ELI5, exhibit a small but consistent decline, implying that excessive emphasis on compression can negatively impact retrieval effectiveness.

Impact of Pre-trained Language Models: The backbone encoder in our framework is designed to provide deterministic point representations of input queries

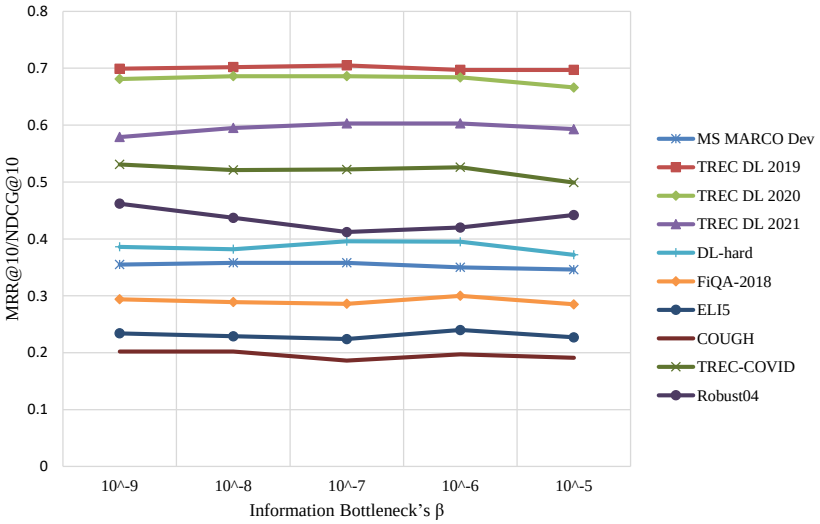

Fig. 6 Impact of Information Bottleneck's β on the Performance of our Proposed `VarCrossEncoder` in terms of NDCG@10 (MRR@10 for Dev set).

and documents within the latent space by leveraging pre-trained language models. To assess the influence of different pre-trained language models on the effectiveness of the proposed model, we conduct a series of experiments where four alternative encoders—DeBERTa [90], RoBERTa [91], ALBERT [92], and BERT [63]—are used in place of DistilBERT [81]. We then evaluate the performance of `VarCrossEncoder` across both in-domain and out-of-domain datasets.

The results, reported in Table 9, demonstrate that while DistilBERT achieves competitive performance, larger pre-trained language models such as DeBERTa and RoBERTa generally provide performance gains in several datasets, particularly in FiQA-2018, ELI5, and Robust04. However, the performance trends also suggest that the relative advantage of larger pre-trained language models is not uniform across all evaluation sets, indicating that the choice of backbone encoder should be informed by domain-specific requirements.

Table 9 Comparison of `VarCrossEncoder` performance when using different pre-trained language models as the backbone encoder across in-domain and out-of-domain datasets.

	Domain	DeBERTa	RoBERTa	ALBERT	BERT	DistilBERT
In-domain	MS MARCO Dev set	0.367	0.351	0.368	0.364	0.358
	DL-Hard	0.393	0.399	0.379	0.384	0.403
	TREC DL 2019	0.716	0.673	0.716	0.708	0.699
	TREC DL 2020	0.651	0.655	0.685	0.685	0.69
	TREC DL 2021	0.633	0.584	0.602	0.613	0.606
Out-of-domain	FiQA-2018	0.381	0.327	0.327	0.332	0.299
	ELI5	0.325	0.265	0.265	0.286	0.229
	COUGH	0.248	0.232	0.232	0.141	0.199
	TREC-COVID	0.567	0.548	0.548	0.436	0.539
	Robust04	0.475	0.463	0.473	0.285	0.443

Table 10 The performance of the baseline (trained on l_{det}) (left), `VarCrossEncoder` with the VAR-based module (trained on l) (middle), and `VarCrossEncoder` with the VAR-based module when trained with the synthetic loss function of $\ell_{synthetic}$ (right).

	`Cross Encoder`				`VarCrossEncoder`				`VarCrossEncoder` ($\ell_{synthetic}$)			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.277	0.075	0.35	0.212	0.299	0.08	0.371	0.234	0.293	0.08	0.363	0.228
ELI5	0.215	0.146	0.421	0.097	0.229	0.155	0.443	0.106	0.226	0.152	0.434	0.104
COUGH	0.177	0.104	0.311	0.102	0.199	0.117	0.35	0.116	0.181	0.105	0.322	0.104
TREC-COVID	0.514	0.6	0.719	0.011	0.539	0.614	0.799	0.012	0.534	0.6	0.791	0.012
Robust04	0.42	0.402	0.655	0.096	0.443	0.425	0.668	0.105	0.429	0.41	0.661	0.098

7 Ablation Study

Our proposed `VarCrossEncoder` approach uses the loss function defined in Equation 21 to learn the model parameters. Given that our proposed model can accommodate different types of loss functions, the main goal of the ablation study is to investigate the impact of different loss functions on the performance of `VarCrossEncoder`.

Equation 5 indicates that all identity embeddings $\mu(t)$ are impacted by $\Sigma(t)$ during the training period. This will prompt the model to predict small Σ for all samples in order to suppress the unstable ingredients in $\widehat{s}$ such that the loss can still converge at the end. As such, given $[CLS]$ as the original deterministic representation, we can apply a deterministic score $\widehat{s_{det}} = \phi_{det}([CLS] \cdot W_{det})$ where W_{det} and $\phi_{det}(.)$ is the same as W and $f_{\varphi}^{Rank}(.)$, respectively. The adoption of $\widehat{s_{det}}$ leads to introducing a deterministic loss, as ℓ_{det} based on Equation 18 where $\sigma(\widehat{s_{det}})$ is used instead of $q(y = 1|[CLS])$. Now, we can define a multi-objective loss function as follows:

$$\ell_{synthetic} = \lambda \cdot \ell + (1 - \lambda) \cdot \ell_{det} \tag{27}$$

where λ is a trade-off hyper-parameter. Finally, we apply $L_{synthetic}$ as the total cost function. We would like to identify whether the linear interpolation of the deterministic loss ℓ_{det} and probabilistic loss ℓ have complementary and synergistic behavior on each other. Tables 10-12 show model performances on the ranking task for each probabilistic module separately when the `VarCrossEncoder` is trained based on synthetic loss $\ell_{synthetic}$. In these experiments, the value of λ is set to 0.5.

Given that the baseline and `VarCrossEncoder` are `Cross Encoders` that are trained based on only the deterministic and probabilistic cost functions, respectively,

Table 11 The performance of the baseline (trained on l_{det}) (left), `VarCrossEncoder` with the MeanVar-based module (trained on l) (middle), and `VarCrossEncoder` with the MeanVar-based module when trained with the synthetic loss function of $\ell_{synthetic}$ (right).

	Cross Encoder				VarCrossEncoder				VarCrossEncoder ($\ell_{synthetic}$)			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.277	0.075	0.35	0.212	0.294	0.078	0.366	0.229	0.289	0.078	0.363	0.225
ELI5	0.215	0.146	0.421	0.097	0.231	0.156	0.442	0.107	0.224	0.153	0.43	0.102
COUGH	0.177	0.104	0.311	0.102	0.202	0.119	0.352	0.117	0.196	0.115	0.342	0.114
TREC-COVID	0.514	0.6	0.719	0.011	0.534	0.592	0.771	0.011	0.502	0.574	0.759	0.011
Robust04	0.42	0.402	0.655	0.096	0.455	0.432	0.69	0.106	0.44	0.42	0.681	0.101

Table 12 The performance of the baseline (trained on l_{det}) (left), `VarCrossEncoder` with the Mean-based module (trained on l) (middle), and `VarCrossEncoder` with the Mean-based module when trained with the synthetic loss function of $\ell_{synthetic}$ (right).

	Cross Encoder				VarCrossEncoder				VarCrossEncoder ($\ell_{synthetic}$)			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.277	0.075	0.35	0.212	0.28	0.077	0.343	0.216	0.285	0.079	0.347	0.218
ELI5	0.215	0.146	0.421	0.097	0.219	0.148	0.423	0.101	0.227	0.155	0.431	0.104
COUGH	0.177	0.104	0.311	0.102	0.185	0.109	0.316	0.108	0.188	0.111	0.317	0.109
TREC-COVID	0.514	0.6	0.719	0.011	0.516	0.584	0.765	0.011	0.497	0.55	0.814	0.01
Robust04	0.42	0.402	0.655	0.096	0.45	0.441	0.665	0.105	0.434	0.419	0.645	0.101

in this scenario, we are interested to see whether the linear integration of both losses in `VarCrossEncoder` is able to act as effectively as when the models are trained separately based on each individual loss. As shown in Table 10, we observe that `VarCrossEncoder` with the Var-based module significantly outperforms the `Cross Encoder` when trained based on synthetic loss over five out-of-domain datasets in terms of all evaluation metrics, which shows the effectiveness of the synthetic loss $\ell_{synthetic}$ over the deterministic loss ℓ for the ranking task. However, `VarCrossEncoder` with synthetic loss shows poor performance against `VarCrossEncoder` with the probabilistic loss when the Var-based module is applied. We can make the same observation for `VarCrossEncoder` with the MeanVar-based module based on the results reported in Table 11.

As reported in Table 12, adopting the Mean-based module as the probabilistic module, the performance of `VarCrossEncoder` with synthetic loss is better than its peers with the probabilistic and deterministic loss in three out of five datasets of FiQA-2018, ELI5, and COUGH. In TREC-COVID and Robust04, `VarCrossEncoder` with only the probabilistic module would be able to provide better performance.

Finally, we have conducted experiments on the in-distribution datasets to investigate the impact of the synthetic loss $\ell_{synthetic}$ on the ranking performance. The performance of `VarCrossEncoder` is reported in Table 13 when trained based on the synthetic loss $\ell_{synthetic}$ in terms of MRR@10 for the Dev set and NDCG@10 for other evaluation sets for each probabilistic module separately. As shown in the table, we can generally conclude that `VarCrossEncoder` are not able to improve the ranking performance of the baseline when trained based on $\ell_{synthetic}$ (except over DL-Hars, TREC DL 2020 and 2021 with Var-based module and over TREC DL 2019 with MeanVar-based module). Also, `VarCrossEncoder` with $\ell_{synthetic}$ shows poor performance in comparison to `VarCrossEncoder` when trained based on only the probabilistic loss.

To further analyze and understand the importance of the synthetic loss, we have performed additional experiments for different values of λ and shown the performance of `VarCrossEncoder` for each probabilistic module separately for all in-distribution

Table 13 The performance of the baseline (trained on l_{det}) (left), `VarCrossEncoder` (trained on l) (middle), and `VarCrossEncoder` when trained with the synthetic loss function of $\ell_{synthetic}$ (right).

	Cross Encoder	VarCrossEncoder			VarCrossEncoder ($\ell_{synthetic}$)		
		VarM	MeanVarM	MeanM	VarM	MeanVarM	MeanM
MS MARCO Dev set	0.356	0.358	0.355	0.345	0.354	0.351	0.337
DL-Hard	0.386	0.403	0.391	0.363	0.399	0.385	0.369
TREC DL 2019	0.698	0.699	0.712	0.687	0.689	0.7	0.692
TREC DL 2020	0.678	0.69	0.682	0.686	0.684	0.674	0.675
TREC DL 2021	0.602	0.606	0.587	0.571	0.605	0.601	0.579

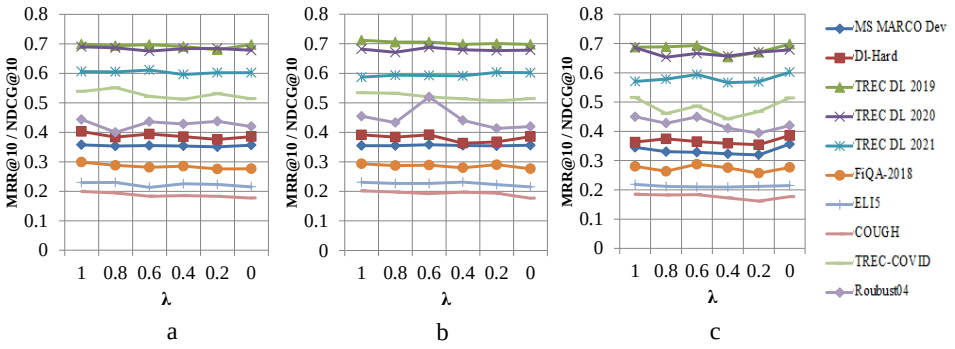

Fig. 7 The Performance of our Proposed `VarCrossEncoder` with (a) VarM-based module. (b) MeanVarM-based module. (c) MeanM-based module for different values of λ in terms of NDCG@10 (MRR@10 for Dev set).

and out-of-distribution datasets in terms of NDCG@10 (MRR@10 for Dev set) in Figure 7. In the in-distribution datasets, the highest performance is achieved for λ of 1 and 0 when the VarM-based and MeanM-based modules are adapted in the model, respectively. Regardless of which probabilistic module is used in the model, the performance of `VarCrossEncoder` is higher when the value of λ is set to 1, which points to cases when the model is trained only based on the probabilistic loss, in three of the five out-of-distribution datasets except Robust04 and TREC-COVID. Based on the performance shown in this Figure, we can conclude that λ values do not have much meaningful impact on the model's performance, especially over the FiQA 2018, ELI5, and COUGH datasets.

In order to study the impact of the loss function on the model's performance, we introduce a new cost function in such a way that we first define a synthetic score, given $\widehat{s}$ and $\widehat{s_{det}}$ as follows:

$$\widehat{s_{synthetic}} = \lambda \cdot \widehat{s} + (1 - \lambda) \cdot \widehat{s_{det}} \tag{28}$$

and then compute the loss based on Equation 21 in which $\widehat{s_{synthetic}}$ is used. Tables 14-16 show the performance of `VarCrossEncoder` when the model's parameters are learned through the cost function that is based on the synthetic score $\widehat{s_{synthetic}}$ in

Table 14 The performance of the baseline (trained based on $\widehat{s_{det}}$) (left), `VarCrossEncoder` with the VAR-based module (trained on $\widehat{s}$) (middle), and `VarCrossEncoder` with the VAR-based module when trained with the synthetic score function of $s_{\widehat{synthetic}}$ (right).

	Cross Encoder				VarCrossEncoder				VarCrossEncoder ($s_{\widehat{synthetic}}$)			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.277	0.075	0.35	0.212	0.299	0.08	0.371	0.234	0.285	0.079	0.354	0.221
ELI5	0.215	0.146	0.421	0.097	0.229	0.155	0.443	0.106	0.229	0.155	0.435	0.106
COUGH	0.177	0.104	0.311	0.102	0.199	0.117	0.35	0.116	0.196	0.115	0.341	0.114
TREC-COVID	0.514	0.6	0.719	0.011	0.539	0.614	0.799	0.012	0.507	0.586	0.742	0.011
Robust04	0.42	0.402	0.655	0.096	0.443	0.425	0.668	0.105	0.464	0.446	0.701	0.107

Table 15 The performance of the baseline (trained based on s_{det}) (left), `VarCrossEncoder` with the MEANVAR-based module (trained based on $\widehat{s}$) (middle), and `VarCrossEncoder` with the MEANVAR-based module when trained with the synthetic score function of $s_{\widehat{synthetic}}$ (right).

	Cross Encoder				VarCrossEncoder				VarCrossEncoder ($s_{\widehat{synthetic}}$)			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.277	0.075	0.35	0.212	0.294	0.078	0.366	0.229	0.291	0.079	0.361	0.225
ELI5	0.215	0.146	0.421	0.097	0.231	0.156	0.442	0.107	0.225	0.152	0.434	0.104
COUGH	0.177	0.104	0.311	0.102	0.202	0.119	0.352	0.117	0.191	0.111	0.339	0.111
TREC-COVID	0.514	0.6	0.719	0.011	0.534	0.592	0.771	0.011	0.5	0.568	0.763	0.01
Robust04	0.42	0.402	0.655	0.096	0.455	0.432	0.69	0.106	0.45	0.431	0.692	0.105

Table 16 The performance of the baseline (trained based on $\widehat{s_{det}}$) (left), `VarCrossEncoder` with the Mean-based module (trained based on $\widehat{s}$) (middle), and `VarCrossEncoder` with the Mean-based module when trained with the synthetic score of $s_{\widehat{synthetic}}$ (right).

	Cross Encoder				VarCrossEncoder				VarCrossEncoder ($s_{\widehat{synthetic}}$)			
Domain	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10	NDCG@10	P@10	MRR@10	AP@10
FiQA-2018	0.277	0.075	0.35	0.212	0.28	0.077	0.343	0.216	0.273	0.074	0.337	0.211
ELI5	0.215	0.146	0.421	0.097	0.219	0.148	0.423	0.101	0.21	0.146	0.397	0.095
COUGH	0.177	0.104	0.311	0.102	0.185	0.109	0.316	0.108	0.183	0.108	0.316	0.105
TREC-COVID	0.514	0.6	0.719	0.011	0.516	0.584	0.765	0.011	0.48	0.534	0.732	0.009
Robust04	0.42	0.402	0.655	0.096	0.45	0.441	0.665	0.105	0.436	0.417	0.695	0.099

terms of NDCG@10 for each probabilistic module separately. The value of λ is set to 0.5 in these experiments.

Based on the results reported in Tables 14-16, an important observation is that adapting the synthetic score $s_{\widehat{synthetic}}$ in the final loss function leads to no improvements against settings where only the probabilistic score s is used in the loss regardless of which types of probabilistic modules are applied in the model (except Robust04). Also, the ranking performance of `VarCrossEncoder` with $s_{\widehat{synthetic}}$ over in-distribution datasets is reported in Table 17 for all probabilistic models in terms of NDCG@10 (MRR@10 for Dev set). Based on the results, we can see that `VarCrossEncoder` with VARM and MEANVARM outperforms the baseline when the synthetic score $s_{\widehat{synthetic}}$ is used in the final loss function. Furthermore, In order to investigate the impact of λ on $s_{\widehat{synthetic}}$ and also on the final performance, we depict the performance of `VarCrossEncoder`in terms of NDCG@10 for each probabilistic module separately in Figure 8. As shown in the Figure, the highest performance is when λ is set to 1 for all probabilistic modules, which means that `VarCrossEncoder` has the best performance in ranking when the final score is computed based on only the probabilistic score.

8 Concluding Remarks

In this work, we have proposed `VarCrossEncoder`, a Probabilistic-Neural Ranking Model, to estimate *data uncertainty* in the retrieval task. `VarCrossEncoder` relies on

Table 17 The performance of the baseline (trained based on $\widehat{s_{det}}$) (left), `VarCrossEncoder` (trained on $\widehat{s}$) (middle), and `VarCrossEncoder` when trained with the synthetic score function of $\widehat{s_{synthetic}}$ (right).

	`Cross Encoder`	VarCrossEncoder			`VarCrossEncoder` ($\widehat{s_{synthetic}}$)		
		`VarM`	`MeanVarM`	`MeanM`	`VarM`	`MeanVarM`	`MeanM`
MS MARCO Dev set	0.356	0.358	0.355	0.345	0.357	0.354	0.307
DL-Hard	0.386	0.403	0.391	0.363	0.375	0.405	0.355
TREC DL 2019	0.698	0.699	0.712	0.687	0.716	0.705	0.674
TREC DL 2020	0.678	0.69	0.682	0.686	0.679	0.691	0.649
TREC DL 2021	0.602	0.606	0.587	0.571	0.614	0.585	0.547

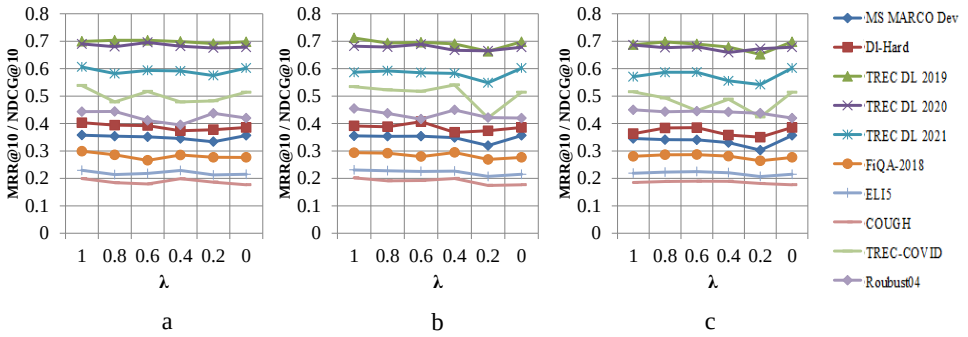

Fig. 8 The Performance of our Proposed `VarCrossEncoder` with (a) VAR-based module. (b) MEAN-VAR-based module. (c) MEAN-based module for different values of λ in terms of NDCG@10 (MRR@10 for Dev set).

learning the probabilistic embeddings for each sample of the query-document pair in order to give a distributional estimation instead of a point estimation in the latent space. We demonstrated that query-document uncertainty can be represented as a Gaussian distribution by adding only the probabilistic module to the state-of-the-art dense retriever of the `Cross Encoder`. Also, we have proposed three types of probabilistic modules depending on which parameters of the Gaussian distribution (mean or variance) are learned. Our proposed model has been evaluated by performing a series of experiments over two groups of query sets, in-distribution and out-of-distribution sets.

Our proposed model has been evaluated by performing a series of experiments over two groups of query sets, in-distribution and out-of-distribution sets. The summary of our key findings is as follows:

- The application of data uncertainty leads to improvement in the performance of the ranking task. Depending on which parameters of the Gaussian distribution are learned, the degree of improvement may be different. The ranking performance is more considerable when uncertainty is learned using variance instead of fixing it.
- Given the `Cross Encoder` is a well-known dense retriever, `VarCrossEncoder` has improved its performance consistently over all in-distribution datasets when it is incorporated with data uncertainty.

- Since the feature statistics of the unseen domain are often not consistent with the training domain due to differing domain characteristics, our proposed `VarCrossEncoder` approach has effectively improved the performance of the baseline over the out-of-distribution datasets in terms of all evaluation metrics.

We would like to extend our work in the following directions:

- `VarCrossEncoder` considers uncertainty by modeling each query-document pair embedding as a Gaussian distribution such that feature (mean) and uncertainty (variance) are learned simultaneously. Other distributions (such as the Dirichlet distribution as a common way to model a textual document) can be used to take uncertainty into account. To this end, we would like to extend the work in this paper further to an end-to-end model that is able to choose a proper distribution for input samples, and hence we are interested in exploring its potential role under a probabilistic ranking scenario.
- Uncertainty modeling by Gaussian distribution leads to mapping each input data to an area instead of a point in the latent space, such that the parameters of mean and variance demonstrate the center and spread of the area respectively while one can define the spread of the area by a neighborhood radius.
- Given that there are some well-known neural ranking models such as PreTTR, and CoCondenser, and given uncertainty is captured via the independent probabilistic module in our proposed `VarCrossEncoder`, we are interested in studying whether one can adapt this module in other such ranking models to consider uncertainty.

References

[1] Chang, J., Lan, Z., Cheng, C., Wei, Y.: Data uncertainty learning in face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5710–5719 (2020)

[2] Salton, G., Wong, A., Yang, C.-S.: A vector space model for automatic indexing. Communications of the ACM **18**(11), 613–620 (1975)

[3] Robertson, S.: Understanding inverse document frequency: on theoretical arguments for idf. Journal of documentation **60**(5), 503–520 (2004)

[4] Aizawa, A.: An information-theoretic perspective of tf–idf measures. Information Processing & Management **39**(1), 45–65 (2003)

[5] Kong, W., Dudek, J.M., Li, C., Zhang, M., Bendersky, M.: Sparseembed: Learning sparse lexical representations with contextual embeddings for retrieval. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2399–2403 (2023)

[6] Rajapakse, T.C.: Dense passage retrieval: Architectures and augmentation methods. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 3494–3494 (2023)

[7] Fan, Y., Xie, X., Cai, Y., Chen, J., Ma, X., Li, X., Zhang, R., Guo, J., *et al.*: Pre-training methods in information retrieval. Foundations and Trends® in Information Retrieval **16**(3), 178–317 (2022)

[8] Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P.S., He, L.: A survey on text classification: From traditional to deep learning. ACM Transactions on Intelligent Systems and Technology (TIST) **13**(2), 1–41 (2022)

[9] Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., *et al.*: A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology **15**(3), 1–45 (2024)

[10] Zhao, W.X., Liu, J., Ren, R., Wen, J.-R.: Dense text retrieval based on pretrained language models: A survey. ACM Transactions on Information Systems **42**(4), 1–60 (2024)

[11] Ma, X., Wang, L., Yang, N., Wei, F., Lin, J.: Fine-tuning llama for multi-stage text retrieval. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2421–2425 (2024)

[12] Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)

[13] MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: Cedr: Contextualized embeddings for document ranking. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1101–1104 (2019)

[14] Liu, L., Chen, Y., Das, M., Yang, H., Tong, H.: Knowledge graph question answering with ambiguous query. In: Proceedings of the ACM Web Conference 2023, pp. 2477–2486 (2023)

[15] Arabzadeh, N., Hamidi Rad, R., Khodabakhsh, M., Bagheri, E.: Noisy perturbations for estimating query difficulty in dense retrievers. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 3722–3727 (2023)

[16] Khodabakhsh, M., Bagheri, E.: Learning to rank and predict: multi-task learning for ad hoc retrieval and query performance prediction. Information Sciences **639**, 119015 (2023)

[17] Ebrahimi, S., Khodabakhsh, M., Arabzadeh, N., Bagheri, E.: Estimating query performance through rich contextualized query representations. In: European Conference on Information Retrieval, pp. 49–58 (2024). Springer

[18] Liu, Y.-A., Zhang, R., Guo, J., Rijke, M.: Robust information retrieval. In: Proceedings of the 47th International ACM SIGIR Conference on Research and

Development in Information Retrieval, pp. 3009–3012 (2024)

[19] Ding, W., Geng, X., Zhang, X.-D.: Learning to rank from noisy data. ACM Transactions on Intelligent Systems and Technology (TIST) **7**(1), 1–21 (2015)

[20] Chen, X., He, B., Hui, K., Sun, L., Sun, Y.: Dealing with textual noise for robust and effective bert re-ranking. Information Processing & Management **60**(1), 103135 (2023)

[21] Zhu, J., Wang, J., Taylor, M., Cox, I.J.: 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, pp. 17–28 (2009). Springer

[22] Cohen, D., Mitra, B., Lesota, O., Rekabsaz, N., Eickhoff, C.: 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, pp. 654–664 (2021)

[23] Zamani, H., Bendersky, M.: Multivariate representation learning for information retrieval. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 163–173 (2023)

[24] Jousselme, A.-L., Maupin, P.: In: Rogova, G., Scott, P. (eds.) Uncertainty Representations for Information Retrieval with Missing Data, pp. 87–104. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-22527-2_5 . https://doi.org/10.1007/978-3-319-22527-2_5

[25] Ji, Y., Wang, J., Gong, Y., Zhang, L., Zhu, Y., Wang, H., Zhang, J., Sakai, T., Yang, Y.: Map: Multimodal uncertainty-aware vision-language pre-training model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 23262–23271 (2023)

[26] Upadhyay, U., Karthik, S., Mancini, M., Akata, Z.: Probvlm: Probabilistic adapter for frozen vison-language models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1899–1910 (2023)

[27] Zhang, D., Wu, X.-J., Yu, J.: Discrete bidirectional matrix factorization hashing for zero-shot cross-media retrieval. In: Pattern Recognition and Computer Vision, pp. 524–536 (2021)

[28] Zhang, D., Wu, X.-J., Chen, G.: Onion: Online semantic autoencoder hashing for cross-modal retrieval. ACM Trans. Intell. Syst. Technol. **14**(2) (2023) https://doi.org/10.1145/3572032

[29] Hu, Z., Cheung, Y.-M., Li, M., Lan, W., Zhang, D., Liu, Q.: Joint semantic preserving sparse hashing for cross-modal retrieval. IEEE Transactions on Circuits

and Systems for Video Technology **34**(4), 2989–3002 (2024)

[30] Chun, S., Oh, S.J., Rezende, R.S., Kalantidis, Y., Larlus, D.: Probabilistic embeddings for cross-modal retrieval. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8415–8424 (2021)

[31] Gawlikowski, J., Tassi, C.R.N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., *et al.*: A survey of uncertainty in deep neural networks. Artificial Intelligence Review **56**(Suppl 1), 1513–1589 (2023)

[32] Blundell, C., Cornebise, J., Kavukcuoglu, K., Wierstra, D.: Weight uncertainty in neural network. In: International Conference on Machine Learning, pp. 1613–1622 (2015). PMLR

[33] Kendall, A., Badrinarayanan, V., Cipolla, R.: Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. In: British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017 (2017). https://www.dropbox.com/s/jgozsaobbk98azy/0205.pdf

[34] Isobe, S., Arai, S.: Deep convolutional encoder-decoder network with model uncertainty for semantic segmentation. In: 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA), pp. 365–370 (2017). IEEE

[35] Choi, J., Chun, D., Kim, H., Lee, H.-J.: Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 502–511 (2019)

[36] Zafar, U., Ghafoor, M., Zia, T., Ahmed, G., Latif, A., Malik, K.R., Sharif, A.M.: Face recognition with bayesian convolutional networks for robust surveillance systems. EURASIP Journal on Image and Video Processing **2019**, 1–10 (2019)

[37] Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? Advances in neural information processing systems **30** (2017)

[38] Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: ACM SIGIR Forum, vol. 51, pp. 202–208 (2017). ACM New York, NY, USA

[39] Santos, C., Ma, X., Nallapati, R., Huang, Z., Xiang, B.: Beyond [CLS] through ranking by generation. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1722–1727 (2020). https://doi.org/10.18653/v1/2020.emnlp-main.134

[40] Penha, G., Hauff, C.: 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, pp. 160–170 (2021)

[41] Yang, T., Luo, C., Lu, H., Gupta, P., Yin, B., Ai, Q.: Can clicks be both labels and features? unbiased behavior feature collection and uncertainty-aware learning to rank. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 6–17 (2022)

[42] Cronen-Townsend, S., Zhou, Y., Croft, W.B.: Predicting query performance. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 299–306 (2002)

[43] Shtok, A., Kurland, O., Carmel, D., Raiber, F., Markovits, G.: Predicting query performance by query-drift estimation. ACM Transactions on Information Systems (TOIS) **30**(2), 1–35 (2012)

[44] Tao, Y., Wu, S.: Query performance prediction by considering score magnitude and variance together. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pp. 1891–1894 (2014)

[45] Chun, S.: Improved probabilistic image-text representations. In: The Twelfth International Conference on Learning Representations (2024). https://openreview.net/forum?id=ft1mr3WlGM

[46] Wei, W., Gui, Z., Wu, C., Zhao, A., Peng, D., Wu, H.: Dynamic visual semantic sub-embeddings and fast re-ranking for image-text retrieval. IEEE Transactions on Multimedia **27**, 3781–3796 (2025) https://doi.org/10.1109/TMM.2025.3535373

[47] Li, H., Song, J., Gao, L., Zhu, X., Shen, H.T.: Prototype-based aleatoric uncertainty quantification for cross-modal retrieval. In: Proceedings of the 37th International Conference on Neural Information Processing Systems (2023)

[48] Shi, Y., Jain, A.K.: Probabilistic face embeddings. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6902–6911 (2019)

[49] Mei, L., Mao, J., Guo, G., Wen, J.-R.: Learning probabilistic box embeddings for effective and efficient ranking. In: Proceedings of the ACM Web Conference 2022. WWW '22, pp. 473–482. Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3485447.3512073 . https://doi.org/10.1145/3485447.3512073

[50] Zhou, X., Gao, Y., Jie, X., Cai, X., Bu, J., Wang, H.: Ease-dr: Enhanced sentence embeddings for dense retrieval. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2374–2378 (2024)

[51] Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., Zaharia, M.: 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, pp. 3715–3734. Association for Computational Linguistics, Seattle, United States (2022). https://doi.org/10.18653/v1/2022.naacl-main.272 . https://aclanthology.org/2022.naacl-main.272/

[52] Kong, W., Khadanga, S., Li, C., Gupta, S.K., Zhang, M., Xu, W., Bendersky, M.: Multi-aspect dense retrieval. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3178–3186 (2022)

[53] Zhou, G., Devlin, J.: Multi-vector attention models for deep re-ranking. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 5452–5456 (2021)

[54] Warburg, F., Jørgensen, M., Civera, J., Hauberg, S.: Bayesian triplet loss: Uncertainty quantification in image retrieval. 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 12138–12148 (2020)

[55] Mitra, B., Diaz, F., Craswell, N.: Learning to match using local and distributed representations of text for web search. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1291–1299 (2017)

[56] Arabzadeh, N., Mitra, B., Bagheri, E.: Ms marco chameleons: challenging the ms marco leaderboard with extremely obstinate queries. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 4426–4435 (2021)

[57] Zhang, P., Song, D., Wang, J., Hou, Y.: Bias-variance decomposition of ir evaluation. In: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1021–1024 (2013)

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

[59] Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P.N., Ahmed, J., Overwijk, A.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: International Conference on Learning Representations (2020)

[60] MacAvaney, S., Nardini, F.M., Perego, R., Tonellotto, N., Goharian, N., Frieder, O.: Expansion via prediction of importance with contextualization. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1573–1576 (2020)

[61] Khattab, O., Zaharia, M.: 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, pp. 39–48 (2020)

[62] Humeau, S., Shuster, K., Lachaux, M.-A., Weston, J.: Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In: International Conference on Learning Representations (2019). https://api.semanticscholar.org/CorpusID:210063976

[63] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: North American Chapter of the Association for Computational Linguistics (2019)

[64] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (2013)

[65] Tishby, N., Pereira, F.C., Bialek, W.: The information bottleneck method. In: Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing, pp. 368–377 (1999). https://arxiv.org/abs/physics/0004057

[66] Voloshynovskiy, S., Kondah, M., Rezaeifar, S., Taran, O., Holotyak, T., Rezende, D.J.: Information bottleneck through variational glasses. In: 4th Workshop on Bayesian Deep Learning (NeurIPS 2019) (2019)

[67] Alemi, A.A., Fischer, I., Dillon, J.V., Murphy, K.: Deep variational information bottleneck. In: ICLR (2017)

[68] Rasmussen, C.: A practical monte carlo implementation of bayesian learning. Advances in Neural Information Processing Systems **8** (1995)

[69] Rezende, D.J., Mohamed, S.: Variational inference with normalizing flows. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, pp. 1530–1538. JMLR.org, ??? (2015)

[70] Fortuin, V.: Priors in bayesian deep learning: A review. International Statistical Review **90**(3), 563–591 (2022) https://doi.org/10.1111/insr.12502 https://onlinelibrary.wiley.com/doi/pdf/10.1111/insr.12502

[71] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., Deng, L.: Ms marco: A human generated machine reading comprehension dataset. In: CoCo@ NIPs (2016)

[72] Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Voorhees, E.M.: Overview of the trec 2019 deep learning track. In: Text REtrieval Conference (TREC) (2020)

[73] Craswell, N., Mitra, B., Yilmaz, E., Campos, D.: Overview of the TREC 2020

deep learning track. CoRR **abs/2102.07662** (2021) 2102.07662

[74] Mackie, I., Dalton, J., Yates, A.: How deep is your learning: the dl-hard annotated deep learning dataset. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2335–2341 (2021)

[75] Craswell, N., Mitra, B., Yilmaz, E., Campos, D.F., Lin, J.: Overview of the trec 2021 deep learning track. In: TREC (2021)

[76] Maia, M., Handschuh, S., Freitas, A., Davis, B., McDermott, R., Zarrouk, M., Balahur, A.: Www'18 open challenge: financial opinion mining and question answering. In: Companion Proceedings of the the Web Conference 2018, pp. 1941–1942 (2018)

[77] Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., Auli, M.: ELI5: long form question answering. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3558–3567 (2019). https://doi.org/10.18653/v1/p19-1346

[78] Petroni, F., Piktus, A., Fan, A., Lewis, P., Yazdani, M., Cao, N.D., Thorne, J., Jernite, Y., Plachouras, V., Rocktaschel, T., Riedel, S.: Kilt: a benchmark for knowledge intensive language tasks. In: North American Chapter of the Association for Computational Linguistics (2020)

[79] Zhang, X.F., Sun, H., Yue, X., Lin, S., Sun, H.: COUGH: A challenge dataset and models for COVID-19 FAQ retrieval. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, pp. 3759–3769 (2021)

[80] Wang, L.L., Lo, K., Chandrasekhar, Y., Reas, R., Yang, J., Burdick, D., Eide, D., Funk, K., Katsis, Y., Kinney, R.M., Li, Y., Liu, Z., Merrill, W., Mooney, P., Murdick, D.A., Rishi, D., Sheehan, J., Shen, Z., Stilson, B., Wade, A.D., Wang, K., Wang, N.X.R., Wilhelm, C., Xie, B., Raymond, D.M., Weld, D.S., Etzioni, O., Kohlmeier, S.: CORD-19: The COVID-19 open research dataset. In: Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020 (2020)

[81] Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. ArXiv **abs/1910.01108** (2019)

[82] Giamphy, E., Sanchis, K., Dashyan, G., Guillaume, J.-L., Hamdi, A., Sanselme, L., Doucet, A.: A quantitative analysis of noise impact on document ranking. In: 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 4612–4618 (2023). IEEE

[83] Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word

representations in vector space. In: International Conference on Learning Representations (2013). https://api.semanticscholar.org/CorpusID:5959482

[84] Ma, E.: NLP Augmentation. https://github.com/makcedward/nlpaug (2019)

[85] Zhou, X., Gao, Y., Jie, X., Cai, X., Bu, J., Wang, H.: Ease-dr: Enhanced sentence embeddings for dense retrieval. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2374–2378 (2024)

[86] Zhang, P., Song, D., Wang, J., Hou, Y.: Bias–variance analysis in estimating true query model for information retrieval. Information processing & management **50**(1), 199–217 (2014)

[87] Wu, C., Zhang, R., Guo, J., Fan, Y., Cheng, X.: Are neural ranking models robust? ACM Transactions on Information Systems **41**(2), 1–36 (2022)

[88] Voorhees, E.M., *et al.*: Overview of the trec 2003 robust retrieval track. In: Trec, pp. 69–77 (2003)

[89] Bigdeli, A., Arabzadeh, N., Bagheri, E.: Learning to jointly transform and rank difficult queries. In: European Conference on Information Retrieval, pp. 40–48 (2024). Springer

[90] He, P., Liu, X., Gao, J., Chen, W.: Deberta: Decoding-enhanced bert with disentangled attention. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=XPZIaotutsD

[91] Zhuang, L., Wayne, L., Ya, S., Jun, Z.: A robustly optimized BERT pre-training approach with post-training. In: Proceedings of the 20th Chinese National Conference on Computational Linguistics, pp. 1218–1227. Chinese Information Processing Society of China, Huhhot, China (2021). https://aclanthology.org/2021.ccl-1.108/

[92] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A lite BERT for self-supervised learning of language representations. In: 8th International Conference on Learning Representations (2020)