Pre-training for Ad-hoc Retrieval: Hyperlink is Also You Need

Zhengyi Ma[1†], Zhicheng Dou[1], Wei Xu[1], Xinyu Zhang[2], Hao Jiang[2], Zhao Cao[2], Ji-Rong Wen[1,3,4]
[1]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[2]Distributed and Parallel Software Lab, Huawei
[3]Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing, China
[4]Key Laboratory of Data Engineering and Knowledge Engineering, MOE, Beijing, China
{zymaa,dou}@ruc.edu.cn

ABSTRACT

Designing pre-training objectives that more closely resemble the downstream tasks for pre-trained language models can lead to better performance at the fine-tuning stage, especially in the ad-hoc retrieval area. Existing pre-training approaches tailored for IR tried to incorporate weak supervised signals, such as query-likelihood based sampling, to construct pseudo query-document pairs from the raw textual corpus. However, these signals rely heavily on the sampling method. For example, the query likelihood model may lead to much noise in the constructed pre-training data. In this paper, we propose to leverage the large-scale hyperlinks and anchor texts to pre-train the language model for ad-hoc retrieval. Since the anchor texts are created by webmasters and can usually summarize the target document, it can help to build more accurate and reliable pre-training samples than a specific algorithm. Considering different views of the downstream ad-hoc retrieval, we devise four pre-training tasks based on the hyperlinks. We then pre-train the Transformer model to predict the pair-wise preference, jointly with the Masked Language Model (MLM) objective. Experimental results on two large-scale ad-hoc retrieval datasets show the significant improvement of our model compared with the existing methods.

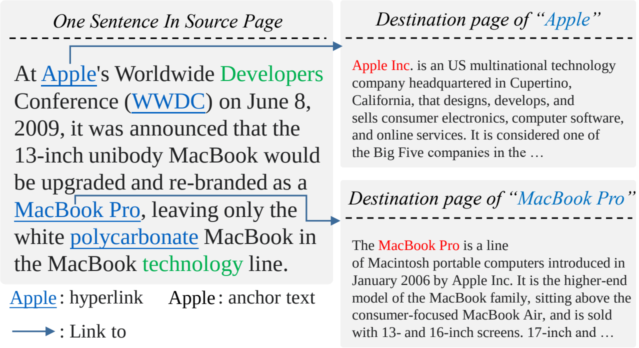
CCS CONCEPTS

• **Information systems → Retrieval models and ranking**.

KEYWORDS

Ad-hoc Retrieval; Pre-training; Hyperlinks; Anchor Texts

ACM Reference Format:
Zhengyi Ma, Zhicheng Dou, Wei Xu, Xinyu Zhang, Hao Jiang, Zhao Cao, and Ji-Rong Wen. 2021. Pre-training for Ad-hoc Retrieval: Hyperlink is Also You Need. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482286


One Sentence In Source Page
Destination page of "Apple"
Destination page of "MacBook Pro"
Apple : hyperlink
Apple : anchor text
: Link to

At Apple's Worldwide Developers Conference (WWDC) on June 8, 2009, it was announced that the 13-inch unibody MacBook would be upgraded and re-branded as a MacBook Pro, leaving only the white polycarbonate MacBook in the MacBook technology line.

Apple Inc. is an US multinational technology company headquartered in Cupertino, California, that designs, develops, and sells consumer electronics, computer software, and online services. It is considered one of the Big Five companies in the …

The MacBook Pro is a line of Macintosh portable computers introduced in January 2006 by Apple Inc. It is the higher-end model of the MacBook family, sitting above the consumer-focused MacBook Air, and is sold with 13- and 16-inch screens. 17-inch and …

Figure 1: An example of the anchor-document relations approximate relevance matches between query-document.

1 INTRODUCTION

Recent years have witnessed the great success of many pre-trained language representation models in the natural language processing (NLP) field [7, 14, 36, 37]. Pre-trained on large-scale unlabeled text corpus and fine-tuned on limited supervised data, these pre-trained models have achieved state-of-the-art performances on many downstream NLP tasks [39, 41, 44]. The success of pre-trained models has also attracted more and more attention in IR community [6, 26, 32, 52, 53]. For example, many researchers have begun to explore the use of pre-trained language models for the ad-hoc retrieval task, which is one of the most fundamental tasks in IR. The task aims to return the most relevant documents given one query solely based on the query-document relevance. Studies have shown that leveraging the existing pre-trained models for fine-tuning the ranking model over the limited relevance judgment data is able to achieve better retrieval effectiveness [16, 31, 32, 53].

Although existing methods of fine-tuning ranking models over pre-trained language models have been shown effective, the pre-training objectives tailored for IR are far from being well explored. Recently, there have been some preliminary studies on this direction [6, 26]. For example, Ma et al. [26] proposed to sample word sets from documents as pseudo queries based on the query likelihood, and use these word sets to simulate query document relevance matching. Different from existing studies, in this work, we propose to **leverage the correlations and supervised signals brought by hyperlinks and anchor texts, and design four novel pre-training objectives to learn the correlations of query and documents for ad-hoc retrieval**.

Hyperlinks are essential for web documents to help users navigating from one page to another. Humans usually select some

[†] This work was done during an internship at Huawei.

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482286

reasonable and representative terms as the anchor text to describe and summarize the destination page. We propose to leverage hyperlinks and anchor texts for IR-oriented pre-training, because: (1) Since anchor texts are usually short and descriptive, based on the classical anchor intuition, **anchor texts share similar characteristics with web queries, and the anchor-document relations approximate relevance matches between query and documents** [9, 15, 48, 55, 57]. For example, as shown in Figure 1, the anchor text "MacBook Pro" is a reasonable query for the introductory page of itself. (2) Anchor texts are created and filtered by web masters (*i.e.*, humans), rather than generated by a specific model automatically. Thus, they can provide more accurate and reliable summarized information of one page, which further brings stronger supervised signals for pre-training. Besides, it can reflect user's information need, and help to model the matching between user needs and documents. (3) Anchor texts can bring terms that are not in the destination page, while the existing methods mostly use the document terms for describing the document. In this way, the model can use more abundant information for capturing semantics and measuring relevance. (4) Hyperlinks widely exist on web pages and are cost-efficient to collect, which can provide large-scale training data for pre-training models. In summary, hyperlinks are appropriate for pre-training tailored for IR, and easy to obtain.

However, straightly building anchor-document pairs to simulate query-document relevance matching may hurt the accuracy of neural retrieval models, since there exist noises even spams in hyperlinks [13, 57]. Besides, the semantics of short anchor texts could be insufficient. For example, as shown in Figure 1, the single term of "Apple" is not a suitable query for the page of "apple company", since "Apple" can also refer to pages about "apple fruit". However, by considering the whole sentence containing the anchor "Apple", we could build more informative queries to describe the page, such as "Apple technology". This indicates that we should leverage the context semantics around the anchor texts for building more accurate anchor-based pre-training data.

Based on the above observation, we propose a pre-training framework **HARP**, which focuses on designing **P**re-training objectives for ad-hoc **R**etreival with **A**nchor texts and **H**yperlinks. Inspired by the self-attentive retrieval architecture [31], we propose to firstly pre-train the language representation model with supervised signals brought by hyperlinks and anchor texts, and then fine-tune the model parameters according to downstream ad-hoc retrieval tasks. The major novelty lies in the pre-training stage. In particular, we carefully devise four self-supervised pre-training objectives for capturing the anchor-document relevance in different views: representative query prediction, query disambiguation, representative document prediction, and anchor co-occurrence modeling. Based on the four tasks, we can build a large number of pair-wise query-document pairs based on hyperlinks and anchor texts. Then, we pre-train the Transformer model to predict pairwise preference jointly with Masked Language Model (MLM) objective. Via such a pre-trained method, HARP can effectively fuse the anchor-document relevance signal data, and learn context-aware language representations. Besides, HARP is able to characterize different situations of ad-hoc retrieval during the pre-training process in a general way. Finally, we fine-tune the learned Transformer model on downstream ad-hoc retrieval tasks to evaluate the performance.

We pre-train the HARP model on English Wikipedia, which contains tens of millions of well-formed wiki articles and hyperlinks. At the fine-tuning stage, we use a ranking model with the same architecture as the pre-trained model. We use the parameters of the pre-trained model to initialize the ranking model, and fine-tune the ranking model on two open-accessed ad-hoc retrieval datasets, including MS-MARCO Document Ranking and Trec-DL 2019. Experimental results show that HARP achieves state-of-the-art performance compared to a number of competitive methods.

Our contributions are three-fold: (1) We introduce the hyperlinks and anchor texts into pre-training for ad-hoc retrieval. By this means, our method can leverage the supervised signal brought by anchor-document relevance, which is more accurate and reliable than the existing methods based on specific sampling algorithms. (2) We design four self-supervised pre-training objectives, including representative query prediction, query disambiguation modeling, representative document prediction, and anchor co-occurrence modeling to pre-train the Transformer model. In such a way, we are able to simulate the query-document matching at the pre-training stage, and capture the relevance matches in different views. (3) We leverage the context semantics around the anchors instead of using the anchor-document relevance straightly. This helps to build more accurate pseudo queries, and further enhance the relevance estimation of the pre-trained model.

2 RELATED WORK

2.1 Pre-trained Language Models

In recent years, pre-trained language models with deep neural networks have dominated across a wide range of NLP tasks [14, 34, 54, 62]. They are firstly pre-trained on a large-scale unlabeled corpus, and fine-tuned on downstream tasks with limited data. With the strong ability to aggregate context, Transformer [46] becomes the mainstream module of these pre-trained models. Some researchers firstly tried to design generative pre-training language models based on uni-directional Transformer [36, 37, 54]. To model the bi-directional context, Devlin et al. [14] pre-trained BERT, which is a large-scale bi-directional Transformer encoder to obtain contextual language representations. Following BERT, many pre-trained methods have achieved encouraging performance, such as robust optimization [25], parameter reduction [21], discriminative training [7], and knowledge incorporation [43, 58]. Inspired by the powerful capacity of BERT for modeling language representations, the IR community has also explored to apply pre-trained models for better measuring the information relevance. By concatenating the query and document with special tokens and feeding them into BERT, many methods has achieved great performance by fine-tuning with BERT [10, 16, 31, 32, 35, 42, 47, 53].

2.2 Pre-training Objectives for IR

Although fine-tuning the downstream IR tasks with pre-trained models has achieved promising results, designing a suitable pre-training objective for ad-hoc retrieval has not been well explored. There have been several successful pre-training tasks for NLP, such as masked language modeling [14, 45], next sentence prediction [14], permutation language modeling [54] and replaced token detection [54]. However, they are designed to model the general

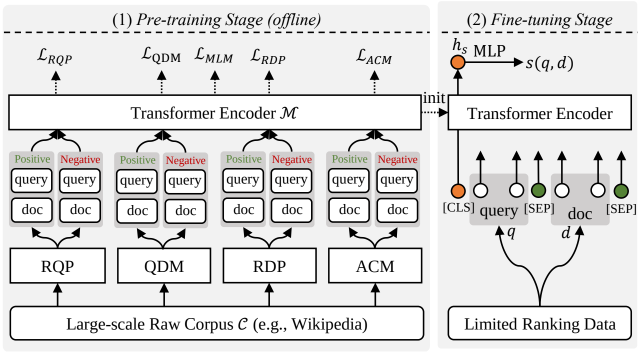


Figure 2: The two-stage architectures of HARP, which consists of (1) pre-training stage, and (2) fine-tuning stage.

contextual dependency or sentence coherence, not the relevance between query-document pairs. A good pre-training task should be relevant to the downstream task for better fine-tuning performance [6]. Some researchers proposed to pre-train on a large-scale corpus with Inverse Cloze Task (ICT) for passage retrieval, where a passage is treated as the document and its inner sentences are treated as queries [6, 23]. Chang et al. [6] also designed Body First Selection (BFS) and Wiki Link Prediction (WLP) to capture the inner-page and inter-page semantic relation. Ma et al. [26] proposed Representative Words Prediction (ROP) task for pre-training in a pair-wise way. They assumed that the sampled word set with higher query likelihood is a more "representative" query. Then, they train the Transformer encoder to predict pairwise scores between two sampled word sets, and achieve state-of-the-art performance.

Different from the above approaches, we propose using the correlations brought by hyperlinks and anchor texts as the supervised signals for the pre-trained language model. Hyperlinks and anchor texts have been used in various existing IR studies, including ad-hoc retrieval [57], query refinement [19], document expansion [15], and query suggestion [11]. However, none of them consider using hyperlinks to design the pre-training objectives for IR. Since hyperlinks widely exist in web documents and can bring complementary descriptions of the target documents, we believe they can bring stronger and more reliable supervised signals for pre-training, which further improve downstream ad-hoc retrieval performance.

3 METHODOLOGY

The key idea of our approach is to leverage the hyperlinks and anchor texts for designing better pre-training objectives tailored for ad-hoc retrieval, and further improve the ranking quality of the pre-trained language model. To achieve this, we design a framework HARP. As shown in Figure 2, the framework of HARP can be divided into two stages: (1) *pre-training* stage and (2) *fine-tuning* stage. In the first stage, we design four pre-training tasks to build the pseudo query-document pairs from the raw corpus with hyperlinks, then pre-train the Transformer model with the four pre-training objectives jointly with the MLM objective. In the second stage, we use the pre-trained model of the first stage to initialize the ranking model, then fine-tune it on the limited retrieval data for proving the effectiveness of our pre-trained model.

In this section, we first provide an overview of our proposed model HARP in Section 3.1, consisting of two stages of pre-training and fine-tuning. Then, we will give the details of the pre-training stage in Section 3.2, and the fine-tuning stage in Section 3.3.

3.1 The Overview of HARP

We briefly introduce the two-stage framework of our proposed HARP as follows.

3.1.1 Pre-training Stage. As shown in Figure 2, in the pre-training stage, we pre-train the Transformer model to learn the query-document relevance based on the hyperlinks and anchor texts. Thus, the input of this stage is the large-scale raw corpus C containing hyperlinks, and the output is the pre-trained Transformer model $\mathcal{M}$. To achieve this, we design four pre-training tasks to capture different views of the anchor-document relevance, generate pseudo query-document pairs to simulate the downstream ad-hoc retrieval task, and pre-train the Transformer model toward these four objectives jointly with the MLM objective. After the offline pre-training on the raw corpus, the Transformer model can learn the query-document matching from the pseudo query-document pairs based on hyperlinks. Thus, it can achieve better performance when applied to the fine-tuning stage.

Based on the above assumptions, we formulate the pre-training stage as follows: Suppose that in a large corpus C (*e.g.*, Wikipedia), we can obtain many textual sentences. We denote one sentence as $S = (w_1, w_2, \cdots, w_n)$, where w_i is the i-th word in S. In sentence S, some words are anchor texts within hyperlinks. We use $A = ((a_1, P_1), (a_2, P_2), \cdots, (a_m, P_m))$ to denote the set of anchor texts in sentence S , where a_i denotes the i-th anchor word in the sentence, and p_i is the destination page. Figure 1 shows an example of anchors in one sentence. For notation simplicity, we treat the multi-words anchor texts as one phrase in the word sequence. In our pre-training corpus with anchors, one source sentence can link to one or more different destination pages using different anchor texts. In the meanwhile, a destination page can also be linked by several source sentences using different anchor texts. In fact, these characteristics of hyperlinks are leveraged in our designed pre-trained tasks to simulate some situations of ad-hoc retrieval. Based on the dataset C, we train a Transformer model $\mathcal{M}$ on this corpus based on four pre-training tasks. The output of the pre-training stage is the model $\mathcal{M}$. Since the pre-training does not depend on any ranking data, the pre-training stage can be done offline for obtaining a good language model $\mathcal{M}$ from the large-scale corpus.

3.1.2 Fine-tuning Stage. As shown in Figure 2, in the fine-tuning stage, we use the pre-trained Transformer model to calculate the relevance score of a query and a document. Fine-tuned on the limited ranking data, our model can learn the data distribution of the specific downstream task and be used for ranking. The formulation of the fine-tuning stage is the same as the ad-hoc retrieval task. Given a query q and a candidate document d, we learn a score function $s(q, d)$ to measure the relevance between q and d. Then, for each candidate document d, we calculate the relevance score of them and return the documents with the highest scores. Specifically, we concatenate the query and document together, and feed them into the Transformer model. Note that the parameter and embeddings of

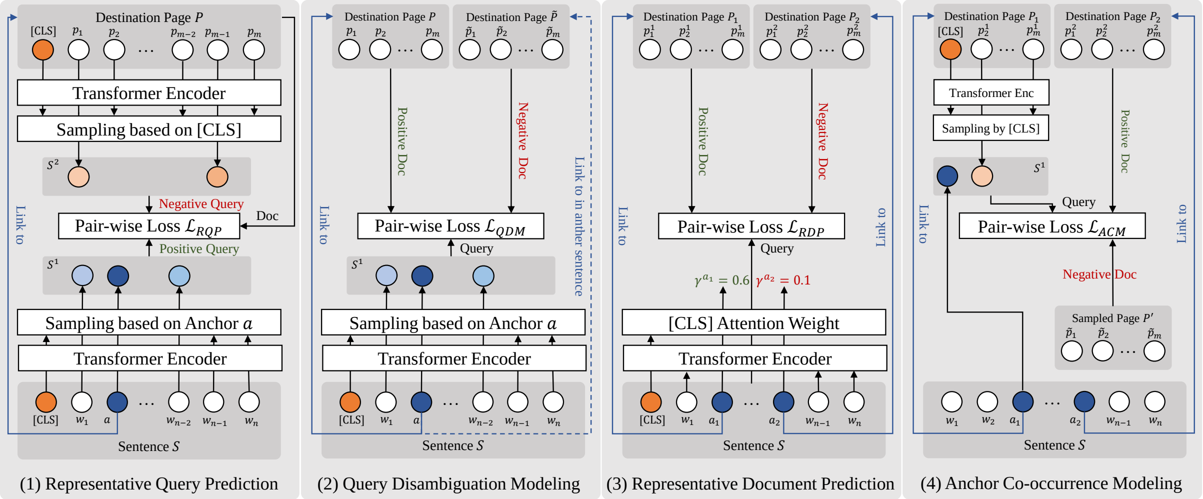


Figure 3: The proposed four pre-training tasks based on hyperlinks: (1) Representative Query Prediction, (2) Query Disambiguation Modeling, (3) Representative Document Prediction, and (4) Anchor Co-occurrence Modeling.

this Transformer model are initialized by the pre-trained model $\mathcal{M}$ in the first stage. Then, we calculate the representations of the [CLS] token at the sequence head and apply a multi-layer perception(MLP) function over this representation to generate the relevance score.

3.2 Pre-training based on Hyperlinks

In the pre-training stage, a pre-training task that more closely resembles the downstream task can better improve the fine-tuning performance. As we introduced in Section 1, the relations between anchor texts and documents can match the relevance of query and documents. Thus, we can leverage these supervised signals brought by anchor texts to build reliable pre-training query-document pairs. Training on these pairs, the model can learn the query-document matching in the pre-training stage, further enhance the downstream retrieval tasks. To achieve this, we design four pre-training tasks based on hyperlinks to construct different loss functions. The architecture of the four pre-training tasks is shown in Figure 3. These four tasks try to learn the correlation of ad-hoc retrieval in different views. Thus, the focus of each task is how to build the query-document pair for pre-training. In the following, we will present the proposed four pre-training tasks in detail.

3.2.1 Representative Query Prediction (RQP). Based on the classic anchor intuition, the relation between anchor texts and the destination page can approximate the query-document relevance [9, 15, 48, 55, 57]. Therefore, our first idea is that the anchor texts could be viewed as a more representative query compared to the word set S^2 directly sampled from the destination page. However, since the anchor texts are usually too short, the semantics they carry could be limited [24, 50, 60]. Fortunately, with the contextual information in the anchor's corresponding sentence S, we can build a more informative pseudo query S^1 with not only the anchor text, but also the contexts in the sentence. The anchor-based context-aware query S^1 should be more representative than the query S^2 comprised of terms sampled from the destination page. We train the model to predict the pair-wise preference of the two queries S^1 and S^2.

Specifically, inspired by the strong ability of BERT [14] to aggregate context and model sequences, we firstly use BERT to calculate the contextual word representations of the sentence S. Specifically, for a sentence $S = (w_1, w_2, \cdots, w_n)$, we get its contextual representation $H = (h_1, h_2, \cdots, h_n)$, where h_t denotes a d-dimension hidden vector of the t-th sentence token. Assume that for anchor text a in sentence s, the corresponding hidden vector a is h_a. We calculate the self-attention weight α^t of each word w_t based on the anchor text a as the average weights across D heads:

$$\alpha^t = \frac{1}{D}\sum_{i=1}^{D}\alpha_i^t = \frac{1}{D}\sum_{i=1}^{D}\mathrm{softmax}(\frac{W_i^Q h_a \cdot W_i^K h_t}{\sqrt{d/D}}), \tag{1}$$

where α_i^t is the attention weight on the i-th head. Typically, a term may appear multiple times within the same sentence. Thus, we add up the attention weights of the same tokens over different positions in the sentence S. Specifically, for each distinct term w_k in the vocabulary $V = \{w_k\}_{k=1}^{K}$, we calculate the final weight of distinct token w_k as:

$$\beta_{w_k} = \sum_{w_t = w_k}\alpha^t. \tag{2}$$

Finally, we normalize the distinct weights of all terms in the vocabulary to obtain a distribution $p(w_k)$ across the terms as:

$$p(w_k) = \frac{\exp(\beta_{w_k})}{\sum_{w_k \in V}\exp(\beta_{w_k})}. \tag{3}$$

The term distribution can measure the contextual similarity between the word w_k and anchor text a. Thus, we use this distribution for sampling l words from the sentence s to form the query S^1. Based on this distribution, the words relevant to the anchor text can be sampled with a higher probability. Thus, we can build a query S^1 based on the reliable signals of the anchor texts. Following [1, 26], the size l of the pseudo query is calculated through a

Poisson distribution as:

$$P(x) = \frac{\lambda^x e^{-\lambda}}{x}, x = 1, 2, 3, \cdots . \tag{4}$$

Finally, we collect the l sampled words and the anchor text a together, and construct a word set S^1 of length l+1. Since S^1 is formed from the anchor text a and its contextual words, there could be high relevance between this word set and destination page of a.

For constructing the pair-wise loss, we also need to construct the negative query. Since proper hard negatives can help to train a better ranking model [18, 52], we propose to sample representative words from the destination page P to construct the hard negative pseudo query for the page, rather than sample words from unrelated pages randomly. For selecting representative words to build the negative query, we firstly use BERT to generate the contextual representations of $P = (p_1, p_2, ...)$ as $(h_1^P, h_2^P ...)$, where h_t^P is the hidden state of the t-th term in P. Then, we also use the self-attention weights to measure the sampling probability of terms in page P. Unlike the phrase for constructing S^1, we calculate the self-attention weights of each terms based on the special token [CLS] as:

$$\alpha_i^t = \begin{cases} \text{softmax}(\frac{W_i^Q h_{[\text{CLS}]} \cdot W_i^K h_t^P}{\sqrt{d/D}}), & P_t \neq a \\ 0, & P_t = a \end{cases} \tag{5}$$

where P_t is the hidden state of the t-th term in passage P. For the term in anchor text a, we set their weight to 0. Thus, the anchor text will not be selected into S^2, and the relevance signal of anchor text in S^1 will indeed enhance the Transformer model. We then perform sum operation for repetitive words following Equation (2), normalize the term distribution following (3), and generate the word set S^2 from passage P. The word set S^2 generated from passage P will be used as the negative query.

Finally, we formulate the objective of the Representative Query Prediction task by a typical pairwise loss, *i.e.*, hinge loss for the pre-training as:

$$\mathcal{L}_{\text{RQP}} = \max(0, 1 - p(S_1|P) + p(S_2|P)), \tag{6}$$

where $p(S|P)$ is the matching score between the word set S and the page P. We concatenate the word set S and P as a single input sequence and feed into the the Transformer with delimiting tokens [SEP]. Then, we calculate the matching score by applying a MLP function over the classification token's representation as:

$$p(S|P) = \text{MLP}(\mathbf{h}^{[\text{CLS}]}), \tag{7}$$

$$\mathbf{h}^{[\text{CLS}]} = \text{Transformer}([\text{CLS}] + \text{S} + [\text{SEP}] + \text{P} + [\text{SEP}]). \tag{8}$$

3.2.2 *Query Disambiguation Modeling (QDM).*

In real-world applications, the queries issued by users are often short and ambiguous [27, 40, 59, 61], such as the query "Apple" (Apple fruit or Apple company?). Thus, building an accurate encoding of the input query is difficult, which further leads to the poor quality of these ambiguous queries. Fortunately, with hyperlinks and anchor texts, we can endow the language representation model with the ability to disambiguate queries in the pre-training stage. We observe that the same anchor texts could link to one or more different pages. Under these circumstances, the anchor could be viewed as an ambiguous query, while the context around the anchor text could help to disambiguate the query. We train the model to predict the true destination page with the semantic information brought by the query context, thus learn disambiguation ability at the pre-training stage.

Specifically, For each distinct anchor text a, we collect all of its occurrence in corpus C as $C^a = ((a, S_1, P_1), (a, S_2, P_2), \cdots , (a, P_{|a|}, p_{|a|}))$, where (a, S, P) means that the destination page of a is P when a is in sentence S. Assume that for the occurrence (a, S, P), following Section 3.2.1, we can build a context-aware word set S^1 from sentence S based on anchor text a. We treat S^1 as the query, page P as the relevant document. Then we sample a negative page $\tilde{P}$ from the pages $(P_1, P_2, \cdots , P_{|a|})$. We train the model to predict the pair-wise preference between the matching of between the query S^1 and the two pages as:

$$\mathcal{L}_{\text{QDM}} = \max(0, 1 - p(S_1|P) + p(S_1|\tilde{P})), \tag{9}$$

where $p(S|P)$ follows the [CLS] score calculation in Equation (7). As illustrated above, the anchor-based contextual word set S^1 sampled from the sentence S can provide additional semantic information for the pre-trained model. Thus, even the anchor text has also pointed to the negative sample, the model can be trained to learn the fine-grained relevance based on the context information around the anchor text. By leveraging the context, the model will learn the ability to disambiguation.

3.2.3 *Representative Document Prediction.*

Although most queries presented to search engines vary between one to three terms in length, a gradual increase in the average query length has been observed in recent studies [2, 12, 20]. Even though these queries could convey more sophisticated information needs of users, they also carry more noises to the search engine. A common strategy to deal with the long queries is to let the model distinguish the important terms in the queries, then focus more on these terms to improve retrieval effectiveness [3, 4, 22]. At the pre-training stage of ad-hoc retrieval, if the model can be trained with more samples with long queries, it will get more robust when facing long queries while fine-tuning. Besides, the language model should be trained to predict the most representative document for the long query, since the long query could focus on different views. Fortunately, the hyperlinks can help to build pre-training samples containing long queries. We observe that there could be more than one anchor text appearing in one sentence. If we treat the sentence as the query, the destination pages could be the relevant documents for the sentence. However, if the anchor text is more important in the sentence, its destination page would be more representative for the sentence. Inspired by this, we propose to predict the relevant document for the sentence containing more than one hyperlinks.

Specifically, for sentence $S = (w_1, w_2, \cdots , w_n)$ and its anchor texts set $A = ((a_1, P_1), (a_2, P_2), \cdots , (a_m, P_m))$, we sample two anchors based on the anchor importance of this sentence. We will treat the sentence S as a long query, and the two destination pages as the documents. In this way, the page is deemed as a more representative document if its anchor texts are of higher importance. To measure the importance of the anchor text, we use the Transformer encoder to build the context-aware representations of the terms, and calculate the hidden vectors $H = (h_1, h_2, \cdots , h_n)$. Assume that for anchor text a, its hidden vector is h_a. We calculate the self-attention weight of anchor a based on the classification token [CLS] to measure its importance as:

$$\gamma^a = \frac{1}{D}\sum_{i=1}^{D}\gamma_i^a = \frac{1}{D}\sum_{i=1}^{D}\text{softmax}(\frac{W_i^Q h_a \cdot W_i^K h_{[\text{CLS}]}}{\sqrt{d/D}}), \tag{10}$$

where we average the attention weights across D heads. The token [CLS] is an aggregate of the entire sequence representation, and it can represent the comprehensive understanding of the input sequence over all tokens. Thus, the attention weight γ^a could measure the contribution of the anchor a to the entire sentence. Then, we merge the repeat anchor texts in one sentence following Equation (2), normalize the weights to a probability distribution $p(a)$ over all anchor texts following Equation (3) as:

$$p(a) = \frac{\exp(\eta_a)}{\sum_{a \in A}\exp(\eta_a)}, \quad \eta_a = \sum_{a_t = a_k}\gamma^a, \tag{11}$$

According to the importance likelihood $p(a)$ of anchors, we sample two anchor texts (a_1, P_1) and (a_2, P_2) from the sentence S. Suppose that a_1 has a higher importance likelihood than a_2 according to Equation (11). We treat the sentence S as the long query, P_1 as the more representative page and P_2 as the less representative page. We minimize the pair-wise loss $\mathcal{L}_{RDP}$ by:

$$\mathcal{L}_{\text{RDP}} = \max(0, 1 - p(S|P_1) + p(S|P_2)), \tag{12}$$

where $p(S|p_1)$ follows the similar calculation in Equation (7).

3.2.4 Anchor Co-occurrence Modeling. Language representation models try to learn the term semantics by modeling the term co-occurrence relation, including the term co-occurrence in a window [28, 33] and in a sequence [14, 34]. As special terms, the anchor texts also share the co-occurrence relations. Besides, since the destination page can help to provide additional information to understand the anchor texts, we can learn more accurate semantics based on the co-occurrence relation by leveraging these destination pages. Therefore, we propose the Anchor Co-occurrence Modeling (ACM) task to model the similarity between the semantics of the anchors in one sentence. By pre-training with ACM, the model could obtain similar representations for the co-occurrenced anchor texts in one sentence, which further improves its ability to model semantics.

Suppose that for a sentence $S = (w_1, w_2, \cdots, w_n)$ and its anchor texts set $A = ((a_1, P_1), (a_2, P_2), \cdots, (a_m, P_m))$, the anchors in A all share the co-occurrence characteristics with each other. We sample a pair of anchors randomly as (a_1, P_1) and (a_2, P_2). We then sample some important words from the page P_1 to form a word set S^1. Then, we let the model learn the semantic matching between S^1 and the passage P^2, thus incorporating the anchor co-occurrence into the pre-trained model. Specifically, for the destination page P_1 of anchor a_1, we use a Transformer encoder to build contextual representations and use the attention weight of [CLS] to measure the term importance:

$$\mu_i^t = \text{softmax}(\frac{W_i^Q h_{[\text{CLS}]} \cdot W_i^K p_t}{\sqrt{d/D}}), \tag{13}$$

where μ_i^t is the attention weight of t-th term based on [CLS] token on the i-th head. Then, we average the term weights across all heads as $\mu^t = \frac{1}{D}\sum_{i=1}^{D}\mu_i^t$. After merging repetitive terms and normalization following Section 3.2.1, we sample words based on the final word importance probabilities. Then, these sampled words form the word set S^1 jointly with the anchor text a_1. Since the anchor set S^1 reflect the information of anchor text a_1, we use S^1 and P_2 as the query and the relevant document to learn the semantic matching degree, respectively. We sample a page $\tilde{P}$ from the corpus C as the negative document, then learn the pair-wise loss of Anchor Co-occurrence Modeling as:

$$\mathcal{L}_{\text{ACM}} = \max(0, 1 - p(S_1|P_2) + p(S_1|\tilde{P})). \tag{14}$$

3.2.5 Final Training Objective. Besides the pair-wise loss to measure the relevance between pseudo queries and documents, the pre-trained model also needs to build good contextual representations for them. Following [14, 26], we also adopt the Masked Language Modeling (MLM) as one of our objectives. MLM is a fill-in-the-blank task, which first masks out some tokens from the input, then trains the model to predict the masked tokens by the rest tokens. Specifically, the MLM loss is defined as:

$$\mathcal{L}_{\text{MLM}} = -\sum_{\hat{x} \in m(x)}\log p(\hat{x}|x_{\backslash m(x)}), \tag{15}$$

where x denote the input sentence, and $m(x)$ and $x_{\backslash m(x)}$ denotes the masked tokens and the rest tokens from x, respectively.

Finally, we pre-train the Transformer model $\mathcal{M}$ towards the proposed four objectives jointly with the MLM objective as:

$$\mathcal{L} = \mathcal{L}_{\text{RQP}} + \mathcal{L}_{\text{QDM}} + \mathcal{L}_{\text{RDP}} + \mathcal{L}_{\text{ACM}} + \mathcal{L}_{\text{MLM}}. \tag{16}$$

All parameters are optimized by the loss function and the whole model is trained in an end-to-end manner.

3.3 Fine-tuning for Document Ranking

In the previous pre-training stage, we pre-train the Transformer model $\mathcal{M}$ to learn the IR matching from the raw corpus based on the hyperlinks and anchor texts. We now incorporate $\mathcal{M}$ into the downstream document ranking task to evaluate the effectiveness of our proposed pre-trained method.

Previous studies have explored utilizing Transformer to measure the sequence pair relevance for ad-hoc document ranking [31, 32, 35]. For the query q and a candidate document d, we aim to calculate a ranking score $s(q, d)$ to measure the relevance between them based on the pre-trained Transformer. Therefore, in this stage, we firstly use the same Transformer architecture as the pre-trained model $\mathcal{M}$, and use the parameters and embeddings of $\mathcal{M}$ to initialize the Transformer model. Then, we add special tokens and concatenate the query and the document as $Y = ([\text{CLS}]; q; [\text{SEP}]; d; [\text{SEP}])$, where $[;]$ is the concatenation operation. A [SEP] token is added at the tail of query and document, while a [CLS] token is added at the sequence head for summary. Finally, We feed the concatenated sequence into Transformer, and use the representations of [CLS] to calculate the final ranking score as:

$$s(q, d) = \text{MLP}(h_s), \quad h_s = \text{Transformer}(Y)_{[\text{CLS}]}. \tag{17}$$

To train the model, we use the cross-entropy loss for optimization:

$$\mathcal{L}_{rank} = \frac{1}{N}\sum_{i=1}^{N} y_i \log(s(q, d)) + (1 - y_i)\log(1 - s(q, d)), \tag{18}$$

where N is the number of samples in the training set.

Table 1: The data statistics of four pre-training tasks.

Tasks	#Tokens	#Pairs	Avg. query length	Avg. doc length
RQP	3.33B	17.71M	2.64	90.48
QDM	0.22B	1.35M	2.80	81.17
RDP	1.56B	6.49M	34.90	85.86
ACM	1.97B	12.64M	4.55	73.64

4 EXPERIMENTS

4.1 Datasets and Evaluation Metrics

4.1.1 Pre-training Corpus. We use English Wikipedia (2021/01/01)[1] as the pre-training corpus, since they are publicly available and have a large-scale collection of documents with hyperlinks for supporting pre-training. Following [26, 43], we use the public WikiExtractor[2] to process the download Wikipedia dump while preserving the hyperlinks. After removing the articles whose length is less than 100 words for data cleaning, it comprises 15,492,885 articles. The data for our proposed four tasks are generated from these articles. We pre-train the model on one combined set of query-document pairs, where each pair is uniformly sampled from the four pre-training tasks in Table 1.

4.1.2 Fine-tuning Datasets. To prove the effectiveness of the proposed pre-training methods, we conduct fine-tuning experiments on two representative ad-hoc retrieval datasets.

- **MS MARCO Document Ranking (MS MARCO)**[3] [30]: It is a large-scale benchmark dataset for document retrieval task. It consists of 3.2 million documents with 367 thousand training queries, 5 thousand development queries, and 5 thousand test queries. The relevance is measured in 0/1.
- **TREC 2019 Deep Learning Track (TREC DL)**[4] [8]: It replaces the test queries in MS MARCO with a novel set with more comprehensive notations. Its test set consists of 43 queries, and the relevance is scored in 0/1/2/3.

4.1.3 Evaluation Metrics. Following the official instructions, we use MRR@100 and nDCG@10 to measure and evaluate the top-ranking performance. Besides, we also calculate MRR@10 and nDCG@100 for MS MARCO and TREC DL, respectively.

4.2 Baselines

We evaluate the performance of our approach by comparing it with three groups of highly related and strong baseline methods:

(1) *Traditional IR models.* **QL** [56] is one of the best performing models which measure the query likelihood of query with Dirichlet prior smoothing. **BM25** [38] is another famous and effective retrieval method based on the probability retrieval model.

(2) *Neural IR models.* **DRMM** [17] is a deep relevance matching model which performs histogram pooling on the transition matrix and uses the binned soft-TF as the input to a ranking neural network. **DUET** [29] propose to use two separate networks to match queries and documents with local and learned distributed representations, respectively. The two networks are jointly trained as part of a single neural network. **KNRM** [51] is a neural ranking model which extracts the features of interaction between query and document terms. The kernel-pooling is used to provide soft match signals for ranking. **Conv-KNRM** [51] is an upgrade of the KNRM model. It adds a convolutional layer for modeling n-gram soft matches and fuse the contextual information of surrounding words for matching.

(3) *Pre-trained Models.* **BERT** [14] is the multi-layer bi-directional Transformer pre-trained with Masked Language Modeling and Next Sentence Prediction(NSP) tasks. **Transformer$_{\text{ICT}}$** [6] is the BERT model retrained with the Inverse Cloze Task (ICT) and MLM. It is specifically designed for passage retrieval in QA scenarios, which teaches the model to predict the removed sentence given a context text. **Transformer$_{\text{WLP}}$** [6] is the BERT model retrained with the Wiki Link Prediction (WLP) and MLM. It is designed for capturing inter-page semantic relations. **PROP** [26] is the state-of-the-art pre-trained model tailored for ad-hoc retrieval. It uses the Representative Words Prediction task for learning the matching between the sampled word sets. We experiment with both of the released models pre-trained on Wikipedia and MS MARCO corpus, *i.e.*, PROP$_{\text{Wiki}}$ and PROP$_{\text{MARCO}}$, respectively.

4.3 Implementation Details

4.3.1 Model Architecture. For our methods HARP, we use the same Transformer encoder architecture as BERT$_{\text{base}}$. The hidden size is 768, and the number of self-attention heads is 12. For a fair comparison, all of the pre-trained baseline models use the same architecture as our model. we use the HuggingFace's Transformers for the model implementation [49].

4.3.2 Pre-training Settings. For the construction of the pseudo queries, we set the expectation of interval λ as 3, and remove the stopwords using the INQUERY stopwords list following [26]. We use the first section to denote the destination page because it is usually the description or summary of a long document [6, 10, 32]. For the MLM objective, we follow the settings in BERT, where we randomly select 15% words for prediction, and the selected tokens are (1) the [MASK] token 80% of the time, (2) a random token 10% of the time, and (3) the unchanged token 10% of the time. We use the Adam optimizer with a learning rate of 1e-4 for 10 epochs, where the batch size is set as 128. For the large cost of training from scratch, we use BERT$_{\text{base}}$ to initialize our method and baseline models.

4.3.3 Fine-tuning Settings. In the fine-tuning stage, the learned parameters in the pre-training stage are used to initialize the embedding and self-attention layers of our model. Following the previous works, we only test the performance of our model on the re-ranking stage [26, 31]. To test the performance of our models with different quality of the candidate document set, we re-rank the document from the two candidate sets, *e.g.*, ANCE Top100 and Official Top100. ANCE Top100 is retrieved based on the ANCE model proposed by Xiong et al. [52], and Official Top100 is released by the official MS MARCO and TREC teams. While fine-tuning, we concatenate the title, URL, body of one document as the document content. The batch size is set as 128, and the maximum length of the input sequence is 512. We fine-tune for 2 epochs, with a 1e-5 learning rate and a warmup portion 0.1. Our code is available online[5].

[1] https://dumps.wikimedia.org/enwiki/

[2] https://github.com/attardi/wikiextractor

[3] https://github.com/microsoft/MSMARCO-Document-Ranking

[4] https://microsoft.github.io/msmarco/TREC-Deep-Learning-2019.html

[5] https://github.com/zhengyima/anchors

Table 2: Evaluation results of all models on two large-scale datasets. "†" denotes the result is significantly worse than our method HARP in t-test with $p < 0.05$ level. The best results are in bold and the second best results are underlined.

Model Type	Model Name	MS MARCO				TREC DL 2019			
		ANCE Top100		Official Top100		ANCE Top100		Official Top100	
		MRR@100	MRR@10	MRR@100	MRR@10	nDCG@100	nDCG@10	nDCG@100	nDCG@10
Traditional IR Models	QL	.2457†	.2295†	.2103†	.1977†	.4644†	.5370†	.4694†	.4354†
	BM25	.2538†	.2383†	.2379†	.2260†	.4692†	.5411†	.4819†	.4681†
Neural IR Models	DRMM	.1146†	.0943†	.1211†	.1047†	.3812†	.3085†	.4099†	.3000†
	DUET	.2287†	.2102†	.1445†	.1278†	.3912†	.3595†	.4213†	.3432†
	KNRM	.2816†	.2740†	.2128†	.1992†	.4671†	.5491†	.4727†	.4391†
	Conv-KNRM	.3182†	.3054†	.2850†	.2744†	.4876†	.5990†	.5221†	.5899†
Pretrained IR Models	BERT	.4184†	.4091†	.3830†	.3770†	.4900	.6084†	.5289†	.6358†
	Transformer$_{\text{ICT}}$	.4194†	.4101†	.3828†	.3767†	.4906	.6053†	.5300	.6386†
	Transformer$_{\text{WLP}}$	.3998†	.3900†	.3698†	.3635†	.4891†	.6143	.5245†	.6276†
	PROP$_{\text{Wiki}}$	.4188†	.4092†	.3818†	.3759†	.4882†	.6050†	.5251†	.6224†
	PROP$_{\text{MARCO}}$	.4201†	.4111†	.3856†	.3800†	.4894	.6166	.5242†	.6208†
	HARP (ours)	**.4472**	**.4393**	**.4012**	**.3961**	**.4949**	**.6202**	**.5337**	**.6562**

Table 3: Document Ranking Performance measured on MS MARCO leaderboard. As the leaderboard only reports aggregated metrics, we cannot report statistical significance.

Method	Dev MRR@100	Eval MRR@100
PROP (ensemble v0.1)	.4551	.4010
BERT-m1 (ensemble)	.4633	.4075
LCE Loss (ensemble)	.4641	.4054
HARP (single)	.4472	.3895
HARP (ensemble)	.4711	.4159

Table 4: Evaluation results of ablation models. "†" denotes the result is significantly worse than our method HARP in t-test with $p < 0.05$ level. The best results are in bold.

Model Name	ANCE Top100		Official Top100	
	MRR@100	MRR@10	MRR@100	MRR@10
BERT	.4184†	.4091†	.3830†	.3770†
w/o MLM	.4435	.4357	.3947†	.3914†
w/o RQP	.4361†	.4278†	.3918†	.3865†
w/o QDM	.4423†	.4340†	.3931†	.3879†
w/o RDP	.4387†	.4305†	.3917†	.3867†
w/o ACM	.4424†	.4341	.3934†	.3882†
HARP (ours)	**.4472**	**.4393**	**.4012**	**.3961**

4.4 Experimental Results

Since the MS MARCO leaderboard limits the frequency of submissions, we evaluate our method and baseline methods on MS MARCO's development set. For TREC DL, we evaluate the test set of 43 queries. The overall performance on the two datasets is reported in Table 2. We can observe that:

(1) Among all models, **HARP achieves the best results in terms of all evaluation metrics**. HARP improves performance with a large margin over two strongest baselines PROP$_{\text{Wiki}}$ and PROP$_{\text{MARCO}}$, which also design objectives tailored for IR. Concretely, HARP significantly outperforms PROP$_{\text{MARCO}}$ by 6.4% in MRR@100 on MS MARCO ANCE Top100. On TREC-DL ANCE Top100 in terms of nDCG@100, HARP outperforms PROP$_{\text{MARCO}}$ by 1.1%. The reason for the improvement reduction on the TREC DL set is that it uses binary notations in the training set but a multi-label notation in the test set, which leads to a gap and difficulty increase. Besides, HARP outperforms the best baselines for both the ANCE Top100 set and the Official Top100 set. These results demonstrate that HARP can capture better matching under the different quality of the candidate list, while not being limited by the candidate list quality or confused by the harder negatives. All these results prove that introducing hyperlinks and anchor texts into pre-training can improve the ranking quality of the pre-trained language model.

(2) **All pre-trained methods outperform methods without pre-training**, indicating that pre-training and fine-tuning are helpful for improving the relevance measuring of models for downstream ad-hoc retrieval. Traditional IR models QL and BM25 are strong baselines on the two datasets, but loses the ability of modeling semantic relevance. Neural IR models use distributed representations to denote the query and document, then apply deep neural networks to measure the IR relevance. Thus, the neural method Conv-KNRM significantly outperforms the traditional methods. The pre-trained methods have dramatic improvements over other methods. This indicates that pre-training on a large corpus and then fine-tuning on downstream tasks is better than training a neural deep ranking model from scratch.

(3) Among all pre-trained methods, **the ones designing objectives tailored for IR perform better**. Transformer$_{\text{ICT}}$ show better performance than BERT, confirming that using a pre-trained task related to retrieval is helpful for downstream tasks. However, **Transformer$_{\text{WLP}}$** perform worse than BERT and **Transformer$_{\text{ICT}}$**. One possible reason is that the queries generated from WLP are noisy since there could be many links in the passage that contribute little to the passage semantics. PROP$_{\text{Wiki}}$ and PROP$_{\text{MARCO}}$ are the

state-of-the-art baselines, which design Representative Words Prediction task tailored for IR. Different from the existing objectives, we design four pre-training tasks based on hyperlinks and anchor texts, which bring more accurate and reliable supervised signals. Hence, HARP achieves significant improvements compared with the existing pre-trained methods.

Besides, to further prove the effectiveness of HARP, we also report some leaderboard results of MS MARCO on eval set in Table 3. We select some representative methods from the leaderboard as the baselines [5, 16, 26]. Following other recent leaderboard submissions, we further incorporate model ensemble. Our ensemble entry uses an trained ensemble of using BERT, RoBERTa [25] and ELECTRA [7] to fine-tune the downstream task. The leaderboard results confirm the effectiveness of our proposed HARP model.

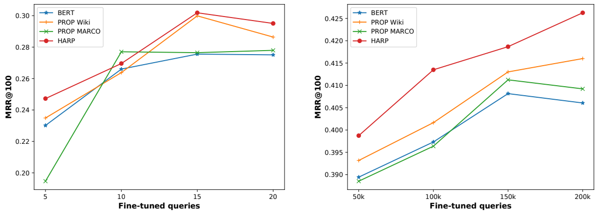
4.5 Further Analysis

We further analyze the influence of different pre-training tasks we proposed (Section 4.5.1), and the performance under different scales of fine-tuning data (Section 4.5.2).

4.5.1 Ablation Study. Our proposed pre-training approach HARP designs four pre-training objectives to leverage hyperlinks and anchor texts tailored for IR. We remove one of them once a time to analyze its contribution. Note that when none of the pre-training tasks are used, our model degenerates to using BERT for fine-tuning directly. Thus, we also provide the result of BERT for comparison. We report the MRR@100 and MRR@10 on MS MARCO dataset.

From the results in Table 4, we can observe that removing any pre-training task would lead to a performance decrease. It indicates that all the pre-training tasks are useful to improve the ranking performance. Specifically, removing RQP causes the most decline in all metrics, which confirms that the correlations and supervised signals brought by hyperlinks can improve the ranking ability of our model in the pre-training phase. The significant performance degradation caused by removing RDP shows that pre-training with long queries contributes to further enhancement of ranking relevance modeling. The influence of removing QDM and ACM is relatively smaller. It proves that considering ambiguous query and modeling the anchor co-occurrence are effective but limited, since the pre-training pairs of QDM are less than other tasks, and the queries sampled from the neighboring anchors in ACM are noisier than the anchors. Removing MLM shows the slightest performance decrease, which indicates that good representations obtained by MLM may not be sufficient for ad-hoc retrieval tasks. It is clearly seen that all model variants perform better than BERT, which is not pre-trained by the IR-oriented objectives.

4.5.2 Low-Resource and Large-scale Settings. Neural ranking models require a considerable amount of training data to learn the representations and matching features. Thus, they are likely to suffer from the low-resource settings in real-world applications, since collecting relevant labels for large-scale queries and documents is costly and time-consuming. This problem can be alleviated by our proposed method, because the pre-training tasks based on hyperlinks and anchor texts can better measure the matching features and resemble the downstream retrieval tasks. To prove that, we simulate the sparsity scenarios by using different scales of queries.



(a) Few-shot queries

(b) large-scale queries

Figure 4: Performance on different scales of fine-tune data

For low-resource settings, we randomly pick 5/10/15/20 queries and fine-tune our model. Besides, we also pick 50k/100k/150k/200k queries to evaluate the performance on different large-scale queries. We report MRR@100 to evaluate the performance. We find:

(1) As shown in Figure 4(a), under few-shot settings, HARP can achieve better results compared to other models, showing the scalability for a small number of supervised data. This is consistent with our speculation as tailoring pre-training objectives for IR can provide a solid basis for fine-tuning, which alleviates the influence of data sparsity problem for ranking to some extent.

(2) As shown in Figure 4(b), under large-scale settings, HARP is consistently better than baselines in all cases. This further proves the effectiveness of our proposed methods to introduce hyperlinks and anchor texts for designing pre-training objectives for IR.

(3) When there are large-scale queries, HARP stably performs better when more queries can be used for training. This implies that HARP is able to make better use of fine-tuning data based on the better understandings of IR learned from the pre-training stage.

5 CONCLUSION

In this work, we propose a novel pre-training framework HARP tailored for ad-hoc retrieval. Different from existing IR-oriented pre-training objectives, we propose to leverage the supervised signals brought by hyperlinks and anchor texts. We devise four pre-training tasks based on hyperlinks, and capture the anchor-document correlations in different views. We pre-train the Transformer model to predict the pair-wise loss functions built by the four pre-training tasks, jointly with the MLM objective. To evaluate the performance of the pre-trained model, we fine-tune the model on the downstream document ranking tasks. Experimental results on two large-scale representative and open-accessed datasets confirm the effectiveness of our model on document ranking.

ACKNOWLEDGMENTS

Zhicheng Dou is the corresponding author. This work was supported by National Natural Science Foundation of China No. 61872370 and No. 61832017, Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, Shandong Provincial Natural Science Foundation under Grant ZR2019ZD06, and Intelligent Social Governance Platform, Major Innovation and Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. We also wish to acknowledge the support of Public Policy and Decision-making Research Lab of Renmin University of China. We are grateful to Yutao Zhu for suggestions on this paper.

REFERENCES

[1] Leif Azzopardi, Maarten de Rijke, and Krisztian Balog. 2007. Building simulated queries for known-item topics: an analysis using six european languages. In *Proceedings of the SIGIR 2007*. ACM, 455–462.

[2] Niranjan Balasubramanian, Giridhar Kumaran, and Vitor R. Carvalho. 2010. Exploring reductions for long web queries. In *SIGIR 2010*. ACM, 571–578.

[3] Michael Bendersky and W. Bruce Croft. 2008. Discovering key concepts in verbose queries. In *Proceedings of the SIGIR 2008*. ACM, 491–498.

[4] Michael Bendersky, Donald Metzler, and W. Bruce Croft. 2010. Learning concept importance using a weighted dependence model. In *WSDM 2010*. ACM, 31–40.

[5] Leonid Boytsov and Zico Kolter. 2021. Exploring Classic and Neural Lexical Translation Models for Information Retrieval: Interpretability, Effectiveness, and Efficiency Benefits. arXiv:cs.CL/2102.06815

[6] Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training Tasks for Embedding-based Large-scale Retrieval. In *ICLR 2020*. OpenReview.net.

[7] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *ICLR 2020*. OpenReview.net.

[8] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M. Voorhees. 2020. Overview of the TREC 2019 deep learning track. (2020).

[9] Na Dai and Brian D. Davison. 2010. Mining Anchor Text Trends for Retrieval. In *ECIR 2010 (Lecture Notes in Computer Science)*, Vol. 5993. Springer, 127–139.

[10] Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In *SIGIR 2019*. ACM, 985–988.

[11] Van Dang and W. Bruce Croft. 2010. Query reformulation using anchor text. In *Proceedings of the WSDM 2010*. ACM, 41–50.

[12] Sudip Datta and Vasudeva Varma. 2011. Tossing coins to trim long queries. In *Proceeding of the SIGIR 2011*. ACM, 1255–1256.

[13] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural Ranking Models with Weak Supervision. In *SIGIR 2017*.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the NAACL 2019*. ACL, 4171–4186.

[15] Zhicheng Dou, Ruihua Song, Jian-Yun Nie, and Ji-Rong Wen. 2009. Using anchor texts with their hyperlink structure for web search. In *SIGIR 2009*. ACM, 227–234.

[16] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. Rethink Training of BERT Rerankers in Multi-stage Retrieval Pipeline. In *ECIR 2021 (Lecture Notes in Computer Science)*, Vol. 12657. Springer, 280–286.

[17] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *CIKM 2016*. ACM, 55–64.

[18] Vladimir Karpukhin, Barlas Oguz, Sewon Min, et al. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP 2020*. ACL, 6769–6781.

[19] Reiner Kraft and Jason Y. Zien. 2004. Mining anchor text for query refinement. In *Proceedings of the WWW 2004*. ACM, 666–674.

[20] Giridhar Kumaran and Vitor R. Carvalho. 2009. Reducing long queries using query quality predictors. In *Proceedings of the SIGIR 2009*. ACM, 564–571.

[21] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR 2020*. OpenReview.net.

[22] Matthew Lease, James Allan, and W. Bruce Croft. 2009. Regression Rank: Learning to Meet the Opportunity of Descriptive Queries. In *ECIR 2009 (Lecture Notes in Computer Science)*, Vol. 5478. Springer, 90–101.

[23] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the ACL 2019*. ACL, 6086–6096.

[24] Johannes Leveling and Gareth J. F. Jones. 2010. Query recovery of short user queries: on query expansion with stopwords. In *SIGIR 2010*. ACM, 733–734.

[25] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, et al. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019).

[26] Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Xiang Ji, and Xueqi Cheng. 2021. PROP: Pre-Training with Representative Words Prediction for Ad-Hoc Retrieval. In *Proceedings of the WSDM 2021 (WSDM '21)*. ACM, 283–291.

[27] Zhengyi Ma, Zhicheng Dou, Guanyue Bian, and Ji-Rong Wen. 2020. PSTIE: Time Information Enhanced Personalized Search. In *Proceedings of the CIKM 2020*. ACM, 1075–1084. https://doi.org/10.1145/3340531.3411877

[28] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*. 3111–3119.

[29] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to Match using Local and Distributed Representations of Text for Web Search. In *WWW 2017*.

[30] 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 *Proceedings of the Workshop on (NIPS 2016) (CEUR Workshop Proceedings)*, Vol. 1773. CEUR-WS.org.

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

[32] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-Stage Document Ranking with BERT. *CoRR* abs/1910.14424 (2019).

[33] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP 2014*. ACL, 1532–1543.

[34] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the NAACL 2018*. ACL, 2227–2237.

[35] Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the Behaviors of BERT in Ranking. arXiv:cs.IR/1904.07531

[36] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. In *Proceedings of Techinical re-port, OpenAI*.

[37] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language Models are Unsupervised Multitask Learners. (2018).

[38] Stephen E. Robertson and Steve Walker. 1994. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In *Proceedings of the SIGIR 1994*. ACM/Springer, 232–241.

[39] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the NAACL 2003*. ACL, 142–147.

[40] Craig Silverstein, Monika Rauch Henzinger, Hannes Marais, and Michael Moricz. 1999. Analysis of a Very Large Web Search Engine Query Log. *SIGIR Forum* 33, 1 (1999), 6–12. https://doi.org/10.1145/331403.331405

[41] Richard Socher, Alex Perelygin, Jean Wu, et al. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP 2013*. ACL.

[42] Zhan Su, Zhicheng Dou, Yutao Zhu, Xubo Qin, and Ji-Rong Wen. 2021. Modeling Intent Graph for Search Result Diversification. In *Proceedings of the SIGIR 2021*. ACM, 736–746. https://doi.org/10.1145/3404835.3462872

[43] Tianxiang Sun, Yunfan Shao, Xipeng Qiu, et al. 2020. CoLAKE: Contextualized Language and Knowledge Embedding. In *COLING 2020*. 3660–3670.

[44] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS 2014*. 3104–3112.

[45] W. L. Taylor. 1953. “Cloze Procedure”: A New Tool for Measuring Readability. *Journalism Quarterly* 30 (1953), 415 – 433.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS 2017*. 5998–6008.

[47] Mengxi Wei, Yifan He, and Qiong Zhang. 2020. Robust Layout-aware IE for Visually Rich Documents with Pre-trained Language Models. In *SIGIR 2020*.

[48] Robert West, Ashwin Paranjape, and Jure Leskovec. 2015. Mining Missing Hyperlinks from Human Navigation Traces: A Case Study of Wikipedia. In *Proceedings of the WWW 2015*. ACM, 1242–1252.

[49] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, et al. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. arXiv:cs.CL/1910.03771

[50] Haocheng Wu, Wei Wu, Ming Zhou, Enhong Chen, Lei Duan, and Heung-Yeung Shum. 2014. Improving search relevance for short queries in community question answering. In *WSDM 2014*. ACM, 43–52.

[51] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In *SIGIR 2017*. 55–64.

[52] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, et al. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. *CoRR* abs/2007.00808 (2020).

[53] Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple Applications of BERT for Ad Hoc Document Retrieval. *CoRR* abs/1903.10972 (2019).

[54] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NIPS 2019*. 5754–5764.

[55] Xing Yi and James Allan. 2010. A content based approach for discovering missing anchor text for web search. In *Proceeding of the SIGIR 2010*. ACM, 427–434.

[56] Chengxiang Zhai and John D. Lafferty. 2017. A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. *SIGIR Forum* 51, 2 (2017), 268–276.

[57] Kaitao Zhang, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2020. Selective Weak Supervision for Neural Information Retrieval. In *WWW 2020*. 474–485.

[58] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the ACL 2019*. ACL, 1441–1451.

[59] Yujia Zhou, Zhicheng Dou, Bingzheng Wei, Ruobing Xie, and Ji-Rong Wen. 2021. Group based Personalized Search by Integrating Search Behaviour and Friend Network. In *SIGIR 2021*. ACM, 92–101.

[60] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2020. Encoding History with Context-aware Representation Learning for Personalized Search. In *Proceedings of the SIGIR 2020*. ACM, 1111–1120.

[61] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2020. Enhancing Re-finding Behavior with External Memories for Personalized Search. In *WSDM*. ACM, 789–797.

[62] Yutao Zhu, Kun Zhou, Jian-Yun Nie, Shengchao Liu, and Zhicheng Dou. 2021. Neural Sentence Ordering Based on Constraint Graphs. In *AAAI 2021*. AAAI Press, 14656–14664. https://ojs.aaai.org/index.php/AAAI/article/view/17722