I^3 Retriever: Incorporating Implicit Interaction in Pre-trained Language Models for Passage Retrieval

Qian Dong
dq22@mails.tsinghua.edu.cn
DCST, Tsinghua University&
Quan Cheng Laboratory&
Zhongguancun Laboratory
Beijing, China

Yiding Liu
liuyiding.tanh@gmail.com
Baidu Inc.
Beijing, China

Qingyao Ai*
aiqy@tsinghua.edu.cn
DCST, Tsinghua University&
Quan Cheng Laboratory&
Zhongguancun Laboratory
Beijing, China

Haitao Li
liht22@mails.tsinghua.edu.cn
DCST, Tsinghua University&
Quan Cheng Laboratory&
Zhongguancun Laboratory
Beijing, China

Shuaiqiang Wang
shqiang.wang@gmail.com
Baidu Inc.
Beijing, China

Yiqun Liu
yiqunliu@tsinghua.edu.cn
DCST, Tsinghua University&
Quan Cheng Laboratory&
Zhongguancun Laboratory
Beijing, China

Dawei Yin
yindawei@acm.org
Baidu Inc.
Beijing, China

Shaoping Ma
msp@tsinghua.edu.cn
DCST, Tsinghua University&
Quan Cheng Laboratory&
Zhongguancun Laboratory
Beijing, China

ABSTRACT

Passage retrieval is a fundamental task in many information systems, such as web search and question answering, where both efficiency and effectiveness are critical concerns. In recent years, neural retrievers based on pre-trained language models (PLM), such as dual-encoders, have achieved huge success. Yet, studies have found that the performance of dual-encoders are often limited due to the neglecting of the interaction information between queries and candidate passages. Therefore, various interaction paradigms have been proposed to improve the performance of vanilla dual-encoders. Particularly, recent state-of-the-art methods often introduce late-interaction during the model inference process. However, such late-interaction based methods usually bring extensive computation and storage cost on large corpus. Despite their effectiveness, the concern of efficiency and space footprint is still an important factor that limits the application of interaction-based neural retrieval models. To tackle this issue, we **I**ncorporate **I**mplicit **I**nteraction into dual-encoders, and propose $\mathbf{I}^3$ retriever. In particular, our implicit interaction paradigm leverages generated pseudo-queries to simulate query-passage interaction, which jointly optimizes with query and passage encoders in an end-to-end manner. It can be fully pre-computed and cached, and its inference process only involves simple dot product operation of the query vector and passage vector, which makes it as efficient as the vanilla dual encoders. We conduct comprehensive experiments on MSMARCO and TREC2019 Deep Learning Datasets, demonstrating the I^3 retriever's superiority in terms of both effectiveness and efficiency. Moreover, the proposed implicit interaction is compatible with special pre-training and knowledge distillation for passage retrieval, which brings a new state-of-the-art performance. The codes are available at https://github.com/Deriq-Qian-Dong/III-Retriever.

CCS CONCEPTS

• **Information systems** → **Language models**; **Learning to rank**; **Similarity measures**; *Novelty in information retrieval.*

KEYWORDS

Learning to Rank; Language models; Semantic Matching

1 INTRODUCTION

Passage retrieval is fundamental in modern information retrieval (IR) systems, typically serving as a preceding stage of reranking. The aim of passage retrieval is to find relevant passages from a large corpus for a given query, which is crucial to the final ranking performance [3, 24, 26, 62, 68]. Conventional methods for passage retrieval (e.g., BM25 [50]) usually consider lexical matching between the terms of query and passage. In recent years, neural retrievers based on pre-trained language models (PLMs) have prospered and achieved the state-of-the-art performance.

*Corresponding author.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10.
https://doi.org/10.1145/3583780.3614923

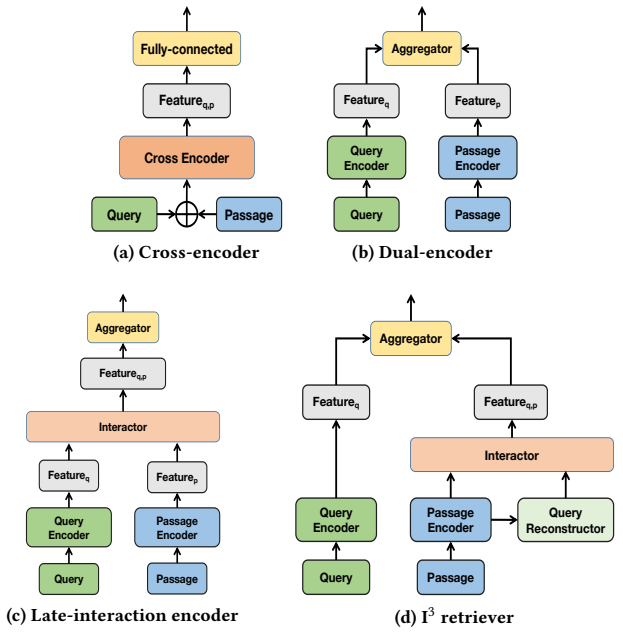

Figure 1: Illustration of three conventional PLM-based IR models: (a) cross-encoder, (b) dual-encoder, (c) late-interaction encoder, and our proposed (d) I^3 retriever.

In particular, existing PLM-based IR models can be broadly categorized into cross-encoders [40], dual-encoders [24] and late-interaction encoders [16, 25], as shown in Figures 1(a), 1(b) and 1(c), respectively. Without considering the fine-grained interactions between the tokens of query and passage, the major merit of dual-encoders is their high efficiency in inference. Yet, their effectiveness is usually considered sub-optimal compared with cross-encoders or other interaction-based models. Cross-encoders take the concatenation of query and passage as input to perform full interaction that effectively captures relevance features. As query-passage interactions are important factors in relevance modeling [17], cross-encoders usually have superior ranking performance. However, their applications are limited to small collections (e.g., the top passages retrieved by dual-encoders) due to their high inference latency. To combine the merits of both methods, late-interaction encoders adopt separate query/passage encoding and apply lightweight interaction schemes (i.e., late interactions) between the vectors of query and passage. They are usually more effective than dual-encoders for passage retrieval and less computationally expensive than cross-encoders.

Despite their effectiveness, late-interaction models are still sub-optimal for passage retrieval on large corpus, mainly due to two problems. First, effective late-interaction models usually relies on token-level representations of passages to allow subsequent token-level interactions [25, 52], where the storage cost of such multi-vector passage representation is enormous. Second, compared to dual-encoders, which adopts simple dot-product operation between single-vector representations of queries and passages, late-interaction models still requires extra computation for each query-passage pair. Such cost could be magnified by the scale of massive corpus and eventually cause unacceptable efficiency degeneration [28]. As such, existing late-interaction methods can hardly be applied to real-world scenarios that require low inference latency and storage cost.

To address these limitations and explore a better solution w.r.t. effectiveness and efficiency for passage retrieval, we propose a novel yet practical paradigm that **I**ncorporates **I**mplicit **I**nteraction (I^3) in dual-encoders. Unlike existing interaction schemes that requires explicit query text as input, the implicit interaction is conducted between a passage and the pseudo-query vectors generated from the passage. Note that the generated pseudo-query vectors are implicit (i.e., latent) without explicit textual interpretation. Such implicit interaction paradigm is appealing, as 1) it is fully decoupled from actual query, and thus allows high online efficiency with offline caching of passage vectors, and 2) compared with using an off-the-shelf generative model [41] to explicitly generate textual pseudo-query, our pseudo-query is represented by latent vectors that are jointly optimized with the dual-encoder backbone, which is more expressive for the downstream retrieval task.

To conduct implicit interaction, we propose a novel model architecture as shown in Figure 1(d). It advances vanilla dual-encoders with two auxiliary modules. First, we introduce a lightweight generative module namely **query reconstructor** to generate pseudo-query vectors for a given passage. Next, we apply a **query-passage interactor** that takes the concatenation of the passage vectors and the generated pseudo-query vectors as input, to perform implicit interaction. The interactor outputs query-aware passage vectors for each passage, which can be pre-computed and cached before deploying the model for online inference. The final query-passage relevance scores can be computed with simple dot-product operation, which gives our model the same high efficiency and low storage cost as dual-encoders. The superior balance between effectiveness and efficiency makes our model more attractive in real-world applications. We summarize our main contributions as follows:

- We propose a novel PLM-based retrieval model, namely I^3 retriever, which incorporates implicit interaction in dual-encoders.
- We introduce two modules in I^3 retriever that are jointly trained with query and passage encoders in an end-to-end manner, i.e., query reconstructor and query-passage interactor. The query reconstructor is able to generate pseudo-queries for the query-passage interactor, which subsequently encodes query-aware information in the final passage vectors.
- We conduct comprehensive evaluation on large scale datasets. The results show that I^3 is able to achieve superior performance w.r.t both effectiveness and efficiency for passage retrieval. We also conduct a thorough study to clarify the effects of implicit interaction.

2 RELATED WORK

In this section, we briefly review some existing studies with respect to three topics, i.e., traditional neural IR models, PLM-based IR models and query generation for IR.

2.1 Conventional Neural IR Models

Modern information retrieval systems usually adopt the two-stage paradigm, i.e. retrieval-then-reranking. Neural IR models can be

categorized as either retrievers or rerankers based on their served stage. Retrievers can pre-compute the vector representation of passages in corpus and thus perform efficient retrieval via approximate nearest neighbor algorithms. Therefore, retrievers usually define sophisticated representation learning module. DSSM [21] is a representative neural retriever, which uses the fully-connected network for representation learning. Besides, convolutional networks [12, 20, 45, 53] and recurrent networks [44, 56] are also widely used for representation learning in neural retrievers. On the other hand, rerankers effectively capture relevance features through sufficient interactions. The interaction module plays a vital role in the effectiveness of rerankers. DRMM [17] designs a matching histogram mapping to model the interaction between terms of query and passage. Conv-KNRM [7] and Arc-II [20] use convolutional networks as the interaction module. However, the computational overhead of rerankers during inference is significantly higher than that of retrievers, and thus rerankers only serve a small set of candidates at the final stage.

2.2 PLM-based IR models

PLM-based retriever. PLM-based retrievers usually compute low dimensional representations for the query and passage using the encoder of pre-trained transformer [55], such as BERT [9] and RoBERTa [31]. DPR [24] is the first to leverage PLM for the task of semantic retrieval, while extensive methods are subsequently proposed to improve the effectiveness. In particular, most of the existing PLM-based retrievers improve the model performance from the following aspects. (1) ***By introducing late-interactions after encoding:*** ColBERT [25], COIL [16] and ME-BERT [35] are three representative studies that explicitly model the interactions after query/passage encodings. The performance is largely boosted by the interactions compared with DPR (i.e., vanilla dual-encoder), while the late interaction also brings significant computational overhead. (2) ***By designing effective fine-tuning processes:*** For example, ANCE [63] proposes to a hard negative sampling technique that greatly improve the effectiveness. Moreover, RocketQAv1 [46] and RocketQAv2 [49] boost the performance of dense retrieval models by leveraging the power of cross-encoder. The relevance features captured through sufficient interaction by the cross-encoder could be properly transferred to retrievers in a cascade or joint training manner. ERNIE-Search [34] narrows the divide between cross-encoder and dual-encoder models through on-the-fly distillation in the process of fine-tuning. ColBERTv2 [52] further improves ColBERT by employing fine-tuning with distillation. (3) ***By designing pre-training tasks tailored for retrieval:*** A handful of studies focused on constructing pseudo-training data for retrieval-oriented pre-training, such as ICT [2], COSTA [37], DCE [29], etc. Besides, several studies [27, 32, 33, 58–60, 60, 67] employ weak generative modules (i.e. decoder) to enhance the query/passage encoding through pre-training. Notably, after pre-training, the weak decoder is discarded and only the enhanced encoder is employed as the backbone of retriever. In this work, we propose a novel approach that incorporates implicit interaction modeling into the dual-encoder architecture by introducing a generative module. To the best of our knowledge, this is the first attempt to introduce a generative module as a backbone in a retriever.

PLM-based reranker. PLM-based rerankers usually take the concatenated query and passage as input and perform full interaction between query and passage via self-attention [11, 13, 18]. In particular, monoBERT [40] is the first work that re-purpose BERT as a reranker. duoBERT [42] integrates monoBERT in a multistage ranking pipeline and further adopts a pairwise classification framework for the final re-ranking. UED [64] utilizes a unified encoder-decoder framework to jointly optimize passage reranking and query generation tasks, demonstrating that these two tasks could facilitate each other. KERM [10] leverages external knowledge graph to more accurately model the interaction between query and passage, and thus achieves the state-of-the-art results. Inspired by the superior performance of PLM-based reranker, our method is equipped with a cross-interaction module that allows effective implicit interaction during passage encoding.

2.3 Query Generation for IR

The technique of query generation has been widely adopted in a variety of IR applications. For example, a well-known query generation method, namely doc2query [43], proposes a sequence-to-sequence model trained on relevant query-passage pairs to generate multiple queries for each passage. These generated queries can be considered as a passage expansion for the downstream retrieval task. This approach is effective in mitigating the issue of term mismatch between queries and passages. Moreover, docT5query [41] employs T5 [48] to generate queries and delivers an improved performance over doc2query. More recently, the application of query generation has been examined in the context of pre-training dense retrievers [59], data augmentation [1, 29, 30] and domain adaptation [8, 36, 57, 61]. However, these studies leverage query generation models as an off-the-shelf tool, which might not be the optimal for the downstream retrieval task. In our study, we introduce a lightweight generative module, i.e., the query reconstructor, which is jointly trained with the retrieval backbone in an end-to-end manner. By doing this, the query reconstructor is learned to generate pseudo-queries that are more helpful for the final retrieval task.

3 PRELIMINARIES

In this section, we introduce the problem definition of passage retrieval, and present several PLM-based IR methods.

3.1 Problem Definition

Modern IR systems usually follow a retrieve-then-rerank pipeline. Given a corpus of passages $\mathcal{G} = \{\mathbf{p}_i\}_{i=1}^{G}$, the aim of **retrieval** is to find a small set of candidate passages (i.e., $\mathcal{K} = \{\mathbf{p}_j^{\mathbf{q}}\}_{j=1}^{K}$) and $K \ll G$) that is relevant to a specific query $\mathbf{q}$. In particular, a passage $\mathbf{p}$ is a sequence of words $\mathbf{p} = \{w_p\}_{p=1}^{|\mathbf{p}|}$, where $|\mathbf{p}|$ denotes the length of $\mathbf{p}$. Similarly, a query is a sequence of words $\mathbf{q} = \{w_q\}_{q=1}^{|\mathbf{q}|}$. After the retrieval stage, reranking is conducted to finalize a better permutation on $\mathcal{K}$, where more relevant passages are ranked higher.

It worth noting that retrieval and reranking models usually have different practical concerns. In particular, both efficiency and effectiveness are vital for retrieval models, as real-world scenarios usually require fast retrieval on large scale corpus. On the other hand, reranking models are more concentrated on effectiveness,

and they should be able to effectively capture the subtle differences between relevant passages. In this work, our attention is focused on the PLM-based retriever, and we propose a implicit interaction paradigm that achieves the state-of-the-art performance in terms of both effectiveness and efficiency for passage retrieval.

3.2 PLM-based Retriever and Reranker

The performance of neural IR models, including retrievers and rerankers, have been significantly boosted by pre-trained language models (PLM), where various ways of leveraging PLM for IR are proposed. As illustrated in Figure 1, PLM-based IR models can be categorized into three types, i.e., dual-encoders, late-interaction encoders and cross-encoders, in terms of the interaction mechanism applied between query and passage. Overall, existing studies indicate that incorporating more interactions between queries and passages in a PLM-based IR method can improve relevance modeling, but it also comes at the cost of extra computational overhead. In the following, we further introduce the detailed structures of these models.

Dual-encoder. Dual-encoders employ two PLM-based encoders to respectively encode the query and passage in a latent embedding space. The relevance score $S(\mathbf{q}, \mathbf{p})$ between query and passage is formulated as

$$S(\mathbf{q}, \mathbf{p}) = \text{Aggregate}\left(\mathbb{E}_q(\mathbf{q})_{[CLS]}, \mathbb{E}_p(\mathbf{p})_{[CLS]}\right). \tag{1}$$

Here, Aggregate$(\cdot)$ is usually implemented as a simple metric (e.g., dot-product) between query and passage vectors, which is computed by query and passage encoders (i.e., $\mathbb{E}_q$ and $\mathbb{E}_p$), respectively. The encoders are stacked transformer layers, where we fetch the representation of [CLS] token in the last layer as final query/passage vector.

The major merit of dual-encoders lies in its high efficiency. As the query and passage are decoupled at encoding, the passages in large corpus $\mathcal{G}$ can be pre-computed and cached offline. By doing this, substantial computational resources could be saved during the online inference for fast retrieval. However, the limitation is also apparent. The absence of interaction between the query and passage during their encoding leads to an inability to effectively capture complex relevance [22, 25, 65].

Cross-encoder. Cross-encoders are considered the most effective PLM-based IR method due to their early incorporation of query-passage interactions. It takes the concatenation of query and passage as input, and computes the relevance score as

$$S(\mathbf{q}, \mathbf{p}) = \text{FC}\left(\mathbb{E}_{q,p}(\mathbf{q} \oplus \mathbf{p})_{[CLS]}\right), \tag{2}$$

where $\oplus$ means the concatenation operation and $\mathbb{E}_{q,p}$ is the PLM encoder. The FC is a fully-connected layer that transforms the [CLS] representation to a relevance score.

Cross-encoders allow full token-level interactions between query and passage via self-attention [9], where relevance features could be adequately captured. This leads to a superior performance in relevance modeling compared with other PLM-based IR models. However, compared with dual-encoders, cross-encoders require extensive online computation, where no intermediate representations could be pre-computed and cached offline. The low efficiency of cross-encoders limits its application for retrieval on large scale corpus, and thus they are mainly designed for reranking stage.

Late-interaction encoder. To balance efficiency and effectiveness, the late-interaction paradigm introduces interaction between query and passage after encoding, which can be formulated as

$$S(\mathbf{q}, \mathbf{p}) = \text{Aggregate}\left(\text{Interact}\left(\mathbb{E}_q(\mathbf{q}), \mathbb{E}_p(\mathbf{p})\right)\right). \tag{3}$$

The Aggregate$(\cdot)$ operation aggregates the relevance features captured from the Interact$(\cdot)$ into a relevance score $S(\mathbf{q}, \mathbf{p})$.

ColBERT [25] is a representative of late-interaction method. Its interaction is implemented as the maximum similarity score between each pair of token representations of query and passage in the final layers. Then, these scores are aggregated into a final relevance score, which can be formulated as

$$S(\mathbf{q}, \mathbf{p}) = \sum_{q=1}^{|\mathbf{q}|} \max_{p=1}^{|\mathbf{p}|} \left(\mathbb{E}_q(\mathbf{q})_{w_q} \cdot \mathbb{E}_p(\mathbf{p})_{w_p}\right). \tag{4}$$

To reduce the computational overhead of ColBERT, COIL [16] restricts the interactions to occur solely between pairs of query and passage tokens that have an exact match. More details about these methods can be found in their original papers [16, 25].

Similar to dual-encoders, late-interaction encoders also decouple the encoding of query and passage, and thus allow pre-computation of all passage vectors in corpus $\mathcal{G}$. However, the late interactions still create considerable computational overhead for each query-passage pair. Worse still, they further cost enormous space footprint for caching multi-vector passage vectors, where dual-encoders only need to store single-vector passage vectors.

Remarks. Overall, former experience tells us that effective interaction usually cost extra computation or storage, where most of the existing studies are proposed to make amends. However, we intend to investigate a different research question: **Can we model query-passage interaction without any efficiency degeneration?** Noting that the dual-encoders are efficient due to its offline pre-computation of passage vectors, the key to answer this question is how to pre-compute and model query-passage interactions offline. However, this is challenging because the actual queries issued by users are agnostic during the pre-computation, while we can only access to the passages in the corpus. In the next section, we propose a novel method, namely I^3 retriever, which tackles this challenge to achieve high effectiveness without hurting efficiency.

4 METHOD

In this section, we present I^3 retriever, an effective approach that incorporates implicit interaction in dual-encoder. We first introduce the overall architecture, which includes query and passage encoders, query reconstructor and query-passage interactor. Then, we present the details of implicit interaction, and the end-to-end optimization and inference of I^3 retriever.

4.1 Overall Architecture

Figure 2 illustrates the overall architecture of the I^3 retriever. In particular, we advance the passage encoder of vanilla dual-encoders with two auxiliary modules, i.e., query reconstructor and query-passage interactor. Overall, the workflow of I^3 can be formulated as follows:

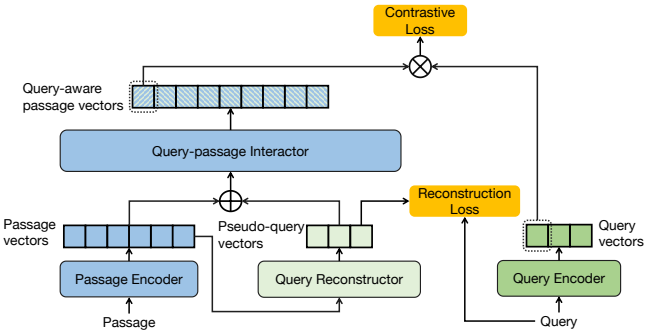

Figure 2: The architecture of I^3 retriever.

- **Query/Passage encoding**. The query and passage encoders (i.e., vanilla dual-encoders) are the backbone of our proposed method. They first encode the tokens of query and passage into latent vectors.
- **Query reconstruction**. Inspired by generative models [47], we introduce a lightweight query reconstructor to generate a pseudo-query for each passage, which can be viewed as a potential query for a specific passage.
- **Query-passage interaction**. We apply a query-passage interactor to conduct cross-encoder-alike interaction between each passage and its pseudo-query. It finalizes a *query-aware passage vector*, which learns to encode passage information that are vital to its potential query.
- **Relevance computation**. The final relevance score is computed as the dot-product between the query vector produced by query encoder, and the query-aware passage vector produced by the query-passage interactor. The simple relevance metric allows high efficiency for online retrieval.

We refer to such interaction over vanilla dual-encoders as **implicit interaction**, since it solely relies on generated pseudo-query vectors, rather than textual query terms. Note that the inference of implicit interaction is conducted on the passage side, and thus it could be pre-computed and cached to enable efficient online retrieval. Next, we focus on the two auxiliary modules and elaborate how they are incorporated to conduct implicit interaction.

4.2 Incorporating Implicit Interaction

Query reconstructor. The query reconstructor is a generative model with stacked transformer layers, which can be viewed as a decoder module for passage encoder. In particular, it takes a set of trainable embedding $\mathbf{I}^0 \in \mathbb{R}^{\overline{q} \times d_{model}}$ as input vectors, and conduct cross-attention with the output vectors of passage encoding. For simplicity, we use the special token, [MASK], as the initial parameters of $\mathbf{I}^0$. Here, $\overline{q}$ is the length of generated queries and d_{model} is the dimension of the embeddings. In each layer $n = 1, ..., N$, the output vectors $\mathbf{I}^n$ are computed as

$$\mathcal{A}_{(\mathbf{I}^{n-1},p)} = \text{softmax}(\frac{(\mathbf{W}_n^Q \mathbf{I}^{n-1})(\mathbf{W}_n^K \mathbb{E}_p(\mathbf{p}))^{\mathrm{T}}}{\sqrt{d_{model}}}), \tag{5}$$

$$\mathbf{I}^n = \sum_{p=1}^{|\mathbf{p}|} \mathcal{A}_{(\mathbf{I}^{n-1},p)} \mathbf{W}_n^V \mathbb{E}_p(\mathbf{p}), \tag{6}$$

where $\mathcal{A}_{(\mathbf{I}^{n-1},p)}$ is the cross-attention between $\mathbf{I}^{n-1}$ and the passage vectors $\mathbb{E}_p(\mathbf{p})$, and $\mathbf{W}_n^*$ are the parameters of query reconstructor. The reconstructed pseudo-query vectors for passage $\mathbf{p}$ is denoted as $\mathbb{K}_q(\mathbf{p}) \coloneqq \mathbf{I}^N$. Notably, the input embedding $\mathbf{I}^0$ is the same for all passages. By doing this, we can reconstruct pseudo-query vectors $\mathbb{K}_q(\mathbf{p})$ from passage vectors in a query agnostic manner.

It worth mentioning that the query reconstructor differs from existing generative language models from two perspectives: 1) Unlike conventional auto-regressive models that generate tokens sequentially, our query reconstructor generates all the $\overline{q}$ vectors in parallel, and thus is more efficient; 2) Our model generates latent vectors rather than actual words to represent the pseudo-query, which is more expressive to represent semantic information for downstream retrieval task.

Query-passage interactor. The interactor $\mathbb{E}_{q,p}(\cdot)$ has a cross-encoder-alike structure that stacks multiple transformer layers. It conducts full cross-interaction between passage vectors $\mathbb{E}_p(\mathbf{p})$ and its reconstructed pseudo-query vectors $\mathbb{K}_q(\mathbf{p})$, i.e., implicit interaction. The interactor refines the passage vectors and outputs query-aware passage vectors. Intuitively, the interactor leverages the pseudo-query to encode important knowledge in the query-aware passage vector that might be relevant to real queries. More formally, the $\mathbb{T}_p(\mathbf{p})$ are computed as

$$\mathbb{T}_p(\mathbf{p}) = \mathbb{E}_{q,p}\left(\mathbb{K}_q(\mathbf{p}) \oplus \mathbb{E}_p(\mathbf{p})\right), \tag{7}$$

where $\oplus$ means the concatenation operation. Finally, we can advance vanilla dual-encoders by rewriting the relevance score $S(\mathbf{q}, \mathbf{p})$ in Eq. 1 as

$$S(\mathbf{q}, \mathbf{p}) = \mathbb{E}_q(\mathbf{q})_{[CLS]} \cdot \mathbb{T}_p(\mathbf{p})_{[CLS]}. \tag{8}$$

By introducing query reconstructor and query-passage interactor in passage encoding, our I^3 retriever is effective and efficient, as 1) it effectively incorporates implicit interaction that encodes vital passage information w.r.t. potential queries, and 2) the implicit interaction is conducted on the passage side in a query agnostic manner, which brings high online inference efficiency that is on par with the vanilla dual-encoder.

4.3 Model Optimization

Retrieval loss. Following previous work [15], our I^3 retriever is optimized by the following contrastive loss

$$\mathcal{L}_c = -\log \frac{\exp\left(S\left(\mathbf{q}, \mathbf{p}_+\right)\right)}{\exp\left(S\left(\mathbf{q}, \mathbf{p}_+\right)\right) + \sum\limits_{\mathbf{p}_- \in \mathcal{N}_-} \exp\left(S\left(\mathbf{q}, \mathbf{p}_-\right)\right)}, \tag{9}$$

where $\mathcal{N}_-$ is a set of hard negative passages (denoted as $\mathbf{p}_-$) for query $\mathbf{q}$. As illustrated in Figure 3, the fine-tuning process consists of two stages, where the optimized models are called retriever 1 and retriever 2, respectively. During the training of retriever 1, the negative samples $\mathcal{N}_-$ are BM25 hard negatives. During the training of retriever 2, hard negatives are also mined using the optimized retriever 1 to complement the negative pool $\mathcal{N}_-$.

Reconstruction loss. In addition to the retrieval loss (i.e., Eq. 9), we also introduce an auxiliary reconstruction loss to guide the query reconstructor, which is defined as

$$\mathcal{L}_r = -\sum_{w_i \in \mathbf{q}} \mathbf{y}_{w_i} \log\left(\mathbf{W}^R \mathbb{K}_q(\mathbf{p})_q\right), \tag{10}$$

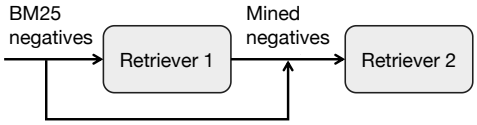

Figure 3: Illustration of our fine-tuning pipeline.

Table 1: Statistics of MSMARCO-DEV and TREC DL 19.

	MSMARCO-DEV	TREC DL 19
#Queries	6980	43
#Rel.Psgs.	7437	4102
Rel.Psgs./Query	1.1	95.4
#Graded.Labels	2	4

where w_i is the i-th word of a pseudo query $\mathbf{q}$ and y_{w_i} indicates the word id of w_i in vocabulary. The pseudo query could be generated by RACE [51] or other keyword extraction methods, such as large language models. $\mathbf{W}^R$ is the parameter of reconstructor, mapping the dimension of $\mathbb{K}_q(\mathbf{p})_q$ from d_{model} to vocabulary size.

The final training loss of I^3 retriever is the combination of the above-mentioned two losses as

$$\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_r, \tag{11}$$

where λ is a hyper-parameter. All the modules are jointly optimized with this loss in an end-to-end manner.

4.4 Model Inference

Offline pre-computation. The computation of query-aware passage vectors $\mathbb{T}_p(\mathbf{p})$ is totally decoupled with online inference w.r.t. a specific query. Therefore, all the passage vectors in the corpus $\mathcal{G}$ could be pre-computed and stored. Note that the pre-computed passage vectors are interacted with pseudo-query vectors generated by the query reconstructor, and thus are more expressive than the passage vectors produced by vanilla dual-encoders. Besides, we adopt single-vector representation for each passage, which avoids massive storage cost.

Online inference. The online inference process is identical to vanilla dual-encoders, and thus our method has the same high efficiency. When a query is received, I^3 applies the query encoder to compute its vector $\mathbb{E}_q(\mathbf{q})_{[CLS]}$. Next, it conducts maximum inner product search (MIPS) over the offline-cached query-aware passage vectors $\{\mathbb{T}_p(\mathbf{p}_i)_{[CLS]}\}_{i=1}^{G}$ to retrieve a set of relevant passages.

Analysis. For the online inference stage, the time complexity of I^3 retriever is $\text{O}(E+G)$, where E and G are the cost of query encoding and MIPS operation over the corpus $\mathcal{G}$, respectively. For late-interaction encoder with multi-vector representations, their time complexity is $\text{O}(E + Q \times P \times G)$, where Q and P indicate the number of vectors representing query and passage, respectively. Moreover, our single-vector representation could be more easily supported by commonly-used indexing techniques [23, 38] Therefore, our method has superior efficiency compared with existing late-interaction encoders.

5 EXPERIMENTAL SETUP

5.1 Datasets

We use MSMARCO-Passage [39] as the large-scale corpus for our experiments. It consists of around 8.8 million passages. Following previous work [16, 24, 25, 63], we train our model on MSMARCO-TRAIN query set including 502,939 queries, and evaluated on two widely used query sets, i.e., MSMARCO-DEV and TREC DL 19. **MSMARCO-DEV** [39] includes 6,980 sparsely-judged queries, each of which has 1.1 relevant passages on average. **TREC DL 19** [5] contains 43 densely-judged queries, which are annotated with fine-grained relevance labels, i.e., irrelevant, relevant, highly relevant and perfectly relevant. Such data can be used to evaluate fine-grained ranking performance. Tabel 1 summarizes the detailed information of the two query sets.

5.2 Baselines

We include the following variants of our methods to ensure a fair comparison with baselines:

- I^3 **retriever**$_1$ is our proposed method that incorporates implicit interaction in dense retrieval.
- I^3 **retriever**$_2$ is an improved version of I^3 retriever$_1$, which further leverages the widely-used negative sampling technique [63].
- I^3 **retriever**$_3$ is initialized from RetroMAE [32] and fine-tuned with hard negatives, following the baselines.
- I^3 **retriever**$_4$ is also initialized from RetroMAE [32] and further distilled using a cross-encoder with the Kullback–Leibler divergence loss function.

I^3 **retriever**$_1$ and I^3 **retriever**$_2$ are compared with dense methods without special pre-training and distillation. We include two sparse retrievers, i.e., BM25 [50] and DeepCT [6], as baselines. We include more dense retrievers, which can be categorized as non-interaction, late-interaction, and early-interaction methods. 1) **Non-interaction methods:** DPR [24] and ANCE [63] are two widely used baselines that do not consider any form of query-passage interaction; 2) **Late-interaction methods:** ME-BERT [35], COIL [16] and ColBER [25] apply lightweight interaction after query/passage encoding; 3) **Early-interaction methods:** DRPQ [54] and DCE [29] model the interaction during the encoding stage. Notably, both DCE and our proposed I^3 retriever conduct interaction between pseudo-query and passage during passage encoding. The key difference lies in that DCE employs docT5Query [41] to explicitly generate pseudo-queries, while I^3 retriever utilizes a lightweight reconstruction module to implicitly reconstruct pseudo-query vectors in an end-to-end manner.

I^3 **retriever**$_3$ and I^3 **retriever**$_4$ are compared with baselines with special pre-training and distillation, respectively. For dense retrieval models with task-specific pre-training, we include the following methods: coCondenser [15] continues to pre-trained on the target corpus with contrastive loss. Other pre-trained methods, such as SimLM [58], Cot-MAE [60] and RetroMAE [32], employ a bottleneck architecture that learns to compress the passage information into a vector through pre-training. We also include the state-of-the-art methods that facilitate dense retrieval with knowledge distillation: TAS-B [19], RocketQAv2 [49] and ERNIE-Search [34]

primarily concentrate on distilling knowledge from a cross-encoder to a single vector retriever. On the other hand, SPLADEv2 [14] and ColBERTv2 [52] focus on distilling knowledge from a cross-encoder to a multi-vector retriever. All baselines with special pre-training or distillation can be categorized as non-interaction retrievers, except for SPLADEv2 [14] and ColBERTv2 [52].

5.3 Implementation Details

For training I^3, we use the Lamb optimizer [66] with a learning rate of 2e-5. The model is trained with a batch size of 16. The ratio of positive and hard negatives is set to 1:127 in the contrastive loss (i.e., Eq. 9). Besides, the hyper-parameter λ in Eq. 11 is decayed with epochs exponentially, starting from an initial value of 1.

For the comparison with dense methods without distillation or special pre-training, all the baselines are initialized with $\text{BERT}_{\text{base}}$ model, except ANCE [63], which utilizes $\text{RoBERTa}_{\text{base}}$. In our model, we set the number of layers of query encoder, passage encoder, query reconstructor and query-passage interactor as 6, 6, 3 and 3, respectively. We configure the length of generated query $\overline{q}$ as 32 to cover the majority of queries in the training data. As such, I^3 retriever has a comparable model size with the baselines on the passage side (i.e., 6+3+3 transformer layers), but fewer parameters on the query side. The query and passage encoders are initialized with $\text{BERT}_{\text{distill}}$. For the comparison with distillation or special pre-training, we directly use the RetroMAE [32] to initialize the backbone of I^3, as pre-training is not the main focus of this paper. To minimize the number of parameters introduced, we configure the query reconstructor and query-passage interactor to consist of a single layer. The query reconstructor and query-passage interactor are random initialized. Prior to fine-tuning, the query reconstructor and query-passage interactor undergoes optimization for 20K steps on passage collection $\mathcal{G}$ via Eq. 11 with $\lambda = 1$ while keeping the parameters of backbone frozen. The pseudo query, generated by a language model Flan-T5-XL [4] in a zero shot setting, along with its corresponding passage, is regarded as a positive pair.

Our proposed model is implemented with PyTorch and Huggingface [1]. All the training and evaluation are conducted on 8 NVIDIA Tesla A100 GPUs (with 40G RAM).

6 EXPERIMENTAL RESULTS

In this section, we present the experimental results and conduct thorough analysis of I^3 to clarify its advantages.

6.1 Overall Comparison

Effectiveness. We first compare the effectiveness of I^3 with all the baselines. The results are shown in Table 2, where the detailed setting of each method is also included, i.e., whether a method employs single vector passage representation, negative mining or a particular interaction scheme. Notably, the baselines are categorized into three groups: methods without special pre-training or distillation, methods with special pre-training, and methods with distillation. We report MRR@10 and Recall@100 on MARCO DEV Passage, and NDCG@10 on TREC DL 19.

First, we can draw several key findings from the first group (i.e., methods without pre-training or distillation):

- I^3 retriever_1 outperforms DPR by a large margin, while maintaining the same inference speed. This proves that the implicit interaction is beneficial for encoding relevance information in the final passage representation.
- Among the PLM-based baselines, COIL and ColBERT significantly surpass other methods. This is because COIL and ColBERT apply effective late interaction between the multi-vector representations of actual query and passage. However, such effectiveness costs extensive computation and storage (i.e., caching multiple vectors for each passage). Compared with COIL and ColBERT, our I^3 retriever_1 method is more efficient, and can achieve comparable performance w.r.t. Recall@1000 on MARCO DEV Passage, and better performance w.r.t. NDCG@10 on TREC DL 19.
- I^3 retriever_1 is significantly better than DCE. Note that DCE also introduces interaction between pseudo-query and passage during passage encoding, where the pseudo-query is drawn from an off-the-shelf docT5query model [41]. As such, we can conclude that the superiority of I^3 retriever_1 can be attributed to the joint optimization of pseudo-query reconstruction and retrieval, which makes the implicit interaction more aligned with the downstream retrieval task.
- I^3 retriever_1 shows more significant improvement on TREC DL 19 than on MARCO-DEV. In particular, I^3 retriever_1 beats all the baseline methods on TREC DL 19, including COIL and ColBERT. This implies that our proposed implicit interaction can more accurately captures fine-grained relevance ranking than the baselines.

Next, we draw more findings from the second and third groups (i.e., methods with pre-training or distillation):

- I^3 can further improve those methods that leverage special pre-training or knowledge distillation, which shows that implicit interaction is compatible with these commonly-used techniques to achieve better results.
- By combining implicit interaction, pre-training and distillation, I^3 retriever_4 is able to achieve the state-of-the-art performance on both datasets and across all metrics.

Efficiency. Tabel 3 shows the efficiency comparison of I^3 and four representative models. We report the inference (i.e., relevance computation) time per query for 1,000, 100,000 and all (around 8.8 million) candidate passages as the key metrics. First, dual-encoders are without doubt the most efficient, as no query-passage interaction is involved. All the passage representations can be pre-computed and cached, which significantly saves the inference time. Second, late-interaction encoders, such as COIL and ColBERT, usually require extra computation to perform effective late interaction during inference. Third, our I^3 model is a promising solution that achieves remarkable performance on both effectiveness and efficiency. Unlike late-interaction that often undermines the inference efficiency, the implicit interaction introduced by I^3 can be pre-computed, and the final query-aware passage representation can be cached. This allows I^3 to be as efficient as vanilla dual-encoders.

6.2 Investigation on Implicit Interaction

In this section, we investigate how the implicit interaction affects the model performance on different passages. Specifically, it worth

[1] https://github.com/huggingface/transformers

Table 2: Performance comparison on MARCO-DEV and TREC DL 19.

Method	Settings			MARCO DEV Passage		TREC DL 19
	Single vector?	Mined-negatives	Interaction	MRR@10	Recall@1000	NDCG@10
BM25 (anserini)	-	-	-	.187	.857	.501
DeepCT	-	-	-	.243	.905	.551
Comparison with dense methods without pre-training or distillation						
DPR	✓		Non-interaction	.314	.953	.590
ANCE	✓	✓	Non-interaction	.330	.959	.648
DCE			Explicit-early	.338	-	-
DRPQ		✓	Explicit-early	.345	.964	-
ME-BERT		✓	Explicit-late	.334	-	.687
COIL			Explicit-late	.355	.963	.704
ColBERT			Explicit-late	.360	.968	.694
I^3 retriever$_1$	✓		Implicit-early	.349	.966	.720
I^3 retriever$_2$	✓	✓	Implicit-early	**.366**	**.976**	**.727**
Comparison with dense methods with special pre-training						
coCondenser	✓	✓	Non-interaction	.382	.984	.684
SimLM	✓	✓	Non-interaction	.391	.986	-
Cot-MAE	✓	✓	Non-interaction	.394	**.987**	-
RetroMAE	✓	✓	Non-interaction	.393	.985	-
I^3 retriever$_3$	✓	✓	Implicit-early	**.403**	**.987**	**.729**
Comparison with dense methods with distillation						
TAS-B	✓	✓	Non-interaction	.340	.975	.712
SPLADEv2		✓	Explicit-late	.368	.979	.729
RocketQAv2	✓	✓	Non-interaction	.388	.981	-
ColBERTv2		✓	Explicit-late	.397	.984	-
ERNIE-Search	✓	✓	Non-interaction	.401	.982	-
SimLM	✓	✓	Non-interaction	.411	.987	.714
Cot-MAE	✓	✓	Non-interaction	.404	.987	-
RetroMAE	✓	✓	Non-interaction	.416	**.988**	.681
I^3 retriever$_4$	✓	✓	Implicit-early	**.418**	**.988**	**.731**

Table 3: Query latency and storage cost.

Methods	# Candidates			Space(GiBs)
	1k	100k	8.8m	
Dual-encoder	**18ms**	**22ms**	**62.2ms**	**25.6**
COIL	41ms	69ms	344ms	110.8
ColBERT	50ms	83ms	430ms	154
Cross-encoder	2.4s	4.0m	5.9h	-
I^3 retriever	**18ms**	**22ms**	**62.2ms**	**25.6**

noting that some passages (namely **Type 1** passages) have relevant queries in the training data. On the other hand, there are many other passages (namely **Type 0** passages) that do not have relevant queries in the training data. In real-world scenarios, Type 1 passages might be those articles with abundant information that is desired by many queries, while Type 0 passages might be articles with specific information that can only be retrieved by a specific query. To investigate the performance of I^3 on the two types of passages, we divide the queries in MSMARCO DEV into two validation sets, namely **Set 0** and **Set 1**, where all the relevant passages in Set 0 are Type 0 passages, and all the relevant passages in Set 1 are Type 1 passages.

Table 4 shows the performance comparison on MSMARCO DEV and the two divided validatation sets. We compare I^3 retriever with our implementation of vanilla dual-encoder with the same negative sampling [63]. We also include cross-encoders in the comparison, where the results are obtained by directly reranking the candidates retrieved by I^3. We can see from the table that 1) I^3 can consistently outperform dual-encoders on both Set 0 and Set 1, which means that the implicit interaction is effective for both types of passages; 2) I^3 can achieve larger gain over dual-encoders on Set 1, and surprisingly outperform cross-encoders, which indicates that the implicit interaction is even more effective for Type 1 passages associated with multiple relevant training queries.

6.3 Case Study on Query Reconstruction

To better understand the implicit interaction incorporated in I^3, we demonstrate two cases in Table 5, and interpret their query reconstruction. Notably, the reconstructed query terms in Table 5 are decoded by $\mathbf{W}^R$ in Eq. 10 and are only used for the purpose of this case study. For each of the two passages, its query reconstruction is trained on one training query, and we presents the rankings of

Table 4: Performance on different groups of passages. The relative improvements are reported over dual-encoder.

Model	Overall		Set 0		Set 1	
	MRR@10	Imp.%	MRR@10	Imp.%	MRR@10	Imp.%
BM25	.187	-45.3	.192	-44.8	.072	-64.2
Dual-encoder	.342	-	.348	-	.201	-
Cross-encoder	.399	**16.7**	.407	**17.0**	.224	11.4
I^3 retriever$_2$	.366	7.0	.372	6.9	.245	**21.9**

Table 5: The cases of passage with multiple relevant queries. The blue texts represent those terms that are consistent with the topic of the training query, and the red texts represent those terms that are inconsistent with the topic of the training query.

Passage	Preheat the oven to 450 degrees F. Season salmon with salt and pepper. Place salmon, skin side down, on a non-stick baking sheet or in a non-stick pan with an oven-proof handle. Bake until salmon is cooked through, about 12 to 15 minutes.		
Relevant queries	Training query	best temperature to cook salmon	
	Testing queries	best oven temperature for baked salmon	Dual-encoder ranks the passage at #1 I^3 ranks the passage at #1
		how long to cook salmon cakes in oven	Dual-encoder ranks the passage at #7 I^3 ranks the passage at #1
Reconstructed query terms	how; long; what; salmon; minute; temperature; oven; bake; cook		
Passage	Tetanus, Diphtheria, Pertussis Vaccine for Adults tdap is a combination vaccine that protects against three potentially life-threatening bacterial diseases: tetanus, diphtheria, and pertussis (whooping cough). Td is a booster vaccine for tetanus and diphtheria. It does not protect against pertussis. Tetanus enters the body through a wound or cut.		
Relevant queries	Training query	what is a tdap immunization	
	Testing queries	what is the tdap vaccine	Dual-encoder ranks the passage at #1 I^3 ranks the passage at #1
		what is the tdap booster	Dual-encoder ranks the passage at #3 I^3 ranks the passage at #1
Reconstructed query terms	what; vaccine; immunity; bacterial; immune; booster; diseases		

I^3 and dual-encoder for two testing queries. Note that one of the testing query is similar to the training query, and the other one is dissimilar to the training query.

First, we find out that the reconstructed query terms can address several key concepts and terms that a query might ask for in a long passage. This indicates that our implicit interaction can help identifying important concepts during passage encoding and eventually boosting the final performance. This finding also justifies the results presented in Table 4, where the relative improvement of I^3 over dual-encoder is larger on Type 1 passages. Second, it worth noting that the reconstructed query terms are not just the memorization of training query. In fact, they are also generalized to the key terms that are not covered by the training query. For example, the first passage contains information about the temperature and the time of cooking salmon. We note that both two aspects are able to be covered by the reconstructed query terms, while the model is only trained on the training query that asks for temperature. As such, both dual-encoder and I^3 can perform well on the testing query that is similar to the training query (i.e., ranking the passage at #1), while I^3 performs much better on the testing query that is dissimilar to the training query. This concludes that the generalization ability of extracting key concepts of passages might be the key of the success of I^3.

7 CONCLUSION

In this paper, we propose a new interaction paradigm for dense retrieval, namely I^3 retriever, which incorporates implicit interaction into dual-encoders. Particularly, I^3 advances conventional dual-encoders with 1) a lightweight query reconstructor and 2) a query-passage interactor, which generate pseudo-query for expressive interaction. By doing this, our I^3 model is equipped with the capability of modeling implicit interaction, leading to an effective and efficient encoding of semantic relevance features in the final passage representations. The evaluation shows that the retrieval performance could be significantly improved without introducing extra computational overhead and space footprint. Besides, we also show that the proposed implicit interaction is compatible with special pretraining and distillation to achieve a better performance.

ACKNOWLEDGMENTS

This work is supported by Quan Cheng Laboratory (Grant No. QCLZD202301).

REFERENCES

[1] Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. InPars: Data Augmentation for Information Retrieval using Large Language Models. arXiv preprint arXiv:2202.05144 (2022).

[2] Wei-Cheng Chang, Felix X Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. arXiv preprint arXiv:2002.03932 (2020).

[3] Anfeng Cheng, Yiding Liu, Weibin Li, Qian Dong, Shuaiqiang Wang, Zhengjie Huang, Shikun Feng, Zhicong Cheng, and Dawei Yin. 2023. Layout-aware Webpage Quality Assessment. arXiv preprint arXiv:2301.12152 (2023).

[4] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. arXiv preprint arXiv:2210.11416 (2022).

[5] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. arXiv preprint arXiv:2003.07820 (2020).

[6] Zhuyun Dai and Jamie Callan. 2020. Context-aware term weighting for first stage passage retrieval. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 1533–1536.

[7] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In Proceedings of the eleventh ACM international conference on web search and data mining. 126–134.

[8] Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. arXiv preprint arXiv:2209.11755 (2022).

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).

[10] Qian Dong, Yiding Liu, Suqi Cheng, Shuaiqiang Wang, Zhicong Cheng, Shuzi Niu, and Dawei Yin. 2022. Incorporating Explicit Knowledge in Pre-trained Language Models for Passage Re-ranking. arXiv preprint arXiv:2204.11673 (2022).

[11] Qian Dong and Shuzi Niu. 2021. Latent Graph Recurrent Network for Document Ranking. In Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11–14, 2021, Proceedings, Part II 26. Springer, 88–103.

[12] Qian Dong and Shuzi Niu. 2021. Legal judgment prediction via relational learning. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 983–992.

[13] Qian Dong, Shuzi Niu, Tao Yuan, and Yucheng Li. 2022. Disentangled graph recurrent network for document ranking. Data Science and Engineering 7, 1 (2022), 30–43.

[14] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE v2: Sparse lexical and expansion model for information retrieval. arXiv preprint arXiv:2109.10086 (2021).

[15] Luyu Gao and Jamie Callan. 2021. Unsupervised corpus aware language model pre-training for dense passage retrieval. arXiv preprint arXiv:2108.05540 (2021).

[16] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. COIL: Revisit exact lexical match in information retrieval with contextualized inverted list. arXiv preprint arXiv:2104.07186 (2021).

[17] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In Proceedings of the 25th ACM international on conference on information and knowledge management. 55–64.

[18] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. 2020. A deep look into neural ranking models for information retrieval. Information Processing & Management 57, 6 (2020), 102067.

[19] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 113–122.

[20] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. Advances in neural information processing systems 27 (2014).

[21] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management. 2333–2338.

[22] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. arXiv preprint arXiv:1905.01969 (2019).

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

[24] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2004.04906 (2020).

[25] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 39–48.

[26] Canjia Li, Xiaoyang Wang, Dongdong Li, Yiding Liu, Yu Lu, Shuaiqiang Wang, Zhicong Cheng, Simiu Gu, and Dawei Yin. 2023. Pretrained Language Model based Web Search Ranking: From Relevance to Satisfaction. arXiv preprint arXiv:2306.01599 (2023).

[27] Haitao Li, Qingyao Ai, Jia Chen, Qian Dong, Yueyue Wu, Yiqun Liu, Chong Chen, and Qi Tian. 2023. SAILER: Structure-aware Pre-trained Language Model for Legal Case Retrieval. arXiv preprint arXiv:2304.11370 (2023).

[28] Haitao Li, Qingyao Ai, Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Zheng Liu, and Zhao Cao. 2023. Constructing Tree-based Index for Efficient and Effective Dense Retrieval. arXiv preprint arXiv:2304.11943 (2023).

[29] Zehan Li, Nan Yang, Liang Wang, and Furu Wei. 2022. Learning Diverse Document Representations with Deep Query Interactions for Dense Retrieval. arXiv preprint arXiv:2208.04232 (2022).

[30] Davis Liang, Peng Xu, Siamak Shakeri, Cicero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. arXiv preprint arXiv:2009.10270 (2020).

[31] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019).

[32] Zheng Liu and Yingxia Shao. 2022. Retromae: Pre-training retrieval-oriented transformers via masked auto-encoder. arXiv preprint arXiv:2205.12035 (2022).

[33] Shuqi Lu, Di He, Chenyan Xiong, Guolin Ke, Waleed Malik, Zhicheng Dou, Paul Bennett, Tie-Yan Liu, and Arnold Overwijk. 2021. Less is More: Pretrain a Strong Siamese Encoder for Dense Text Retrieval Using a Weak Decoder. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2780–2791.

[34] Yuxiang Lu, Yiding Liu, Jiaxiang Liu, Yunsheng Shi, Zhengjie Huang, Shikun Feng Yu Sun, Hao Tian, Hua Wu, Shuaiqiang Wang, Dawei Yin, et al. 2022. Ernie-search: Bridging cross-encoder with dual-encoder via self on-the-fly distillation for dense passage retrieval. arXiv preprint arXiv:2205.09153 (2022).

[35] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. Transactions of the Association for Computational Linguistics 9 (2021), 329–345.

[36] Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2020. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. arXiv preprint arXiv:2004.14503 (2020).

[37] Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, and Xueqi Cheng. 2022. Pre-train a Discriminative Text Encoder for Dense Retrieval via Contrastive Span Prediction. arXiv preprint arXiv:2204.10641 (2022).

[38] Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE transactions on pattern analysis and machine intelligence 42, 4 (2018), 824–836.

[39] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In CoCo@ NIPS.

[40] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. arXiv preprint arXiv:1901.04085 (2019).

[41] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. Online preprint 6 (2019).

[42] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with bert. arXiv preprint arXiv:1910.14424 (2019).

[43] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. arXiv preprint arXiv:1904.08375 (2019).

[44] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech, and Language Processing 24, 4 (2016), 694–707.

[45] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In Twenty-Fourth international joint conference on artificial intelligence.

[46] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 5835–5847.

[47] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[48] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. 21, 140 (2020), 1–67.

[49] Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. RocketQAv2: A Joint Training Method for Dense Passage Retrieval and Passage Re-ranking. arXiv preprint arXiv:2110.07367 (2021).

[50] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. Foundations and Trends® in Information Retrieval 3, 4 (2009), 333–389.

[51] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. Text mining: applications and theory (2010), 1–20.

[52] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 3715–3734.

[53] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In Proceedings of the 23rd ACM international conference on conference on information and knowledge management. 101–110.

[54] Hongyin Tang, Xingwu Sun, Beihong Jin, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. Improving document representations by generating pseudo query embeddings for dense retrieval. arXiv preprint arXiv:2105.03599 (2021).

[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems. 5998–6008.

[56] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A deep architecture for semantic matching with multiple positional sentence representations. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30.

[57] Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2021. Gpl: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. arXiv preprint arXiv:2112.07577 (2021).

[58] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Simlm: Pre-training with representation bottleneck for dense passage retrieval. arXiv preprint arXiv:2207.02578 (2022).

[59] Xing Wu, Guangyuan Ma, and Songlin Hu. 2022. Query-as-context Pre-training for Dense Passage Retrieval. arXiv preprint arXiv:2212.09598 (2022).

[60] Xing Wu, Guangyuan Ma, Meng Lin, Zijia Lin, Zhongyuan Wang, and Songlin Hu. 2022. Contextual mask auto-encoder for dense passage retrieval. arXiv preprint arXiv:2208.07670 (2022).

[61] Xuanji Xiao, Huaqiang Dai, Qian Dong, Shuzi Niu, Yuzhen Liu, and Pei Liu. 2023. Social4Rec: Distilling User Preference from Social Graph for Video Recommendation in Tencent. arXiv preprint arXiv:2302.09971 (2023).

[62] Xiaohui Xie, Qian Dong, Bingning Wang, Feiyang Lv, Ting Yao, Weinan Gan, Zhijing Wu, Xiangsheng Li, Haitao Li, Yiqun Liu, et al. 2023. T2Ranking: A large-scale Chinese Benchmark for Passage Ranking. arXiv preprint arXiv:2304.03679 (2023).

[63] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020).

[64] Ming Yan, Chenliang Li, Bin Bi, Wei Wang, and Songfang Huang. 2021. A Unified Pretraining Framework for Passage Ranking and Expansion. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35. 4555–4563.

[65] Wenwen Ye, Yiding Liu, Lixin Zou, Hengyi Cai, Suqi Cheng, Shuaiqiang Wang, and Dawei Yin. 2022. Fast Semantic Matching via Flexible Contextualized Interaction. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. 1275–1283.

[66] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. arXiv preprint arXiv:1904.00962 (2019).

[67] Kun Zhou, Xiao Liu, Yeyun Gong, Wayne Xin Zhao, Daxin Jiang, Nan Duan, and Ji-Rong Wen. 2022. MASTER: Multi-task Pre-trained Bottlenecked Masked Autoencoders are Better Dense Retrievers. arXiv preprint arXiv:2212.07841 (2022).

[68] Lixin Zou, Weixue Lu, Yiding Liu, Hengyi Cai, Xiaokai Chu, Dehong Ma, Daiting Shi, Yu Sun, Zhicong Cheng, Simiu Gu, et al. 2022. Pre-trained language model-based retrieval and ranking for Web search. ACM Transactions on the Web 17, 1 (2022), 1–36.