Empowering Dual-Encoder with Query Generator for Cross-Lingual Dense Retrieval

Houxing Ren[1]* **Linjun Shou**[2] **Ning Wu**[2] **Ming Gong**[2] **Daxin Jiang**[2]†
[1]School of Computer Science and Engineering, Beihang University
[2]Microsoft STC Asia
renhouxing@buaa.edu.cn {lisho,wuning,migon,djiang}@microsoft.com

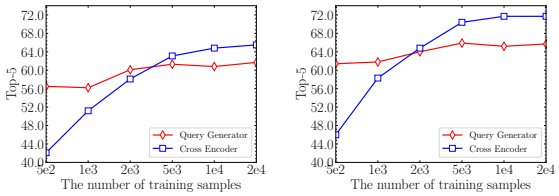
Abstract

In monolingual dense retrieval, lots of works focus on how to distill knowledge from cross-encoder re-ranker to dual-encoder retriever and these methods achieve better performance due to the effectiveness of cross-encoder re-ranker. However, we find that the performance of the cross-encoder re-ranker is heavily influenced by the number of training samples and the quality of negative samples, which is hard to obtain in the cross-lingual setting. In this paper, we propose to use a query generator as the teacher in the cross-lingual setting, which is less dependent on enough training samples and high-quality negative samples. In addition to traditional knowledge distillation, we further propose a novel enhancement method, which uses the query generator to help the dual-encoder align queries from different languages, but does not need any additional parallel sentences. The experimental results show that our method outperforms the state-of-the-art methods on two benchmark datasets.

1 Introduction

Information Retrieval (IR) aims to retrieve pieces of evidence for a given query. Traditional methods mainly use sparse retrieval systems such as BM25 (Robertson and Zaragoza, 2009), which depend on keyword matching between queries and passages. With the development of large-scale pre-trained language models (PLMs) (Vaswani et al., 2017; Devlin et al., 2019) such as BERT, dense retrieval methods (Lee et al., 2019; Karpukhin et al., 2020) show quite effective performance. These methods usually employed a dual-encoder architecture to encode both queries and passages into dense embeddings and then perform approximate nearest neighbor searching (Johnson et al., 2021).

Recently, leveraging a cross-encoder re-ranker as the teacher model to distill knowledge to a dual-encoder has shown quite effective to boost the dual-encoder performance. Specifically, these methods first train a warm-up dual-encoder and a warm-up cross-encoder. Then, they perform knowledge distillation from the cross-encoder to the dual-encoder by KL-Divergence or specially designed methods. For example, RocketQAv2 (Qu et al., 2021) proposed dynamic distillation, and AR2 (Zhang et al., 2021) proposed adversarial training.



(a) BM25. (b) DPR.

Figure 1: The performance of cross-encoder and query generator when varying the number of training samples and retrievers. We use BM25 and DPR as retrievers, respectively. For the cross-encoder (BERT-Large), we use retrieved top-100 passages which do not contain the answer as negative and contrastive loss for training. For the query generator (T5-Base), we firstly train it with the query generation task and then fine-tune the model with the same setting as BERT-Large. The reported performance is the top-5 score of re-ranked top 500 passages on the NQ test set.

However, there are two major problems when scaling the method to the cross-lingual dense retrieval setting. Firstly, the cross-encoder typically requires large amounts of training data and high-quality negative samples due to the gap between pre-training (token-level task) and fine-tuning (sentence-level task), which are usually not satisfied in the cross-lingual setting (Asai et al., 2021a). Due to expensive labeling and lack of annotators in global languages, especially low-resource languages, the training data in cross-lingual are quite limited. Then with the limited training data, the dual-encoder is not good enough to provide

*Work done during internship at Microsoft STCA.
†Corresponding author.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 3107 - 3121
December 7-11, 2022

high-quality negative samples to facilitate the cross-encoder. Secondly, the cross-lingual gaps between different languages have a detrimental effect on the performance of cross-lingual models. Although some cross-lingual pre-training methods such as InfoXLM (Chi et al., 2021) and LaBSE (Feng et al., 2022) have put lots of effort into this aspect by leveraging parallel corpus for better alignment between different languages, these parallel data are usually expensive to obtain and the language alignment could be damaged in the fine-tuning stage if without any constraint.

To solve these problems, we propose to employ a query generator in the cross-lingual setting, which uses the likelihood of a query against a passage to measure the relevance. On the one hand, the query generator can utilize pre-training knowledge with small training data in fine-tuning stage, because both of its pre-training and fine-tuning have a consistent generative objective. On the other hand, the query generation task is defined over all tokens from the query rather than just the *[CLS] token* in the cross-encoder, which has been demonstrated to be a more efficient training paradigm (Clark et al., 2020). As shown in Figure 1, with the number of training samples dropping, the performance of BERT-Large drops more sharply than T5-Base. Besides, the query generator is less sensitive to high-quality negative samples. As we can see, using BM25 as the retriever to mine negative samples for re-ranker training, the gap between cross-encoder and query generator is smaller than the gap using DPR as the retriever. Finally, the query generator can provide more training data by generation, which is precious in the cross-lingual setting. To sum up, the query generator is more effective than the cross-encoder in the cross-lingual setting.

Based on these findings, we propose a novel method, namely QuiCK, which stands Query generator improved dual-encoder by Cross-lingual Knowledge distillation. Firstly, at the passage level, we employ a query generator as the teacher to distill the relevant score between a query and a passage into the dual-encoder. Secondly, at the language level, we use the query generator to generate synonymous queries in other languages for each training sample and align their retrieved results by KL-Divergence. Considering the noise in the generated queries, we further propose a scheduled sampling method to achieve better performance.

The contributions of this paper are as follows:

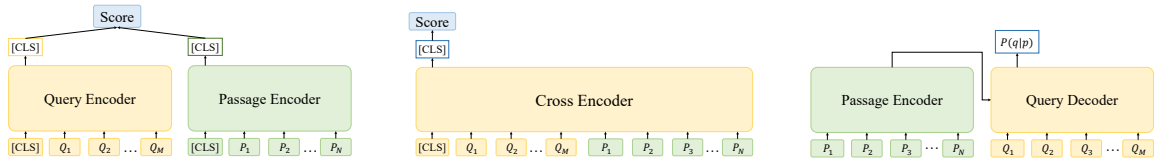
- We propose a cross-lingual query generator as a teacher model to empower the cross-lingual dense retrieval model and a novel iterative training approach is leveraged for the joint optimizations of these two models.
- On top of the cross-lingual query generator, a novel cost-effective alignment method is further designed to boost the dense retrieval performance in low-resource languages, which does not require any additional expensive parallel corpus.
- Extensive experiments on two public cross-lingual retrieval datasets demonstrate the effectiveness of the proposed method.

2 Related Work

Retrieval. Retrieval aims to search relevant passages from a large corpus for a given query. Traditionally, researchers use bag-of-words (BOW) based methods such as TF-IDF and BM25 (Robertson and Zaragoza, 2009). These methods use a sparse vector to represent the text, so we call them sparse retrievers. Recently, some studies use neural networks to improve the sparse retriever such as DocTQuery (Nogueira et al., 2019) and DeepCT (Dai and Callan, 2020).

In contrast to sparse retrievers, dense retrievers usually employ a dual-encoder to encode both queries and passages into dense vectors whose lengths are much less than sparse vectors. These methods mainly focus on two aspects: pre-training (Lee et al., 2019; Guu et al., 2020; Lu et al., 2021; Gao and Callan, 2021a,b; Zhou et al., 2022) and fine-tuning methods, including negative sampling (Karpukhin et al., 2020; Luan et al., 2021; Xiong et al., 2021; Zhan et al., 2021) and multi-view representations (Khattab and Zaharia, 2020; Humeau et al., 2020; Tang et al., 2021; Zhang et al., 2022). Another fine-tuning method is jointly training the dual-encoder with a cross-encoder. For example, RDR (Yang and Seo, 2020) and FID-KD (Izacard and Grave, 2021) distill knowledge from a reader to the dual-encoder; RocketQA (Qu et al., 2021), PAIR (Ren et al., 2021a), RocketQAv2 (Ren et al., 2021b), and AR2 (Zhang et al., 2021) jointly train the dual-encoder with a cross-encoder to achieve better performance.

Recently, with the development of cross-lingual pre-trained models (Conneau et al., 2020), researchers pay more attention to cross-lingual dense



(a) Dual-Encoder. (b) Cross-Encoder. (c) Query Generator.

Figure 2: Overview of different model architectures designed for retrieval or re-ranking.

retrieval (Asai et al., 2021a; Longpre et al., 2020). For example, CORA (Asai et al., 2021b) leverages a generator to help mine retrieval training data, Sentri (Sorokin et al., 2022) proposes a single encoder and self-training, and DR.DECR (Li et al., 2021) uses parallel queries and sentences to perform cross-lingual knowledge distillation.

Re-ranking. Re-ranking aims to reorder the retrieved passages as the relevant scores. Due to the small number of retrieved passages, re-ranking usually employs high-latency methods to obtain better performance, *e.g.,* cross-encoder. Traditionally, the re-ranking task is heavily driven by manual feature engineering (Guo et al., 2016; Hui et al., 2018). With the development of pre-trained language models (*e.g.,* BERT), researchers use the pre-trained models to perform re-ranking tasks (Nogueira and Cho, 2019; Li et al., 2020). In addition to cross-encoder, researchers also try to apply generator to re-ranking. For example, monoT5 (Nogueira et al., 2020) proposes a prompt-based method to re-rank passages with T5 (Raffel et al., 2020) and other studies (dos Santos et al., 2020; Zhuang et al., 2021; Lesota et al., 2021) propose to use the log-likelihood of the query against the passage as the relevance to perform the re-ranking task.

Recently, with the size of pre-trained models scaling up, the generative models show competitive zero-shot and few-shot ability. Researchers start to apply large generative models to zero-shot and few-shot re-ranking. For example, SGPT (Muennighoff, 2022) and UPR (Sachan et al., 2022) propose to use generative models to perform zero-shot re-ranking. P^3 Ranker (Hu et al., 2022) demonstrates that generative models achieve better performance in the few-shot setting. Note that all of these works are concurrent to our work. Instead of using a query generator as a re-ranker only, we propose to leverage the query generator as a teacher model to enhance the performance of the cross-lingual dual-encoder. In addition to the traditional knowledge distillation, we further propose a novel cost-effective alignment method to boost the dense retrieval performance in low-resource languages.

3 Preliminaries

In this section, we give a brief review of dense retrieval and re-ranking. The overviews of all methods are presented in Figure 2.

Dual-Encoder. Given a query q and a large corpus C, the retrieval task aims to find the relevant passages for the query from a large corpus. Usually, a dense retrieval model employs two dense encoders (*e.g.,* BERT) $E_Q(\cdot)$ and $E_P(\cdot)$. They encode queries and passages into dense embeddings, respectively. Then, the model uses a similarity function, often dot-product, to perform retrieval:

$$f_{de}(q, p) = E_Q(q) \cdot E_P(p), \quad (1)$$

where q and p denote the query and the passage, respectively. During the inference stage, we apply the passage encoder $E_P(\cdot)$ to all the passages and index them using FAISS (Johnson et al., 2021) which is an extremely efficient, open-source library for similarity search. Then given a query q, we derive its embedding by $\boldsymbol{v}_q = E_Q(q)$ and retrieve the top k passages with embeddings closest to $\boldsymbol{v}_q$.

Cross-Encoder Re-ranker. Given a query q and top k retrieved passages C, the re-ranking task aims to reorder the passages as the relevant scores. Due to the limited size of the corpus, the re-ranking task usually employs a cross-encoder to perform interaction between words across queries and passages at the same time. These methods also introduce a special token *[SEP]* to separate q and p, and then the hidden state of the *[CLS] token* from the cross-encoder is fed into a fully-connected layer to output the relevant score:

$$f_{ce}(q, p) = \boldsymbol{W} \times E_C(q||p) + b, \quad (2)$$

where "||" denotes concatenation with the *[SEP] token*. During the inference stage, we apply the

cross-encoder $E_C(\cdot)$ to all <q, p> pair and reorder the passages by the scores.

Query Generator Re-ranker. Similar to cross-encoder re-ranker, query generator re-ranker also aims to reorder the passages as the relevant scores. For the query generator re-ranker, we use the log-likelyhood of the query against the passage to measure the relevance:

$$f_{qg}(q,p) = \log P(q|p) = \sum_{t=0} \log P(q_t|q_{<t}, p), \tag{3}$$

where $q_{<t}$ denotes the previous tokens before q_t. The rest of settings are the same as the cross-encoder re-ranker and are omitted here.

Training. The goal of retrieval and re-ranking is to enlarge the relevant score between the query and the relevant passages (*a.k.a.,* positive passages) and lessen the relevant score between the query and the irrelevant passages (*a.k.a.,* negative passages). Let $\{q_i, p_i^+, p_{i,0}^-, p_{i,1}^-, \ldots, p_{i,n}^-\}$ be the i-th training sample. It consists of a query, a positive passage, and n negative passages. Then we can employ the contrastive loss function, called InfoNCE (van den Oord et al., 2018), to optimize the model:

$$\mathcal{L}_R = -\log \frac{e^{f(q_i,p_i^+)}}{e^{f(q_i,p_i^+)} + \sum_{j=0}^{n} e^{f(q_i,p_{i,j}^-)}}, \tag{4}$$

where f denotes the similarity function, *e.g.,* f_{de} in Eq. (1), f_{ce} in Eq. (2), or f_{qg} in Eq. (3).

Cross-lingual Retrieval. In the cross-lingual information retrieval task, passages and queries are in different languages. In this paper, we consider the passages are in English and the queries are in non-English languages. A sample consists of three components: a query in a non-English language, a positive passage in English, and a span answer in English. Given a non-English query, the task aims to retrieve relevant passages in English to answer the query. If a retrieved passage contains the given span answer, it is regarded as a positive passage, otherwise, it is a negative passage.

4 Methodology

In this section, we present the proposed QuiCK. The overview of the proposed method is presented in Figure 3. We start with the training of the query generator, then present how to perform distillation and alignment training for the dual-encoder, and we finally discuss the entire training process.

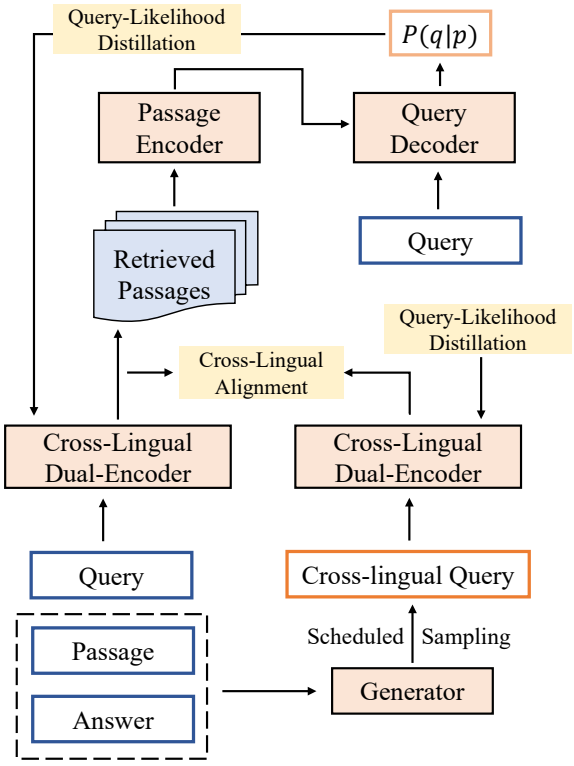


Figure 3: Overview of the proposed QuiCK.

4.1 Training of Query Generator

In our method, we employ mT5 (Xue et al., 2021) as the query generator. The query generator has two roles: teacher and generator. As a teacher, it aims to better re-rank the candidate passages with relevance and distill the knowledge to the dual-encoder. As a generator, it aims to generate synonymous queries in different languages.

Input Format. As we employ mT5, we design a prompt template for input sentences. Considering that most passages are long, we propose introducing the span answer as input to encourage the generator to focus on the same segment and generate parallel queries in different languages. As a result, we use *"generate [language] query: answer: [span answer] content: [content]"* as the template. For a specific sample, we fill the three placeholders with the language of the target query, the span answer, and the passage content, respectively.

Training. Considering the two roles of the query generator, the entire training process for the query generator contains two stages: query generation training and re-ranking training.

Firstly, we train the generator with the generation task, which takes the positive passage as input and aims to generate the query. The task can be for-

mulated as maximizing the conditional probability:

$$\begin{aligned}\hat{q} &= \arg\max_{q} P(q|p,a) \\ &= \arg\max_{q} \prod_{t=0} P(q_t|p,a,q_{<t})\end{aligned}, \tag{5}$$

where q_t is the t-th token of the generated query, a denotes the span answer, and $q_{<t}$ represents the previous decoded tokens. Then we can employ cross-entropy loss to optimize the model:

$$\mathcal{L}_{QG} = \frac{1}{T}\sum_{t=0}^{T} -\log P(q_t|p,a,q_{<t}), \tag{6}$$

where T denotes the number of the query tokens.

Secondly, we train the generator with the re-ranking task, which takes a query and a passage as input and outputs the relevant score of the two sentences. The detailed training process is introduced in Section 3 and is omitted here.

4.2 Distillation for Dual-Encoder

We then present how to distill knowledge from query generator to dual-encoder. Similar to previous methods (Ren et al., 2021b; Zhang et al., 2021), we employ KL-Divergence to perform distillation. Formally, given a query q and a candidate passage set $C_q = \{p_i\}_{1\le i\le n}$ which is retrieved by the dual-encoder, we compute relevant scores by query generator and dual-encoder, respectively. After that, we normalize the scores by softmax and compute the KL-Divergence as the loss:

$$\begin{aligned} s_{qg}(q,p) &= \frac{\exp(f_{qg}(q,p))}{\sum_{p'\in C_q}\exp(f_{qg}(q,p'))}, \\ s_{de}(q,p) &= \frac{\exp(f_{de}(q,p))}{\sum_{p'\in C_q}\exp(f_{de}(q,p'))}, \\ \mathcal{L}_D &= \sum_{p\in C_q} s_{qg}(q,p)\frac{s_{qg}(q,p)}{s_{de}(q,p)}, \end{aligned} \tag{7}$$

where f_{qg} and f_{de} denote the relevant score given by the query generator and the dual-encoder which are presented in Section 3.

4.3 Alignment for Dual-Encoder

Alignment is a common topic in the cross-lingual setting, which can help the model better handle sentences in different languages. Previous works (Zheng et al., 2021; Yang et al., 2022) usually use parallel data or translated data to perform alignment training among different languages. Here, we propose a novel method to align queries in different languages for cross-lingual retrieval, which does not need any parallel data. The core idea of our method is to leverage the query generator to generate synonymous queries in other languages to form parallel cases.

Generation. For each case in the training set, we generate a query in each target language (*a.k.a.,* if there are seven target languages, we generate seven queries for the case). Then we use the confidence of the generator to filter the generated queries. Specially, we set filter thresholds to accept 50% of generated queries.

Scheduled Sampling. In this work, we select a generated query to form a pair-wise case with the source query. Considering the semantics of generated queries, we carefully design a scheduled sampling method to replace the random sampling. For a generated query q', we first use the dual-encoder to retrieve passages for the source query q and generated query q', respectively, namely C_q and C'_q. Then we calculate a coefficient for the generated query q' as

$$\begin{aligned} c' &= \frac{|C_q \cap C'_q|}{\max(|C_q|,|C'_q|)}, \\ c' &= \begin{cases} c' & \text{if } c' \ge T, \\ 0 & \text{if } c' < T, \end{cases} \end{aligned} \tag{8}$$

where threshold T is a hyper-parameter and $|\cdot|$ denotes the size of the set. The basic idea is that the larger the union of retrieved passages, the more likely the queries are to be synonymous. When sampling the generated query, we first calculate coefficients $\{c'_1,\ldots,c'_m\}$ for all generated queries $\{q'_1,\ldots,q'_m\}$, then normalize them as the final sampling probability p:

$$p_i = \frac{c'_i}{\sum_{j=0}^{m} c'_i}, \tag{9}$$

where m denotes the number of generated queries. During the training stage, for each training case, we sample a generated query to form the pair-case with the source query q based on the probabilities.

Alignment Training. After sampling a generated query, we present the how to align the source query and the generated query. Different to previous works (Zheng et al., 2021), we employ asymmetric KL-Divergence rather than symmetric KL-Divergence due to the different quality of the source

Algorithm 1: The training algorithm.

Input: Dual-Encoder R, Query Generator G, Corpus C, and Training Set D.

Initialize R and G with pre-trained model;
Train the warm-up R with Eq. (4) on D;
Train the warm-up G with Eq. (6) on D;
Generate queries for each sample in D;
Build ANN index for R;
Retrieve relevant passages on corpus C;
Fine-tune the G with Eq. (4) on D and retrieved negative passages.
while *models has not converged* **do**
    Fine-tune the R with Eq. (11) on D and retrieved passages;
    Refresh ANN index for R;
    Retrieve relevant passages on corpus C;
    Fine-tune the G with Eq. (4) on D and retrieved negative passages.
end

query and the generated query:

$$\mathcal{L}_A = \sum_{p \in C_q \cup C'_q} c' s_{de}(q,p) \frac{s_{de}(q,p)}{s_{de}(q',p)}, \qquad (10)$$

where q denotes the query, C_q denotes the set of retrieved passages, superscript "$'$" denotes the generated case, and c' is the coefficient of the generated query. Note that s_{de} in Eq. (10) are normalized across $C_q \cup C'_q$ instead of C_q or C'_q in Eq. (7).

4.4 Training of Dual-Encoder

As shown in Figure 3, we combine the distillation loss and the alignment loss as final loss:

$$\mathcal{L} = \mathcal{L}_D + \mathcal{L}'_D + \alpha \times \mathcal{L}_A, \qquad (11)$$

where $\mathcal{L}_D$ denotes the distillation loss for the source queries, $\mathcal{L}'_D$ denotes the distillation loss for the generated queries, $\mathcal{L}_A$ denotes the alignment loss, and α is a hyper-parameter to balance the loss.

Based on the training method of dual-encoder and query generator, we conduct an iterative procedure to improve the performance. We present the entire training procedure in Algorithm 1.

5 Experiments

In this section, we construct experiments to demonstrate the effectiveness of our method.

5.1 Experimental Setup

Datasets. We evaluate the proposed method on two public cross-lingual retrieval datasets: XOR-Retrieve (Asai et al., 2021a) and MKQA (Longpre et al., 2020). The detailed descriptions of the two datasets are presented in Appendix A.

Evaluation Metrics. Following previous works (Asai et al., 2021a; Sorokin et al., 2022), we use R@2kt and R@5kt as evaluation metrics for the XOR-Retrieve dataset and R@2kt as evaluation metrics for the MKQA dataset. The metrics measure the proportion of queries to which the top k retrieved tokens contain the span answer, which is fairer with different passage sizes.

Implementation Details. For the warm-up training stage, we follow XOR-Retrieve to first train the model on NQ (Kwiatkowski et al., 2019) data and then fine-tune the model with XOR-Retrieve data. For the iteratively training stage, we generate seven queries for each case (because the XOR-Retrieve data contains seven languages). We set the number of retrieved passages as 100, the number of iterations as 5, threshold T in Eq. (8) as 0.3 and coefficient α in Eq. (11) as 0.5. The detailed hyper-parameters are shown in Appendix C. And we conduct more experiments to analyze the parameter sensitivity in Appendix D.

All the experiments run on 8 NVIDIA Tesla A100 GPUs. The implementation code is based on HuggingFace Transformers (Wolf et al., 2020). For the dual-encoder, we use XLM-R Base (Conneau et al., 2020) as the pre-trained model and use the average hidden states of all tokens to represent the sentence. For the query generator, we leverage mT5 Base (Xue et al., 2021) as the pre-trained model, which has almost the same number of parameters as a large cross-encoder.

5.2 Results

Baselines. We compare the proposed QuiCK with previous state-of-the-art methods, including mDPR, DPR+MT (Asai et al., 2021a), Sentri (Sorokin et al., 2022), DR.DECR (Li et al., 2021). Note that Sentri introduces a shared encoder with large size, DR.DECR introduces parallel queries and parallel corpus, but our method only utilizes an encoder with base size, XOR-Retrieve and NQ training data. For more fairly comparison, we also report their ablation results. Here, "Bi-Encoder" denotes two unshared encoders with base size. "KD$_{XOR}$" de-

Table 1: Comparison results on XOR-Retrieve dev set. The best results are in bold. "$*$" denotes the results are copied from the source paper. Results unavailable are left blank.

Methods	R@2kt								R@5kt							
	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg
mDPR*	38.8	48.4	52.5	26.6	44.2	33.3	39.9	40.5	48.9	60.2	59.2	34.9	49.8	43.0	55.5	50.2
DPR + MT*	43.4	53.9	55.1	40.2	50.5	30.8	20.2	42.0	52.4	62.8	61.8	48.1	58.6	37.8	32.4	50.6
Sentri*	47.6	48.1	53.1	46.6	49.6	44.3	67.9	51.0	56.8	62.2	65.5	53.2	55.5	52.3	80.3	60.8
w/ Bi-Encoder*	47.8	39.1	48.9	51.2	40.2	41.2	49.4	45.4	55.1	43.3	59.5	59.4	51.2	52.0	56.9	53.9
DR.DECR*	-	-	-	-	-	-	-	66.0	70.2	**85.9**	69.4	65.1	68.8	68.8	83.2	73.1
w/o KD^*_{XOR}	-	-	-	-	-	-	-	60.6	-	-	-	-	-	-	-	68.6
w/o KD^*_{PC}	-	-	-	-	-	-	-	56.6	-	-	-	-	-	-	-	63.6
QuiCK	52.8	70.1	62.2	54.8	62.8	57.8	70.6	61.3	63.8	78.0	65.3	63.5	69.8	67.1	74.8	68.9
QuiCK w/ LaBSE	**67.3**	**78.9**	**65.9**	**59.8**	**66.3**	**63.7**	**80.7**	**68.9**	**72.2**	83.2	**69.7**	**68.0**	**70.9**	**71.7**	**84.9**	**74.4**

Table 2: Comparison results on XOR-Retrieve test set.

Methods	R@2kt	R@5kt
GAAMA	52.8	59.9
Sentri	52.7	61.0
CCP	54.8	63.0
Sentri 2.0	58.5	64.6
DR.DECR	63.0	70.3
QuiCK w/ LaBSE	**65.6**	**72.0**

Table 3: Average performance of 20 unseen languages in MKQA test set. "$*$" denotes the results are copied from the Sentri paper.

Methods	R@2kt
CORA*	41.1
BM25 + MT*	42.0
Sentri*	53.3
w/ Bi-Encoder*	45.3
QuiCK	53.4
QuiCK w/ LaBSE	**60.3**

notes a distillation method which introduces synonymous English queries. "KD_{PC}" denotes a distillation method which introduces parallel corpus. In addition, we also employ LaBSE base (Feng et al., 2022) to evaluate the proposed QuiCK with parallel corpus, which is a state-of-the-art model pre-trained with parallel corpus.

XOR-Retrieve. Table 1 shows the results on XOR-Retrieve dev set. The proposed QuiCK outperforms mDPR, DPR+MT, and Sentri with a clear edge in almost all languages. Although QuiCK does not introduce any parallel corpus, it also outperforms DR.DECR w/o KD_{XOR}. Finally, QuiCK based on LaBSE outperforms all baselines, especially DR.DECR w/o KD_{XOR}, and even outperforms DR.DECR which utilizes both parallel queries and parallel corpus. Note that knowledge distillation with parallel corpus in DR.DECR is designed for cross-lingual dense retrieval, but LaBSE is a general pre-trained model for all cross-lingual tasks. These results show the effectiveness of the proposed QuiCK. Our method combines two methods in dense retrieval and cross-lingual tasks, namely distillation and alignment. We further analyze the contribution of each component in Section 5.3.

In addition, we show the results on XOR-Retrieve test set in Table 2, which is copied from the leaderboard[1] on June 15, 2022. As we can see, our method achieves the top position on the leaderboard of XOR-Retrieve.

MKQA. Furthermore, we evaluate the zero-shot performance of our method on the MKQA test set. Following previous works (Sorokin et al., 2022), we directly evaluate the dual-encoder training on XOR-Retrieve data and report the performance of unseen languages on MKQA. As shown in Table 3, our method outperforms all baselines and even performs better than Sentri. Note that Sentri uses a shared encoder with large size. The comparison between Sentri and Sentri w/ Bi-Encoder shows that the large encoder has better transfer ability. Finally, the proposed QuiCK w/ LaBSE outperforms all baselines with a clear edge. It shows the better transfer ability of our methods.

5.3 Methods Analysis

Ablation Study. Here, we check how each component contributes to the final performance. We construct the ablation experiments on XOR-Retrieve data. We prepare four variants of our method:

[1] https://nlp.cs.washington.edu/xorqa

Table 4: Ablation results on XOR-Retrieve dev set.

Methods	R@2kt	R@5kt
QuiCK	**61.3**	**68.9**
w/o Sampling	59.5	67.5
w/o Alignment	59.9	67.1
w/o Generation	58.8	65.9
w/o All	41.5	53.4

Table 5: Effect of alignment based on different pre-trained languages models.

Methods	XLM-R		LaBSE	
	R@2kt	R@5kt	R@2kt	R@5kt
QuiCK	**61.3**	**68.9**	**68.9**	**74.4**
w/o Alignment	59.9	67.1	67.9	73.8

(1) w/o Sampling denotes without the scheduled sampling but keep the threshold T for c', *a.k.a.*, if $c' \geq T$, then $c' = 1$, otherwise $c' = 0$; (2) w/o Alignment denotes without $\mathcal{L}_A$ in Eq. (11); (3) w/o Generation denotes without $\mathcal{L}'_D$ and $\mathcal{L}_A$ in Eq. (11); (4) w/o All denotes without the enhanced training, *a.k.a.*, the warm-up dual-encoder.

Table 4 presents all comparison results of the four variants. As we can see, the performance rank of R@5kt can be given as: w/o All < w/o Generation < w/o Alignment < w/o Sampling < QuiCK. These results indicate that all components are essential to improve performance. And we can find the margin between w/o Alignment and w/o Sampling is small, it denotes that the generated queries are noisy and demonstrate the effectiveness of our schedule sampling strategy.

Effect of Alignment. As we mentioned in Section 1, the alignment established in the pre-training stage may be damaged without any constraint in the fine-tuning stage. Here, we construct experiments on both XLM-R and LaBSE to analyze the effectiveness of the proposed alignment training. As shown in Table 5, the proposed alignment training is effective based on the two models. It indicates that the alignment constraint in the fine-tuning stage is effective for models which pre-trained with parallel corpus. And we find that the gains of alignment training based on XLM-R are larger than LaBSE, which shows that the alignment constraint is more effective for models which do not pre-trained with parallel corpus.

Cross-Encoder versus Query Generator. Here, we analyze the re-ranking ability of cross-encoder and query generator. Here, we use the warm-up dual-encoder to retrieve passages, vary the number of candidate passages, and then evaluate the re-ranked result. As shown in Figure 4, when we use the top-100 candidate passages, the performance of the cross-encoder and generator is almost the same. But as the number of candidate passages increases, especially when it surpasses 500, the gap between the performance of the cross-encoder and query generator gradually becomes larger. It shows low generalization performance of the cross-encoder when there are not enough training samples.

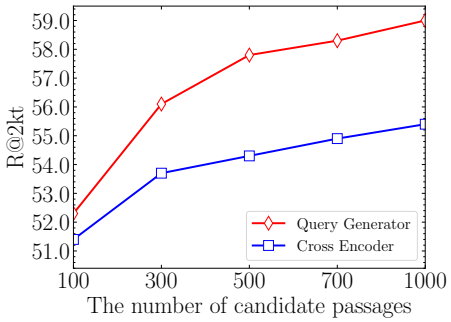


Figure 4: Re-ranking performance of cross-encoder and query generator on XOR-Retrieve dev set with different numbers of candidate passages.

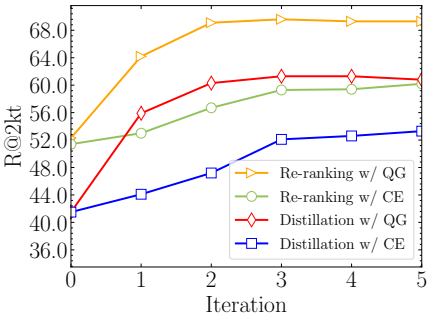


Figure 5: The changes of R@2kt during the iteratively training on XOR-Retrieve dev set. Here, "QG" denotes Query Generator and "CE" denotes Cross Encoder.

Visualization of the Training Procedure. We visualize the performance changes of R@2kt during the training of both dual-encoder and query generator re-ranker which re-ranks the retrieved top-100 passages. We also incorporate a cross-encoder (initialized with XLM-R Large) to perform distillation and re-ranking for comparison. As shown in Figure 5, the R@2kt of all models gradually increases as the iteration increases. While the training advances closer to convergence, the improvement gradually slows down. In the end, the performance of the dual-encoder is improved by approximately 17%, and the performance of the query generator is

improved by approximately 20%. Finally, comparing the performance of the cross-encoder and the query generator, we can find that there are approximately 6% gaps for both teachers and students. It shows the effectiveness of our method.

6 Conclusion

In this paper, we showed that the cross-encoder performs poorly when there are not sufficient training samples which are hard to obtain in the cross-lingual setting. Then we proposed a novel method that utilizes the query generator to improve the dual-encoder. We firstly proposed to use a query generator as the teacher. After that, we proposed a novel alignment method for cross-lingual retrieval which does not need any parallel corpus. Extensive experimental results show that the proposed method outperforms the baselines and significantly improves the state-of-the-art performance. Currently, our method depends on training data in all target languages. As future work, we will investigate how to perform the proposed method for zero-shot cross-lingual dense retrieval.

7 Limitations

The limitations are summarized as follows.

- The method depends on training data in all target languages. Intuitively, the method can be directly applied to the zero-shot cross-lingual dense retrieval if we only take the passage as input for the query generator, but the query generator performs poorly in the zero-shot setting. As future work, novel pre-training tasks for cross-lingual generation can be considered.
- The method does not investigate how to effectively train the query generator for the re-ranking task, just directly applies the training method for the cross-encoder re-ranker. We believe the potential of query generators for re-ranking is strong and designing a special re-ranking training method for query generators such as token-level supervision may be interesting for future work.
- The method requires large GPU sources. The final model approximately costs 12 hours on 8 NVIDIA Tesla A100 GPUs. Although researchers who do not have enough GPU sources can use the "gradient accumulation" technique to reduce GPU memory consumption, they also need to pay more time.
- This work does not consider the inconsistency between different countries (*e.g.,* law and religion), which leads to inconsistent positive passages for synonymous queries in different languages (*e.g.,* the legal age of marriage varies from country to country). Because we find that most of the queries in XOR-Retrieve contain the target country such as *"Mikä on yleisin kissa laji Suomessa?" (translation: What is the most common cat breed in Finland?)*.

References

Akari Asai, Jungo Kasai, Jonathan H. Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021a. XOR QA: cross-lingual open-retrieval question answering. In *NAACL-HLT*, pages 547–564. Association for Computational Linguistics.

Akari Asai, Xinyan Yu, Jungo Kasai, and Hanna Hajishirzi. 2021b. One question answering model for many languages with cross-lingual dense passage retrieval. In *NeurIPS*, pages 7547–7560.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. In *NAACL-HLT*, pages 3576–3588. Association for Computational Linguistics.

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*. OpenReview.net.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*, pages 8440–8451. Association for Computational Linguistics.

Zhuyun Dai and Jamie Callan. 2020. Context-aware term weighting for first stage passage retrieval. In *SIGIR*, pages 1533–1536. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Cícero Nogueira dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Beyond [CLS] through ranking by generation. In *EMNLP (1)*, pages 1722–1727. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *ACL (1)*, pages 878–891. Association for Computational Linguistics.

Luyu Gao and Jamie Callan. 2021a. Condenser: a pre-training architecture for dense retrieval. In *EMNLP (1)*, pages 981–993. Association for Computational Linguistics.

Luyu Gao and Jamie Callan. 2021b. Unsupervised corpus aware language model pre-training for dense passage retrieval. *CoRR*, abs/2108.05540.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *CIKM*, pages 55–64. ACM.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Xiaomeng Hu, Shi Yu, Chenyan Xiong, Zhenghao Liu, Zhiyuan Liu, and Ge Yu. 2022. P^3 ranker: Mitigating the gaps between pre-training and ranking fine-tuning with prompt-based learning and pre-finetuning. *CoRR*, abs/2205.01886.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2018. Co-pacrr: A context-aware neural IR model for ad-hoc retrieval. In *WSDM*, pages 279–287. ACM.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *ICLR*. OpenReview.net.

Gautier Izacard and Edouard Grave. 2021. Distilling knowledge from reader to retriever for question answering. In *ICLR*. OpenReview.net.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In *SIGIR*, pages 39–48. ACM.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *ACL (1)*, pages 6086–6096. Association for Computational Linguistics.

Oleg Lesota, Navid Rekabsaz, Daniel Cohen, Klaus Antonius Grasserbauer, Carsten Eickhoff, and Markus Schedl. 2021. A modern perspective on query likelihood with deep generative retrieval models. In *ICTIR*, pages 185–195. ACM.

Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2020. PARADE: passage representation aggregation for document reranking. *CoRR*, abs/2008.09093.

Yulong Li, Martin Franz, Md. Arafat Sultan, Bhavani Iyer, Young-Suk Lee, and Avirup Sil. 2021. Learning cross-lingual IR from an english retriever. *CoRR*, abs/2112.08185.

Shayne Longpre, Yi Lu, and Joachim Daiber. 2020. MKQA: A linguistically diverse benchmark for multilingual open domain question answering. *CoRR*, abs/2007.15207.

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 *EMNLP (1)*, pages 2780–2791. Association for Computational Linguistics.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Trans. Assoc. Comput. Linguistics*, 9:329–345.

Niklas Muennighoff. 2022. SGPT: GPT sentence embeddings for semantic search. *CoRR*, abs/2202.08904.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pre-trained sequence-to-sequence model. In *EMNLP (Findings)*, volume EMNLP 2020 of *Findings of ACL*, pages 708–718. Association for Computational Linguistics.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to doctttttquery. *Online preprint*.

Rodrigo Frassetto Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

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 *NAACL-HLT*, pages 5835–5847. Association for Computational Linguistics.

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

Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021a. PAIR: leveraging passage-centric similarity relation for improving dense passage retrieval. In *ACL/IJCNLP (Findings)*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2173–2183. Association for Computational Linguistics.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021b. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. In *EMNLP (1)*, pages 2825–2835. Association for Computational Linguistics.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. *CoRR*, abs/2204.07496.

Nikita Sorokin, Dmitry Abulkhanov, Irina Piontkovskaya, and Valentin Malykh. 2022. Ask me anything in your native language. In *NAACL-HLT*, pages 395–406. Association for Computational Linguistics.

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. In *ACL/IJCNLP (1)*, pages 5054–5064. Association for Computational Linguistics.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

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*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP (Demos)*, pages 38–45. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *ICLR*. OpenReview.net.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *NAACL-HLT*, pages 483–498. Association for Computational Linguistics.

Huiyun Yang, Huadong Chen, Hao Zhou, and Lei Li. 2022. Enhancing cross-lingual transfer by manifold mixup. *CoRR*, abs/2205.04182.

Sohee Yang and Minjoon Seo. 2020. Is retriever merely an approximator of reader? *CoRR*, abs/2010.10999.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *SIGIR*, pages 1503–1512. ACM.

Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2021. Adversarial retriever-ranker for dense text retrieval. *CoRR*, abs/2110.03611.

Shunyu Zhang, Yaobo Liang, Ming Gong, Daxin Jiang, and Nan Duan. 2022. Multi-view document representation learning for open-domain dense retrieval. In *ACL (1)*, pages 5990–6000. Association for Computational Linguistics.

Bo Zheng, Li Dong, Shaohan Huang, Wenhui Wang, Zewen Chi, Saksham Singhal, Wanxiang Che, Ting Liu, Xia Song, and Furu Wei. 2021. Consistency regularization for cross-lingual fine-tuning. In *ACL/IJCNLP (1)*, pages 3403–3417. Association for Computational Linguistics.

Jiawei Zhou, Xiaoguang Li, Lifeng Shang, Lan Luo, Ke Zhan, Enrui Hu, Xinyu Zhang, Hao Jiang, Zhao Cao, Fan Yu, Xin Jiang, Qun Liu, and Lei Chen. 2022. Hyperlink-induced pre-training for passage retrieval in open-domain question answering. In *ACL (1)*, pages 7135–7146. Association for Computational Linguistics.

Shengyao Zhuang, Hang Li, and Guido Zuccon. 2021. Deep query likelihood model for information retrieval. In *ECIR (2)*, volume 12657 of *Lecture Notes in Computer Science*, pages 463–470. Springer.

Appendix

A Dataset

XOR-Retrieve. XOR-Retrieve dataset is a cross-lingual retrieval dataset which aims to retrieve relevant passages from the English corpus for non-English queries. XOR-Retrieve data contains queries in seven typologically diverse languages: Arabic (Ar), Bengali (Bn), Finnish (Fi), Japanese (Ja), Korean (Ko), Russian (Ru), and Telugu (Te). The statistics are presented in Table 6.

Table 6: Data statistics for XOR-Retrieve.

	Train	Dev	Test
Ar	2,574	350	137
Bn	2,582	312	128
Fi	2,088	360	530
Ja	2,288	296	449
Ko	2,469	299	646
Ru	1,941	366	235
Te	1,308	238	374
Corpus size	18,003,200		

MKQA. MKQA dataset is a translated dataset of 10,000 query-answer pairs from NQ to 26 different languages, and the dataset is only used for evaluation. In our experiments, since we measure the R@2kt score, we filter the samples which do not have span answers. Then, we get 6,620 parallel queries in each language. Finally, we directly evaluate the dual-encoder trained on XOR-Retrieve data and use the same corpus with XOR-Retrieve in the experiments. Note that Arabic (Ar), English (En), Finnish (Fi), Japanese (Ja), Korean (Ko), and Russian (Ru) are seen in the training stage and we only report the performance of the rest 20 languages. As a result, it has a total of 132,400 samples.

B Efficiency Report

We list the time cost of training and inference in Table 7 which is made with 8 NVIDIA A100 GPUs.

Table 7: Efficiency Report.

Training	Warm-up	3h
	Per Iteration of Dual-Encoder	1h
	Per Iteration of Generator	0.3h
	Index Refresh	0.35h
	Overall	11.5h
Inference	Build Index	0.35h
	Query Encoding	40ns
	Dense Retrieval	2ms

Table 8: Hyper-parameters.

	Parameters	Value
	Max Query Length	32
	Max Passage Length	128
Training Warm-up Dual-Encoder	Learning Rate	1e-5
	Batch Size	128
	Negative Size	255
	Optimizer	AdamW
	Scheduler	Linear
	Warmup Proportion	0.1
	Training Steps on NQ	18400
	Training Steps on XOR	2000
Training Warm-up Generator on QG	Learning Rate	1e-4
	Batch Size	64
	Optimizer	AdamW
	Scheduler	Linear
	Warmup Proportion	0.1
	Training Steps	5000
Training Warm-up Generator on Re-ranking	Learning Rate	1e-5
	Batch Size	32
	Negative Size	15
	Optimizer	AdamW
	Scheduler	Linear
	Warmup Proportion	0.1
	Training Steps	1000
Iteraively Training of Dual-Encoder	Learning Rate	1e-5
	Batch Size	64
	Candidate Size	32
	Optimizer	AdamW
	Scheduler	Linear
	Warmup Proportion	0.1
	Training Steps	3000
	Threshold T in Eq. (8)	0.3
	Coefficient α in Eq. (11)	0.5
	# of iterations	5
Iteraively Training of Generator	Learning Rate	1e-5
	Batch Size	32
	Negative Size	15
	Optimizer	AdamW
	Scheduler	Linear
	Warmup Proportion	0.1
	Training Steps	500
	# of iterations	5

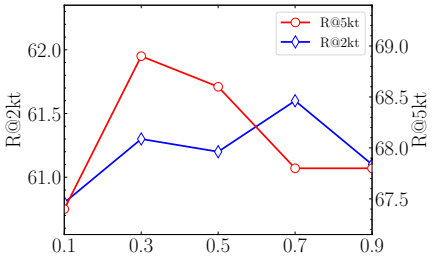
C Hyper-parameters

We present all hyper-parameters in Table 8.

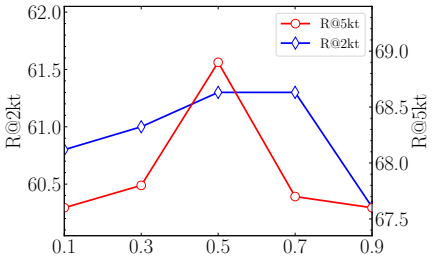
D Additional Experiments

D.1 Parameter Sensitivity

In this section, we tune the parameters of the proposed method to analyze parameter sensitivity. We vary both the threshold T (Eq. (8)) and the coefficient α (Eq. (11)) in the set $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. We report the tuning results with both R@2kt and R@5kt on the XOR-Retrieve dev set in Figure 6. As we can see, $T = 0.3$ and $\alpha = 0.5$ lead to the optimal R@5kt which is the ordering basis on



(a) Threshold (T).



(b) Coefficient (α).

Figure 6: Parameter sensitivity.

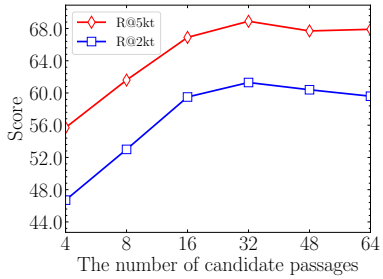


Figure 7: Effect of the number of candidate passages.

Table 9: Effect of model size.

	XLM-R Base		XLM-R Large	
	R@2kt	R@5kt	R@2kt	R@5kt
Ar	52.8	63.8	**64.4**	**72.2**
Bn	70.1	78.0	**77.0**	**80.9**
Fi	62.2	65.3	**67.0**	**70.1**
Ja	54.8	63.5	**56.4**	**67.2**
Ko	62.8	69.8	**67.4**	**73.3**
Ru	57.8	67.1	**66.7**	**72.2**
Te	70.6	74.8	**79.4**	**84.0**
Avg	61.3	68.9	**68.4**	**74.3**

the leaderboard.

In addition, we find that the optimal R@2kt and R@5kt are led by different thresholds T. Because the two metrics have different sensitivities to data quality, low-quality data is helpful to R@5kt but harmful to R@2kt. As a result, a small threshold T leads to more low-quality alignment training data and further leads to higher R@5kt but lower R@2kt. On the contrary, the optimal R@2kt and R@5kt are led by the same coefficient α.

Overall, our model is relatively stable when varying the two parameters, and consistently better than Sentri and Dr.DECR w/o KD$_{PC}$.

D.2 Effect of The Number of Candidates

Here, we investigate the effect of the number of candidates which is demonstrated to have a significant effect on the final performance. As shown in Figure 7, a large number of candidates leads to better performance. And when the number surpasses 32, the improvement gradually slows down. The results indicate that 32 candidates can better represent the whole corpus.

D.3 Effect of Model Size

Following Sentri (Sorokin et al., 2022), we employ a shared encoder (*i.e.,* the parameters of the query encoder and the passage encoder are the same) with large size as the dual-encoder and evaluate our method. As shown in Table 9, QuiCK with XLM-R Large achieves a significant performance improvement, which further demonstrates the effectiveness of the proposed QuiCK.

D.4 Effect of Scheduled Sampling

Previously, we demonstrate the effectiveness of the scheduled sampling by ablation study. Here, we present two generated samples to qualitatively analyze the scheduled sampling. As shown in Table 10, for the first case, the generated queries in other languages have the same semantics as the query from the source language. The sample is effective in alignment training and is helpful to achieve better performance. For the second case, the generated query in Finnish is relevant to the query from the source language but not synonymous. The sample is harmful to the model training. These samples indicate that the scheduled sampling is necessary for alignment training. In this way, we can reduce the impact of the cases which do not have the same semantics and further achieve better performance.

D.5 Effect of Span Answer

In our method, we employ the span answer to encourage the query generator to generate synonymous queries. Here, we conduct experiments to evaluate the effect of the span answer. Specially, we use another template: *"generate [language] query: [content]"* where we only need to fill two placeholders with the language of the target query and

Table 10: Two generated examples. The span answers are in bold.

Passage: Johanna Maria Magdalena "Magda" Goebbels (née Ritschel; **11 November 1901 – 1 May 1945**) was the wife of Nazi Germanyś Propaganda Minister Joseph Goebbels. A prominent member of the Nazi Party, she was a close ally, companion and political supporter of Adolf Hitler. Some historians refer to her as the unofficial "First Lady" of Nazi Germany, while others give that title to Emmy Göring.
Source Query (Ja): ヨハンナ・マリア・マクダレナ・ゲッベルスは何歳で死去した？ **Translation:** At what age did Johanna Maria McDalena Goebbels die?
Generated Query (Ru): В каком возрасте умерла Магда Геббельс? **Translation:** At what age did Magda Goebbels die?
Generated Query (Fi): Minä vuonna Magda Goebbels kuoli? **Translation:** In what year did Magda Goebbels die?
Passage: Charles V **(24 February 1500 – 21 September 1558)** was ruler of both the Holy Roman Empire from 1519 and the Spanish Empire (as Charles V of Spain) from 1516, as well as of the lands of the former Duchy of Burgundy from 1506. He stepped down from these and other positions by a series of abdications between 1554 and 1556. Through inheritance, he brought together under his rule extensive territories in western, central, and southern Europe, and the Spanish viceroyalties in the Americas and Asia.
Source Query (Ko): 신성 로마 제국 카를 5세 재위기간은 얼마나 되나요? **Translation:** How long was the reign of Charles V of the Holy Roman Empire?
Generated Query (Ru): Сколько лет правил Карл V? **Translation:** How many years did Charles V rule?
Generated Query (Fi): Minä vuonna Charles V hallitsi Rooman valtakuntaa? **Translation:** In what year did Charles V rule the Roman Empire?

Table 11: A generated example with different input templates. The span answer is in bold. Here, "QG" denotes query generator.

Passage: The Higgs boson is an elementary particle in the Standard Model of particle physics, produced by the quantum excitation of the Higgs field, one of the fields in particle physics theory. It is named after physicist **Peter Higgs**, who in 1964, along with five other scientists, proposed the mechanism which suggested the existence of such a particle. Its existence was confirmed in 2012 by the ATLAS and CMS collaborations based on collisions in the LHC at CERN.
Generated Query by QG w/ span answer (Fi): Kuka on kehittänyt Higgs-boson? **Translation:** Who developed the Higgs boson?
Generated Query by QG w/ span answer (Ko): 히그슨을 처음 발견한 사람은 누구인가? **Translation:** Who first discovered Higson?
Generated Query by QG w/o span answer (Fi): Milloin Higgs on löydetty? **Translation:** When was Higgs Found?
Generated Query by QG w/o span answer (Ko): Кто был первым исследователем физики Higgs? **Translation:** Who was Higgs' first physics researcher?

the passage content. We also incorporate the cross-encoder for comparison. We use the re-rankers to re-rank the retrieved results of the warm-up dual-encoder initialized with XLM-R. Note that introducing the span answer into the cross-encoder makes the re-ranking task easier, because the cross-encoder only needs to check whether the passage contains the span answer. The scores of this cross-encoder almost degenerate into hard labels and it is difficult to effectively train the dual-encoder by distilling knowledge from this cross-encoder.

We show the results in Table 12. Based on these results, we have the following findings. On the one hand, the query generator trained with span answers is better than the query generator without span answers. It shows that taking span answers as input leads to better performance on re-ranking tasks for the query generator. On the other hand, both the two query generator is better than the cross-encoder when re-ranking top-1000 retrieved passages, it shows the effectiveness of the query generator in the cross-lingual setting.

In addition, we show queries generated by the two query generators in Table 11. As we can see, for the query generator that does not take the span answer as input, the generated queries can be answered by the passage, but they focus on different segments of the passage and they are not synonymous. On contrary, for the query generator that takes the span answer as input, generated queries can be answered by the passage and they are synonymous. It shows that taking the span answer as input can effectively encourage the generator to generate synonymous queries.

E Detailed Results

Due to the limited space, we only present average performance for some experiments in Section 5. Here, we present the detailed performance in all languages of these experiments. Firstly, we present the detailed performance of all methods on the MKQA test set in Table 13. Secondly, we present the detailed performance of ablation results in Table 14. Finally, we present the detailed performance for evaluating the effect of alignment in Table 15.

Table 12: Performance comparison of different re-rankers on XOR-Retrieve dev set. The best results are in bold. Here, "QG" denotes query generator.

Methods	R@2kt								R@5kt							
	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg
Dual-Encoder	35.3	43.1	50.3	35.7	44.6	31.2	50.0	41.5	49.5	54.9	59.2	45.2	55.1	31.2	63.4	53.4
Re-ranking top-100 retrieved passages																
QG w/ Answer	**51.1**	**57.2**	53.5	**43.2**	**55.1**	**43.9**	61.8	**52.3**	**56.6**	60.5	60.5	**51.0**	**60.7**	46.4	67.6	**57.6**
QG w/o Answer	48.9	55.6	54.1	41.9	53.3	43.5	61.8	51.3	**56.6**	**60.9**	59.9	49.8	59.3	46.8	**68.1**	57.3
Cross-Encoder	49.2	53.6	**57.6**	41.1	54.0	41.4	**63.0**	51.4	55.3	59.9	**62.1**	49.8	60.4	**47.3**	66.4	57.3
Re-ranking top-1000 retrieved passages																
QG w/ Answer	**53.7**	**66.1**	**56.7**	**52.3**	**59.3**	**56.1**	**68.9**	**59.0**	**61.2**	**71.1**	62.1	**58.1**	65.6	**61.6**	**74.4**	**64.9**
QG w/o Answer	52.4	62.8	56.1	49.0	58.2	55.7	64.3	56.9	59.9	70.7	62.1	57.3	64.9	59.9	73.5	64.0
Cross-Encoder	50.8	58.2	55.1	45.2	**59.3**	50.6	65.5	55.0	**61.2**	67.4	**63.4**	53.5	**66.3**	56.1	73.9	63.1

Table 13: Detailed performance on MKQA test set. "$*$" denotes that the results are copied from the Sentri paper.

Methods	Da	De	Es	Fr	He	Hu	It	Km	Ms	Nl
CORA*	44.5	44.6	45.3	44.8	27.3	39.1	44.2	22.2	44.3	47.3
BM25 + MT*	44.1	43.3	44.9	42.5	36.9	39.3	40.1	31.3	42.5	46.5
Sentri*	57.6	56.5	55.9	55.1	47.9	51.8	54.3	43.9	56.0	56.3
w/ Bi-Encoder*	50.0	47.8	48.7	47.4	37.7	43.4	41.8	37.8	49.5	47.3
QuiCK	58.3	56.4	55.2	55.5	44.7	52.4	52.3	42.0	56.9	57.5
QuiCK w/ LaBSE	**63.3**	**61.8**	**62.2**	**62.4**	**56.1**	**58.9**	**60.6**	**53.0**	**64.2**	**63.0**

No	Pl	Pt	Sv	Th	Tr	Vi	Zh-cn	Zh-hk	Zh-tw	Avg
48.3	44.8	40.8	43.6	45.0	34.8	33.9	33.5	41.5	41.0	41.1
43.3	46.5	45.7	49.7	46.5	42.5	43.5	37.5	37.5	36.1	42.0
56.5	55.8	54.8	56.9	55.3	53.0	54.4	50.2	50.7	49.4	53.3
49.1	47.0	47.7	50.0	46.5	45.6	47.3	42.6	41.5	41.0	45.3
57.0	54.9	54.7	58.0	55.7	53.9	54.9	50.4	49.3	48.9	53.4
62.8	**62.0**	**61.5**	**63.3**	**60.5**	**60.6**	**61.8**	**57.3**	**56.3**	**56.0**	**60.3**

Table 14: Detailed performance for ablation study on XOR-Retrieve dev set.

	R@2kt								R@5kt							
	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg
QuiCK	52.8	**70.1**	60.2	**54.8**	**62.8**	**57.8**	**70.6**	**61.3**	**63.8**	**78.0**	**65.3**	**63.5**	**69.8**	**67.1**	74.8	**68.9**
w/o Sampling	**53.1**	68.4	60.2	50.2	61.1	55.7	68.1	59.5	63.4	**78.0**	64.6	60.6	68.1	63.7	74.4	67.5
w/o Alignment	51.8	68.1	**60.8**	51.0	60.4	57.4	70.2	59.9	61.5	74.7	64.6	62.7	68.8	62.4	**75.2**	67.1
w/o Generation	55.0	68.1	59.6	47.3	60.7	54.9	66.4	58.8	62.8	74.0	65.0	56.4	66.3	62.0	74.8	65.9
w/o All	35.3	43.1	50.3	35.7	44.6	31.2	50.0	41.5	49.5	54.9	59.2	45.2	55.1	31.2	63.4	53.4

Table 15: Detailed performance of alignment based on different pre-trained languages models.

	R@2kt								R@5kt							
	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg	Ar	Bn	Fi	Ja	Ko	Ru	Te	Avg
XLM-R	**52.8**	**70.1**	60.2	**54.8**	**62.8**	**57.8**	**70.6**	**61.3**	**63.8**	**78.0**	**65.3**	**63.5**	**69.8**	**67.1**	74.8	**68.9**
w/o Alignment	51.8	68.1	**60.8**	51.0	60.4	57.4	70.2	59.9	61.5	74.7	64.6	62.7	68.8	62.4	**75.2**	67.1
LaBSE	**67.3**	**78.9**	**65.9**	**59.8**	66.3	**63.7**	**80.7**	**68.9**	72.2	**83.2**	**69.7**	**68.0**	70.9	**71.7**	**84.9**	**74.4**
w/o Alignment	65.7	78.3	65.0	58.9	**67.0**	62.9	77.3	67.9	**72.5**	80.9	**69.7**	66.8	**71.9**	70.5	84.5	73.8