ADVERSARIAL RETRIEVER-RANKER FOR DENSE TEXT RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

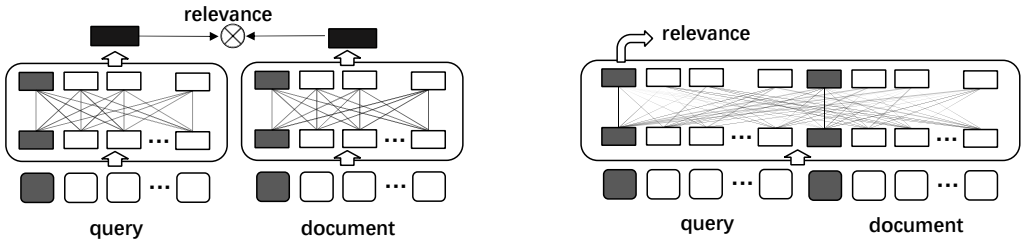
Current dense text retrieval models face two typical challenges. First, it adopts a siamese dual-encoder architecture to encode query and document independently for fast indexing and searching, whereas neglecting the finer-grained term-wise interactions. This results in a sub-optimal recall performance. Second, it highly relies on a negative sampling technique to build up the negative documents in its contrastive loss. To address these challenges, we present ***A**dversarial **R**etriever-**R**anker* (AR2), which consists of a dual-encoder retriever plus a cross-encoder ranker. The two models are jointly optimized according to a minimax adversarial objective: the retriever learns to retrieve negative documents to cheat the ranker, while the ranker learns to rank a collection of candidates including both the ground-truth and the retrieved ones, as well as providing progressive direct feedback to the dual-encoder retriever. Through this adversarial game, the retriever gradually produces harder negative documents to train a better ranker, whereas the cross-encoder ranker provides progressive feedback to improve retriever. We evaluate AR2 on three benchmarks. Experimental results show that AR2 consistently and significantly outperforms existing dense retriever methods and achieves new state-of-the-art results on all of them. This includes the improvements on Natural Questions R@5 to 77.9% ($+2.1\%$), TriviaQA R@5 to 78.2% ($+1.4\%$), and MS-MARCO MRR@10 to 39.5% ($+1.3\%$). We will make our code, models, and data publicly available.

1 INTRODUCTION

Dense text retrieval (Lee et al., 2019; Karpukhin et al., 2020b) has achieved great successes in a wide variety of both research and industrial areas, such as search engine (Brickley et al., 2019; Shen et al., 2014), recommendation system (Hu et al., 2020), open-domain question answering (Chen et al., 2017; Liu et al., 2020), etc. A typical dense retrieval model adopts a dual-encoder (Huang et al., 2013) architecture to encode queries and documents into low-dimensional embedding vectors, and the relevance between query and document is measured by the similarity between embeddings. In the real-world dense text retrieval applications, it pre-computes all the embedding vectors of documents in the corpus, and leverages the approximate nearest neighbor (ANN) (Johnson et al., 2019) technique for efficiency. To train a dense retriever, contrastive loss with negative samples is widely applied in the literature. During training, it utilizes a negative sampling method to obtain negative documents for a given query-document pair, and then minimizes the contrastive loss which relies on both the positive document and the sampled negative ones (Shen et al., 2014; Chen et al., 2017; Radford et al., 2021).

Recent studies on contrastive learning (Xiong et al., 2020; Karpukhin et al., 2020a) show that the iterative "hard-negative" sampling technique can significantly improve the performance compared with "random-negative" sampling approach, as it can pick more representative negative samples to learn a more discriminative retriever. In the work (Qu et al., 2021), it suggests to leverage cross-encoder model to heuristically filter "hard-negative" samples to further improve performance and shows the importance of sampling technique in the contrastive learning.

On the other hand, the model architecture of dual-encoders enables the encoding of queries and documents independently which is essential for document indexing and fast retrieval. However, this



(a) Retriever

(b) Ranker

Figure 1: Illustration of two modules in AR2. (a) Retriever: query and document are encoded independently by a dual-encoder. (b) Ranker: concatenated, jointly encoded by a cross-encoder.

ignores the modeling of finer-grained interactions between query and document which could be a sub-optimal solution in terms of retrieval accuracy.

Motivated by these phenomenons, we propose an ***A**dversarial **R**etriever-**R**anker* (AR2) framework. The intuitive idea of AR2 is inspired by the "retriever-ranker" architecture in the classical information retrieval systems. AR2 consists of two modules: a dual-encoder model served as the retrieval module in Figure 1a and a cross-encoder model served as the ranker module in Figure 1b. The cross-encoder model takes the concatenation of query and document as input text, and could generate more accurate relevance scores compared with the dual-encoder model, since it can fully explore the interactions between query and document through a self-attention mechanism using a conventional transformer model (Vaswani et al., 2017). Instead of training "retriever-ranker" modules independently in some IR systems (Manning et al., 2008; Mitra & Craswell, 2017), AR2 constructs a unified minimax game for training the retriever and ranker models interactively, as shown in Figure 2.

In particular, it adopts a minimax objective from the adversarial game (Goodfellow et al., 2014) where the retrieval model is optimized to produce relevant documents to fool the ranker model, whereas the ranker model learns to distinguish the ground-truth relevant document and retrieved ones by its opponent retrieval model. Within the adversarial "retriever-ranker" training framework, the retrieval model receives the smooth training signals from the ranker model which helps alleviate the harmful effects of "false-negative" issues. For example, a "false-negative" example which is rated as high-relevance by the ranker model, will also be granted with high probability by retrieval model in order to fool the ranker, meanwhile the ranker model with better generalization capability is more resistant to label noises compared with the retrieval model.

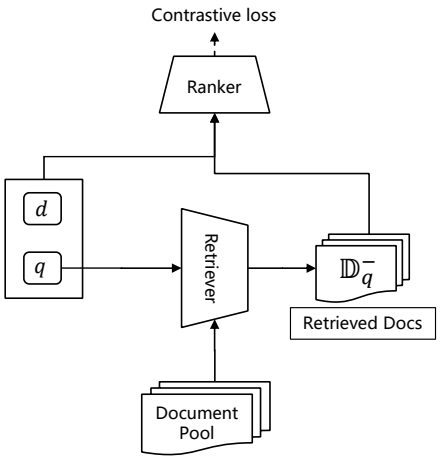


Figure 2: Illustration of AR2 training pipeline. q, d, and $\mathbb{D}_q^-$ represent query, positive document, and retrieved documents, respectively.

In the empirical studies of AR2, we further introduce a distillation regularization approach to help stabilize/improve the training of the retriever. Intuitively, the retriever would be converged to a sharp conditional-probabilities over documents given a query within the adversarial training framework, i.e., high retrieval probabilities for the top relevant documents and near-zero retrieval ones for the rest. However, it is not a desirable property as it might impede exploring different documents during training. Thus, we incorporate the distillation loss between the retriever and ranker models as a smooth term for further improvement.

In experiments, we evaluate AR2 on three widely-used benchmarks for dense text retrieval: Natural Questions, Trivia QA and MS-MARCO. Experimental results show that AR2 achieves state-of-the-art performance on all these datasets. Meanwhile, we provide a comprehensive ablation study to demonstrate the advantage of different AR2 components.

2 PRELIMINARIES

Dense Text Retrieval: we mainly consider a contrastive-learning paradigm for dense text retrieval in this work, where the training set consists of a collection of text pairs. $C = \{(q_1, d_1), ..., (q_n, d_n)\}$. In the scenario of open-domain question answering, a text pair (q, d) refers to a question and a corresponding document which contains the answer. A typical dense retrieval model adopts a dual encoder architecture, where questions and documents are represented as dense vectors separately and the relevance score $s_\theta(q, d)$ between them is measured by the similarity between their embeddings:

$$s_\theta(q, d) = \langle E(q; \theta), E(d; \theta)) \rangle \tag{1}$$

where $E(\cdot; \theta)$ denotes the encoder module parameterized with θ, and $\langle \cdot \rangle$ is the similarity function, e.g., inner-product, euclidean distance, etc. Based on the embeddings, existing solutions generally leverage on-the-shelf fast ANN-search (Johnson et al., 2019) for efficiency.

A conventional contrastive-learning algorithm could be applied for training the dual encoders (Shen et al., 2014; Chen et al., 2017; Liu et al., 2020). For example, given a training instance (q, d), we select n negative irrelevant documents $(d_1^-, ..., d_n^-)$ (denoted as $\mathbb{D}_q^-$) to optimize the loss function of the negative log likelihood of the positive document:

$$L_\theta(q, d, \mathbb{D}_q^-) = -\log \frac{e^{\tau s_\theta(q,d)}}{e^{\tau s_\theta(q,d)} + \sum_{i=1}^{n} e^{\tau s_\theta(q,d_i^-)}} \tag{2}$$

where τ is a hyper-parameter to control the temperature. In the previous works (Shen et al., 2014; Chen et al., 2017; Liu et al., 2020), it presents an effective strategy on negative document sampling, called "In-Batch Negatives" where negative documents are randomly sampled from a collection of documents which are within the same mini-batch as question-document training pairs.

Recently, some studies e.g., ANCE (Xiong et al., 2021), Condenser (Gao & Callan, 2021b) have shown that selecting "hard-negatives" in the training can significantly improve the retrieval performance in open-domain question answering. For example, instead of sampling negative document randomly, "hard-negatives" are retrieved through the last checkpoint of dual encoder model. However, a more recent work RocketQA (Qu et al., 2021) continues to point out that the retrieved "hard-negatives" could potential be "false-negatives" in some cases, which might limit the performance.

Generative Adversarial Network: GANs have been widely studied for generating the realistic-looking images in computation vision (Goodfellow et al., 2014; Brock et al., 2018). In the past years, the idea of GANs has been applied in information retrieval (Wang et al., 2017). For example, IRGAN (Wang et al., 2017), proposes an minimax retrieval framework which constructs two types of IR models: a generative retrieval model and a discriminative retrieval model. The two IR models are optimized through a minimax game: the generative retrieval model generates relevant documents that look like ground-truth relevant documents to fool the discriminative retrieval model, whereas the discriminative retrieval model learns to draw a clear distinction between the ground-truth relevant documents and the generated ones made by its opponent generative retrieval model. The minimax objective is formulated as:

$$J^{G^*, D^*} = \min_\theta \max_\phi \mathrm{E}_{d \sim p_{\text{true}}(\cdot|q)} \left[\log D_\phi(d, q)\right] + \mathrm{E}_{d^- \sim G_\theta(\cdot|q)} \left[\log\big(1 - D_\phi(d^-, q)\big)\right] \tag{3}$$

where $G_\theta(\cdot|q)$ and $D_\phi(d^-, q)$ denote the generative retrieval model and discriminative retrieval model in IRGAN, respectively. It is worth noting the original IRGAN model doesn't work for dense text retrieval tasks as it doesn't contain the dual-encoder model for document indexing or fast retrieval.

3 METHOD

In this section, we introduce the proposed adversarial retriever-ranker (**AR2**) approach. It consists of two modules: the dual-encoder retriever module G_θ as in Figure 1a, and the cross-encoder ranker module D_ϕ as in Figure 1b. G_θ and D_ϕ computes the relevance score between question and document as follows:

$$\begin{aligned} G_\theta(q, d) &= E_\theta(q)^T E_\theta(d) \\ D_\phi(q, d) &= \mathbf{w}_\phi{}^T E_\phi\left([q, d]\right) \end{aligned} \tag{4}$$

where $E_\theta(\cdot)$ and $E_\phi(\cdot)$ are language model encoders which can be initialized with any pre-trained language models, $\mathbf{w}_\phi$ is the linear projector in D_ϕ and $[q, d]$ is the concatenation of question and document.

In **AR2**, the retriever and ranker modules are optimized jointly through a contrastive minimax objective:

$$J^{G^*,D^*} = \min_\theta \max_\phi \mathbf{E}_{\mathbb{D}_q^- \sim G_\theta(q,\cdot)} \left[\log p_\phi(d|q, d, \mathbb{D}_q^-)\right] \tag{5}$$

where $\mathbb{D}_q^-$:$\{d_i^-\}_{i=1}^n$ is the set of n negative documents sampled by $G_\theta(q, \cdot)$ given q, and $p_\phi(d|q, d, \mathbb{D}_q^-)$ denotes the probability of selecting the ground-truth document d from the $n + 1$ documents: $\{d\} \cup \mathbb{D}_q^-$ by the ranker module D_ϕ;

$$p_\phi(d|q, d, \mathbb{D}_q^-) = \frac{e^{\tau D_\phi(q,d)}}{e^{\tau D_\phi(q,d)} + \sum_{i=1}^n e^{\tau D_\phi(q,d_i^-)}} \tag{6}$$

According to the objective function (Eqn. 5), the dual-encoder retrieval model $G_\theta(q, \cdot)$ would try to sample the high-relevant documents to fool the ranker model, whereas the ranker model $D_\phi(q, \cdot)$ is optimized to draw distinction between ground truth passage and the ones sampled by $G_\theta(q, \cdot)$. We present the illustration of the AR2 framework in Figure 2. In order to optimize the minimax objective function, we adopt a conventional iterative-learning mechanism to optimize the retriever and ranker modules coordinately.

3.1 Training the Ranker D_ϕ

Given the fixed retriever G_θ, the ranker model D_ϕ is updated by maximizing the log likelihood of selecting ground-truth d from $\{d\} \cup \mathbb{D}_q^-$ given a query q:

$$\phi^* = \text{argmax}_\phi \log p_\phi(d|q, d, \mathbb{D}_q^-) \tag{7}$$

$\mathbb{D}_q^-$ is the document set with size n sampled by G_θ according to Eqn. 5. In experiments, we first retrieve top-100 negative documents, and then randomly sample n examples from them to obtain $\mathbb{D}_q^-$.

3.2 Training Retriever G_θ

With fixing the ranker D_ϕ, the model parameters θ^* for retriever G_θ is optimized by minimizing the expectation of log likelihood of function. In particular, by isolating θ from the minimax function (Eqn. 5), the objective for the retriever can be written as:

$$\theta^* = \text{argmin}_\theta J^\theta = \mathbf{E}_{\mathbb{D}_q^- \sim G_\theta(q,\cdot)} \left[\log p_\phi(d|q, d, \mathbb{D}_q^-)\right] \tag{8}$$

However, it is intractable to optimize θ directly through Eqn. 8, as the computation of probability $\mathbb{D}_q^- \sim G_\theta(q, \cdot)$ is not following a close form. Thus, we seek to minimize an alternative upper-bound of the loss criteria:

$$J^\theta \leq \hat{J}^\theta = \mathbf{E}_{d_i^- \sim p_\theta(d_i^-|q,\mathbb{D}_q^-)} \left[\log p_\phi(d|q, d, d_i^-)\right] \tag{9}$$

The detailed deviation of Eqn. 9 is provided in the Appendix A.1. Therefore, the gradient of parameter θ can be computed as the derivative of $\hat{J}^\theta$ with respect to θ:

$$\nabla_\theta \hat{J}^\theta = \mathbf{E}_{d_i^- \sim p_\theta(d_i^-|q,\mathbb{D}_q^-)} \nabla_\theta \log p_\theta(d_i^-|q, \mathbb{D}_q^-) \left[\log p_\phi(d|q, d, d_i^-)\right] \tag{10}$$

Here, the same approach is applied to obtain set $\mathbb{D}_q^-$ as in Eqn. 7.

Regularization: we further introduce a distillation regularization term in G_θ's training, which encourages the retriever model to distill from the ranker model.

$$J_\mathcal{R}^\theta = H(p_\phi(\cdot|q, \mathbb{D}), p_\theta(\cdot|q, \mathbb{D})) \tag{11}$$

$H(\cdot)$ is the cross entropy function. $p_\phi(\cdot|q, \mathbb{D})$ and $p_\theta(\cdot|q, \mathbb{D})$ denotes the conditional probabilities of document in the whole corpus $\mathbb{D}$ by the ranker and the retriever model, respectively. In practice, we also limit the sampling space over documents be a fixed set, i.e., $\{d\} \cup \mathbb{D}_q^-$. Thus the regularization loss for the retriever model can be rewritten as:

$$J_\mathcal{R}^\theta = H(p_\phi(\cdot|q, d, \mathbb{D}_q^-), p_\theta(\cdot|q, d, \mathbb{D}_q^-)) \tag{12}$$

Algorithm 1 Adversarial Retriever-Ranker (AR2)

Require: Retriever G_θ; Ranker D_ϕ; Document pool $\mathbb{D}$; Training dataset C.
1: Initialize the retriever G_θ and the ranker D_ϕ with pre-trained language models.
2: Train the warm-up retriever G_θ^0 on training dataset C.
3: Build ANN index on $\mathbb{D}$
4: Retrieve negative samples on $\mathbb{D}$.
5: Train the warm-up ranker D_θ^0
6: **while** AR2 has not converged **do**
7: **for** Retriever training step **do**
8: Sample n documents $\{d_i^-\}_n$ from ANN index.
9: Update parameters of the retriever G_θ.
10: **end for**
11: Refresh ANN Index.
12: **for** Ranker training step **do**
13: Sample n hard negatives $\{d_i^-\}_n$ from ANN index.
14: Update parameters of the ranker D_ϕ.
15: **end for**
16: **end while**

3.3 Index refresh

During each training iteration of retriever and ranker models in AR2, we refresh the document index to keep the retrieved document set updated. To build the document index, we take the document encoder from the retrieval model to compute the embeddings $E(d;\theta)$ for every document d from the corpus: $d \in C$, and then build the inner-product based ANN search index with FAISS tool (Johnson et al., 2019).

In summary, Algorithm 1 shows the full implementation details of the proposed AR2.

4 Experiments

4.1 Datasets:

We conduct experiments on three popular benchmarks: Natural Questions (Kwiatkowski et al., 2019), Trivia QA (Joshi et al., 2017), and MS-MARCO Passage Ranking (Nguyen et al., 2016). Their detailed statistics are presented in Table 1.

Table 1: The statistics of Natural Questions, Trivia QA , and MS-MARCO. Following DPR (Karpukhin et al., 2020b), we filter the training samples without gold documents.

Datasets	Train(Original)	Train(Processed)	Dev	Test	Documents
Natural Questions	79,168	58,880	8,757	3,610	21,015,324
Trivia QA	78,785	60,413	8,837	11,313	21,015,324
MS-MARCO	502,939	502,939	6,980	6,837	8,841,823

Natural Questions (NQ) collects real questions from Google search engine and each question is paired with an answer span and golden passages in Wikipedia pages. In NQ, the goal of the retrieval stage is to find positive passages from a large passage pool. We report Recall of top-k (R@k), which represents the proportion of top k retrieved passages that contain the answers.

Trivia QA is a reading comprehension corpus authored by trivia enthusiasts. Each sample is a $\langle question, answer, evidence\rangle$ triple. In the retrieval stage, the goal is to find passages that contains the answer. We also use Recall of top-k as the evaluation metric for Trivia QA.

MS-MARCO Passage Ranking is widely used in information retrieval. It collects real questions from Bing search engine. Each question is paired with several web documents. Following previous works (Ren et al., 2021; Qu et al., 2021), we report MRR@10, R@50, R@1k on the dev set. Mean

Table 2: The comparison of retrieval performance on Natural Questions test set, Trivia QA test set, and MS-MARCO dev set. The results of the first two blocks are from published papers. If the results are not provided, we mark them as "-".

	Natural Questions			**Trivia QA**			**MS-MARCO**		
	R@5	R@20	R@100	R@5	R@20	R@100	MRR@10	R@50	R@1k
BM255 (Yang et al., 2017)	-	59.1	73.7	-	66.9	76.7	18.7	59.2	85.7
GAR (Mao et al., 2021a)	60.9	74.4	85.3	73.1	80.4	85.7	-	-	-
doc2query (Nogueira et al., 2019b)	-	-	-	-	-	-	21.5	64.4	89.1
DeepCT (Dai & Callan, 2019)	-	-	-	-	-	-	24.3	69.0	91.0
docTTTTTquery (Nogueira et al., 2019a)	-	-	-	-	-	-	27.7	75.6	94.7
DPR (Karpukhin et al., 2020b)	-	78.4	85.3	-	79.3	84.9	-	-	-
ANCE (Xiong et al., 2021)	-	81.9	87.5	-	80.3	85.3	33.0	-	95.9
RDR (Yang & Seo, 2020)	-	82.8	88.2	-	82.5	87.3	-	-	-
ColBERT (Khattab & Zaharia, 2020)	-	-	-	-	-	-	36.0	82.9	96.8
RocketQA (Qu et al., 2021)	74.0	82.7	88.5	-	-	-	37.0	85.5	97.9
COIL (Gao et al., 2021)	-	-	-	-	-	-	35.5	-	96.3
ME-BERT (Luan et al., 2021)	-	-	-	-	-	-	33.8	-	-
Joint Top-k (Sachan et al., 2021a)	72.1	81.8	87.8	74.1	81.3	86.3	-	-	-
Individual Top-k (Sachan et al., 2021a)	75.0	84.0	89.2	76.8	83.1	87.0	-	-	-
PAIR (Ren et al., 2021)	74.9	83.5	89.1	-	-	-	37.9	86.4	98.2
DPR-PAQ (Oğuz et al., 2021)									
-BERT$_{base}$	74.5	83.7	88.6	-	-	-	31.4	-	-
-RoBERTa$_{base}$	74.2	84.0	89.2	-	-	-	31.1	-	-
Condenser (Gao & Callan, 2021b)	-	83.2	88.4	-	81.9	86.2	36.6	-	97.4
coCondenser (Gao & Callan, 2021a)	75.8	84.3	89.0	76.8	83.2	87.3	38.2	-	98.4
AR2-G^0	69.7	80.8	87.1	74.4	81.7	86.6	34.8	84.2	98.0
AR2-G	**77.9**	**86.0**	**90.1**	**78.2**	**84.4**	**87.9**	**39.5**	**87.8**	**98.6**

Reciprocal Rank(MRR) is the mean of Reciprocal Rank(RR) across questions, calculated as the reciprocal of the rank where the first relevant document was retrieved.

4.2 Implementation Details

First step, we follow the experiments in (Sachan et al., 2021b; Gao & Callan, 2021a) to continuous pre-training the ERNIE-2.0-base model (Sun et al., 2020) with Inverse Cloze Task (ICT) training (Sachan et al., 2021b) for NQ and TriviaQA datasets, and coCondenser training (Gao & Callan, 2021a) for MS-MARCO dataset.

Second step, we follow the experiment settings of DPR (Karpukhin et al., 2020b) to train a warm-up dual-encoder retrieval model $\mathbf{G^0}$. It is initialized with the continuous pretrained ERNIE-2.0-based model as we obtained in step one. Then we train a warm-up cross-encoder model $\mathbf{D^0}$ initialized with the ERNIE-2.0-Large. $\mathbf{D^0}$ learns to rank the Top-k documents selected by $\mathbf{G^0}$ with contrastive learning. The detailed hyper-parameters in training are listed in Appendix A.3.

Third step, we iteratively train the ranker (AR2-$\mathbf{D}$) model initialized with ERNIE-2.0-large and retriever (AR2-$\mathbf{G}$) initialized with $\mathbf{G}^0$ according to Algorithm 1. The number of training iterations is set to 10. During each iteration of training, the retriever model is scheduled to train with 1500 mini-batches, while the ranker model is scheduled to train with 500 mini-batches. The document index is refreshed after each iteration of training. The other detailed hyper-parameters are shown in Appendix A.3.

All the experiments in this work run on 8 NVIDIA Tesla A100 GPUs. The implementation code of AR2 is based on Huggingface Transformers (Wolf et al., 2020) with utilizing gradient checkpointing (Chen et al., 2016), Apex[1], and gradient accumulation to reduce GPU memory consumption.

4.3 Results

Performance of Retriever AR2-G: The comparison of retrieval performance on NQ, Trivia QA, and MS-MARCO are presented in Table 2.

We compare AR2-$\mathbf{G}$ with previous state-of-the-art methods, including sparse and dense retrieval models. The top block shows the performance of sparse retrieval methods. BM25 (Yang et al., 2017)

[1] https://github.com/NVIDIA/apex

Table 3: Performance of rankers before and after AR2 training on NQ test set.

Retriever	Ranker	R@1	R@5	R@10
AR2-$\mathbf{G^0}$	-	48.3	69.7	76.2
	AR2-$\mathbf{D^0}$	60.6	78.7	82.6
	AR2-$\mathbf{D}$	64.2	79.0	82.6
AR2-$\mathbf{G}$	-	58.7	77.9	82.5
	AR2-$\mathbf{D^0}$	61.1	80.1	84.3
	AR2-$\mathbf{D}$	65.6	81.5	84.9

Table 4: Performance of AR2-$\mathbf{G}$ on NQ test set with different negative sample size n.

	R@1	R@5	R@20	R@100	Latency
n=1	56.3	76.4	85.3	89.7	210ms
n=5	57.8	76.9	85.3	89.7	330ms
n=7	58.0	77.2	85.2	89.7	396ms
n=11	58.0	77.1	85.4	89.8	510ms
n=15	57.8	77.3	85.6	90.1	630ms

Table 5: Comparison of AR2 and IRGAN

	R@1	R@5	R@20	R@100
AR2	58.7	77.9	86.0	90.1
IRGAN	55.2	75.2	84.5	89.2

Table 6: Effect of regularization in AR2

	R@1	R@5	R@20	R@100	Entropy
AR2-$\mathbf{G}$	58.7	77.9	86.0	90.1	2.10
– w/o R	57.8	77.3	85.6	90.1	1.70

is a traditional sparse retriever based on the exact term matching. DeepCT (Dai & Callan, 2019) uses BERT to dynamically generate lexical weights to augment BM25 Systems. doc2Query (Nogueira et al., 2019b), docTTTTTQuery (Nogueira et al., 2019a), and GAR (Mao et al., 2021a) use text generation to expand queries or documents to make better use of BM25. The middle block lists the results of strong dense retrieval methods, including DPR (Karpukhin et al., 2020a), ANCE (Xiong et al., 2021), RDR (Yang & Seo, 2020), RocketQA (Qu et al., 2021), Joint and Individual Top-k (Sachan et al., 2021b), PAIR (Ren et al., 2021), DPR-PAQ (Oğuz et al., 2021), Condenser (Gao & Callan, 2021b). coCondenser (Gao & Callan, 2021a), ME-BERT (Luan et al., 2021), CoIL (Gao et al., 2021). These methods improve the performance of dense retrieval by constructing hard negative samples, jointly training the retriever and downstream tasks, pre-training, knowledge distillation, and muti-vector representations.

The bottom block in Table 2 shows the results of proposed AR2 models. AR2-$\mathbf{G^0}$ refers to the warm-up retrieval model in AR2 (details can be found in section 4.2) which leverages the existing continuous pre-training technique for dense text retrieval tasks. i.e., It shows better performance compared with DPR (Karpukhin et al., 2020a) and ANCE (Xiong et al., 2020), etc approaches that don't adopt the continuous pre-training procedure. We also observed that AR2-$\mathbf{G}$: the retrieval model trained with the adversary framework, significantly outperforms the warm-up AR2-$\mathbf{G^0}$ model, and achieves new state-of-the-art performance on all three datasets.

4.4 Analysis

In this section, we conduct a set of detailed experiments on analyzing the proposed AR2 training framework to helps understand its pros and cons.

Performance of Ranker AR2-$\mathbf{D}$: To evaluate the performance of ranker AR2-$\mathbf{D}$ on NQ, we first retrieve top-100 documents for each query in the test set with the help of dual-encoder AR2-$\mathbf{G}$ model, and then re-rank them with the scores produced by the AR2-$\mathbf{D}$ model. The results are shown in Table 3. "-" represents without ranker. AR2-D^0 refers to the warm-up ranker model in AR2. The results show that the ranker obtains better performance compared with only using retriever. It suggests that we could use a two-stage ranking strategy to further boost the retrieval performance. Comparing the results of AR2-D and AR2-D^0. We further find that the ranker AR2-$\mathbf{D}$ gets a significant gain with adversarial training.

Impact of Negative Sample Size: In the training of AR2, the number of negative documents n would affect the model performance and training time as well. In Table 4, we show the performance and the training latency per batch with different negative sample size n. In this setting, we evaluate AR2 without the regularization term. We observe the improvement over R@1 and R@5 by increasing n from 1 to 7, and marginal improvement when keep increasing n from 7 to 15. The latency of training per batch is almost linear increased by improving n.

Comparison with IRGAN: The original IRGAN (Wang et al., 2017) doesn't work for dense text retrieval task as it doesn't contain the dual-encoder retrieval model for fast document indexing and

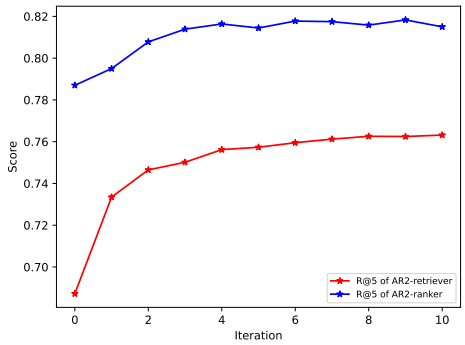


Figure 3: NQ R@5 on the number of iteration for both the AR2-retriever and the AR2-ranker.

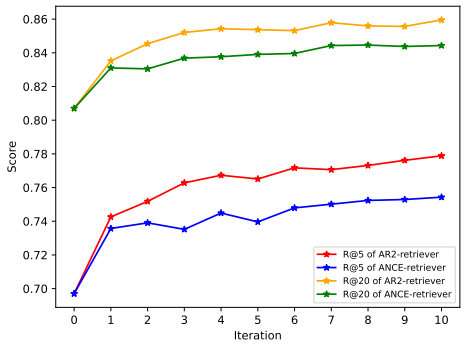


Figure 4: The comparison of ANCE and AR2 on NQ test set.

Table 7: The results of the second-stage ranking on Natural Questions test set. Note that we copy the numbers of the first block from the RIDER paper (Mao et al., 2021b).

Retriever	Ranker	R@1	R@5	R@10	R@20	R@50	R@100
GAR$^+$ (Mao et al., 2021a)	-	46.8	70.7	77.0	81.5	-	88.9
GAR$^+$ (Mao et al., 2021a)	BERT	51.4	67.6	75.7	82.4	-	88.9
GAR$^+$ (Mao et al., 2021a)	BART	55.2	73.5	78.5	82.2	-	88.9
GAR$^+$ (Mao et al., 2021a)	RIDER	53.5	75.2	80.0	83.2	-	88.9
AR2-**G**	-	58.7	77.9	82.5	86.0	88.5	90.1
AR2-**G**	AR2-**D**	65.6	81.5	84.9	87.2	89.5	90.1

search. However, it provides an conventional GAN framework for training the generative and discriminative models jointly for IR tasks. To compare the proposed AR2 with IRGAN, we replaced the generative and discriminative models in IRGAN with the retriever and ranker models in AR2, respectively. Therefore, with the configuration of the same model architectures for generator(retriever) and discriminator(ranker), The performance of the retriever is shown in Table 5. We see that AR2 outperforms IRGAN significantly.

Effect of Regularization: To study the effectiveness of regularization, we conducted ablation studies by removing the regularization term in the training of retrieval model. In Table 6, "*R*" refers to the regularization item, it shows that the regularization approach help to improve the R@1 and R@5 evaluation metrics. In additional, we compute the average entropy of distribution $p_\theta(\cdot|q, d, \mathbb{D}_q)$ on the NQ test set, here $\mathbb{D}_q$ is the retrieved top-15 documents. The average entropy measures the sharpness of distribution $p_\theta(\cdot|q, d, \mathbb{D}_q)$. In experiments, the average entropies for with *R* and w/o *R* in AR2-**G** are 2.10 and 1.70 respectively. It indicates that the regularization term could help smooth the prediction of probabilities in retriever.

Visualization of the Training Procedure: We visualize the changes of R@5 during the AR2-**G** training. The result is shown in Figure 3. We see that R@5 of both AR2-retriever and AR2-ranker gradually increases, as adversarial iteration increases. AR2-retriever has the most significant improvement after the first iteration, about 4.5%. While the training advance closer to the convergence, the improvement of R@5 also gradually slows down. In the end, AR2-retriever is improved by approximately 8% and AR2-ranker is improved by approximately 3%.

Adversarial Training versus Iterative Hard-Negative Sampling: To give a fair comparison of AR2 and ANCE (Xiong et al., 2020), we retrain the ANCE model by initializing it with the same warm-up AR2-$\mathbf{G^0}$ which leverages the advantage of continuous pre-training technique. In experiments, ANCE trains the retriever with an iterative hard-negative sampling approach instead of adversarial training in AR2. In Figure 4, we observe that AR2 steadily outperforms ANCE during training in terms of R@5 and R@10 evaluation metrics with the same model-initialization. It shows that AR2 is a superior training framework compared with ANCE.

Performance of the Pipeline: We evaluate the performance of the retrieve-then-rank pipeline on NQ dataset. The results are shown in Table 7. GAR$^+$ is a sparse retriever which ensembles GAR (Mao et al., 2021a) and DPR (Karpukhin et al., 2020a). BERT (Nogueira & Cho, 2019), BART (Nogueira et al., 2020), and RIDER (Mao et al., 2021b) are three ranking methods. BERT ranker is a cross-encoder, which makes a binary relevance decision for each query-passage pair. BART ranker generates relevance labels as target tokens in a seq2seq manner. RIDER re-ranks the retrieved passages based on the lexical overlap with the top predicted answers from the reader. The results show that AR2 pipeline significantly outperforms existing public pipelines.

5 Related work

Text Retrieval: Text retrieval aims to find related documents from a large corpus given a query. Retrieval-then-rank is the widely-used pipeline (Huang et al., 2020; Zou et al., 2021).

For the first stage retrieval, early researchers use sparse vector space models, e.g. BM25 (Yang et al., 2017) as the retriever which are based on the exact term matching. Recently, some works improve the traditional sparse retriever with neural network, e.g. Dai & Callan (2019) use BERT to dynamically generate term weights, doc2Query (Nogueira et al., 2019b), docTTTTTQuery (Nogueira et al., 2019a), and GAR (Mao et al., 2021a) use text generation to expand queries or documents to make better use of BM25.

Recently, dense retrieval methods have become a new paradigm for the first stage of retrieval. Various methods have been proposed to enhance dense retrieval, e.g. DPR (Karpukhin et al., 2020a) and ME-BERT (Luan et al., 2021) use in-batch negatives and contruct hard negatives by BM25; ANCE (Xiong et al., 2021), RocketQA (Qu et al., 2021), and ADORE (Zhan et al., 2021) improve the hard negative sampling by iterative replacement, denoising, and dynamic sampling, respectively; PAIR (Ren et al., 2021) leverages passage-centric similarity relation into training object; FID-KD (Izacard & Grave, 2020) and RDR (Yang & Seo, 2020) distill knowledge from reader to retriever; Guu et al. (2020) and Sachan et al. (2021b) enhance retriever by jointly training with downstream tasks. Some researches focus on the pre-training of dense retrieval, such as ICT (Lee et al., 2019), Condenser (Gao & Callan, 2021b) and Cocondenser (Gao & Callan, 2021a).

For the second stage ranking, previous works typically use cross-encoder based methods. The cross-encoder models which capture the token-level interactions between the query and the document (Guo et al., 2016; Xiong et al., 2017), have shown to be more effective. Various methods are proposed to enhance ranker, e.g. Nogueira & Cho (2019) use BERT to makes a binary relevance decision for each query-passage pair; Nogueira et al. (2020) adopt BART to generate relevance labels as target tokens in a seq2seq manner; Khattab & Zaharia (2020) and Gao et al. (2020) adopt the lightweight interaction based on the representations of dense retrievers to reduce computation. However, negative samples are statically sampled in these works. In AR2, negative samples for training the ranker will be dynamically adjusted with the progressive retriever.

Generative Adversarial Nets: Generative Adversarial Nets (Goodfellow et al., 2014) have been widely studied in the generation field, i.e., image generation (Brock et al., 2018), text generation (Yu et al., 2017). With a minimax game, GAN aims to train a generative model to fit the real data distribution under the guidance of a discriminative model. Few works study GAN to text retrieval. A related work is IRGAN (Wang et al., 2017). It proposes a minimax retrieval framework that aims to unify the generative and discriminative retrieval models.

6 Conclusion

In this paper, we introduce AR2, an adversarial retriever-ranker framework to jointly train the end-to-end retrieve-then-rank pipeline. In AR2, the retriever retrieves hard negatives to cheat the ranker, and the ranker learns to rank the collection of positives and hard negatives while provides progressive rewards to the retriever. AR2 can iteratively improve the performance of both retriever and ranker because (1) the retriever is guided by the progressive ranker; (2) the ranker learns better through the harder negatives sampled by the progressive retriever. AR2 achieves new state-of-the-art performance on all three competitive benchmarks.

REFERENCES

Dan Brickley, Matthew Burgess, and Natasha Noy. Google dataset search: Building a search engine for datasets in an open web ecosystem. In *WWW*, 2019.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2018.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *ACL*, 2017.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Zhuyun Dai and Jamie Callan. Deeper text understanding for ir with contextual neural language modeling. In *SIGIR*, 2019.

Luyu Gao and Jamie Callan. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540*, 2021a.

Luyu Gao and Jamie Callan. Is your language model ready for dense representation fine-tuning? *arXiv preprint arXiv:2104.08253*, 2021b.

Luyu Gao, Zhuyun Dai, and Jamie Callan. Modularized transfomer-based ranking framework. In *EMNLP*, 2020.

Luyu Gao, Zhuyun Dai, and Jamie Callan. COIL: revisit exact lexical match in information retrieval with contextualized inverted list. In *NAACL-HLT*, 2021.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NIPS*, 2014.

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

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *ICML*, 2020.

Linmei Hu, Siyong Xu, Chen Li, Cheng Yang, Chuan Shi, Nan Duan, Xing Xie, and Ming Zhou. Graph neural news recommendation with unsupervised preference disentanglement. In *ACL*, 2020.

Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. Embedding-based retrieval in facebook search. In *KDD*, 2020.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*, 2013.

Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. *arXiv preprint arXiv:2012.04584*, 2020.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 2019.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017.

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

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*, 2020b.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *SIGIR*, 2020.

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. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466, 2019.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *ACL*, 2019.

Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, Jiusheng Chen, Daxin Jiang, Jiancheng Lv, and Nan Duan. Rikinet: Reading wikipedia pages for natural question answering. In *ACL*, 2020.

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

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK, 2008. ISBN 978-0-521-86571-5. URL `http://nlp.stanford.edu/IR-book/information-retrieval-book.html`.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. Generation-augmented retrieval for open-domain question answering. In *ACL*, 2021a.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. Reader-guided passage reranking for open-domain question answering. In *Findings ofACL/IJCNLP*, 2021b.

Bhaskar Mitra and Nick Craswell. Neural models for information retrieval. *CoRR*, abs/1705.01509, 2017. URL `http://arxiv.org/abs/1705.01509`.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*, 2016.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

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

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*, 2019b.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*, 2020.

Barlas Oğuz, Kushal Lakhotia, Anchit Gupta, Patrick Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen-tau Yih, Sonal Gupta, et al. Domain-matched pre-training tasks for dense retrieval. *arXiv preprint arXiv:2107.13602*, 2021.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *NAACL-HLT*, 2021.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. PAIR: leveraging passage-centric similarity relation for improving dense passage retrieval. In *Findings of ACL/IJCNLP*, 2021.

Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. End-to-end training of neural retrievers for open-domain question answering. In *ACL/IJCNLP*, 2021a.

Devendra Singh Sachan, Siva Reddy, William Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering. *arXiv preprint arXiv:2106.05346*, 2021b.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *WWW*, 2014.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. Ernie 2.0: A continual pre-training framework for language understanding. In *AAAI*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *SIGIR*, 2017.

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. Transformers: State-of-the-art natural language processing. In *EMNLP*, 2020.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. End-to-end neural ad-hoc ranking with kernel pooling. In *SIGIR*, 2017.

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

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

Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Enabling the use of lucene for information retrieval research. In *SIGIR*, 2017.

Sohee Yang and Minjoon Seo. Is retriever merely an approximator of reader? *arXiv preprint arXiv:2010.10999*, 2020.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, 2017.

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

Lixin Zou, Shengqiang Zhang, Hengyi Cai, Dehong Ma, Suqi Cheng, Shuaiqiang Wang, Daiting Shi, Zhicong Cheng, and Dawei Yin. Pre-trained language model based ranking in baidu search. In *KDD*, 2021.

A APPENDIX

A.1 PROOF

Proof of Eqn. 9: Suppose $d_i^- \in \mathbb{D}_{||}^-$ is sampled iid.

$$\begin{aligned} J^\theta &= \mathbf{E}_{\mathbb{D}_q^- \sim G_\theta(q,\cdot)} \left[\text{log} p_\phi(d|q,d,\mathbb{D}_q^-) \right] \\ &\leq \mathbf{E}_{\mathbb{D}_q^- \sim G_\theta(q,\cdot)} \left[\text{log} p_\phi(d|q,d,d_i^-) \right] \\ &\approx \mathbf{E}_{d_i^- \sim p_\theta(d_i^-|q,\mathbb{D}_q^-)} \left[\text{log} p_\phi(d|q,d,d_i^-) \right] \\ &= \hat{J}^\theta \end{aligned} \tag{13}$$

In the first and second lines of Eqn. 13, $\mathbb{D}_q^-$ indicates set of the documents sampled by $G_\theta(q,\cdot)$. In third line, $\mathbb{D}_q^-$ is approximated to be the set of top-n retrieved negative documents. From line 2 to line 3 in Eqn. 13, we restrict the sampling space to be the top-n set instead of the whole corpus for approximation.

Proof of Eqn. 10:

$$\begin{aligned} \nabla_\theta \hat{J}^\theta &= \nabla_\theta \mathbf{E}_{d_i^- \sim p_\theta(d_i^-|q,\mathbb{D}_q^-)} \left[\text{log} p_\phi(d|q,d,d_i^-) \right] \\ &= \sum_i \nabla_\theta p_\theta(d_i^-|q,\mathbb{D}_q^-) \left[\text{log} p_\phi(d|q,d,d_i^-) \right] \\ &= \sum_i p_\theta(d_i^-|q,\mathbb{D}_q^-) \nabla_\theta \log p_\theta(d_i^-|q,\mathbb{D}_q^-) \left[\text{log} p_\phi(d|q,d,d_i^-) \right] \\ &= \mathbf{E}_{d_i^- \sim p_\theta(d_i^-|q,\mathbb{D}_q^-)} \nabla_\theta \log p_\theta(d_i^-|q,\mathbb{D}_q^-) \left[\log p_\phi(d|q,d,d_i^-) \right] \end{aligned} \tag{14}$$

A.2 EFFICIENCY REPORT

We list the time cost of training and inference in Table 8. The evaluation is made with 8 Nvidia A100 GPUs. The max step of ANCE training is from the ANCE's open-source website [2].We estimate the overall training time without taking account of the time of continuous pretraining step and warming-up step.

Table 8: Comparison of Efficiency

	DPR	ANCE	AR2(n=15)	AR2(n=1)
Training				
Batch Size	128	128	64	64
Max Step	20k	136k	20k	20k
BP for Retriever	1.8h	11h	2.3h	1h
BP for Ranker	-	-	0.75h	0.35h
Iteration Number	0	10	10	10
Index Refresh	0.5	0.5h	0.5h	0.5h
Overall	1.85h	16h	9.1h	6.4h
Inference				
Encoding of Corpus	20min	20min	20min	20min
Query Encoding	40ns	40ns	40ns	40ns
ANN Index Build	2min	2min	2min	2min
ANN Retrieval(Top-100)	2ms	2ms	2ms	2ms

[2] https://github.com/microsoft/ANCE

A.3 HYPERPARAMETERS

Table 9: Hyperparameters for AR2 training.

	Parameter	NQ	TriviaQA	MS-MARCO
Default	Max query length	32	32	32
	Max passage length	128	128	128
AR2-G^0	Learning rate	1e-5	1e-5	1e-4
	Negative size	255	255	127
	Batch size	128	128	64
	Temperature τ	1	1	1
	Optimizer	AdamW	AdamW	AdamW
	Scheduler	Linear	Linear	Linear
	Warmup proportion	0.1	0.1	0.1
	Training epoch	40	40	3
AR2-D^0	Learning rate	1e-5	1e-5	1e-5
	Negative size	15	15	15
	Batch size	64	64	256
	Temperature τ	1	1	1
	Optimizer	AdamW	AdamW	AdamW
	Scheduler	Linear	Linear	Linear
	Warmup proportion	0.1	0.1	0.1
	Training step per iteration	1500	1500	1500
	Max step	2000	2000	4000
AR2-G	Learning rate	1e-5	1e-5	5e-6
	Negative size	15	15	15
	Batch size	64	64	64
	Temperature τ	1	1	1
	Optimizer	AdamW	AdamW	AdamW
	Scheduler	Linear	Linear	Linear
	Warmup proportion	0.1	0.1	0.1
	Training step per iteration	1500	1500	1500
	Max step	15000	15000	15000
AR2-D	Negative size	15	15	15
	Learning rate	1e-6	1e-6	5e-7
	Batch size	64	64	64
	Temperature τ	1	1	1
	Optimizer	AdamW	AdamW	AdamW
	Scheduler	Linear	Linear	Linear
	Warmup proportion	0.1	0.1	0.1
	Training step per iteration	500	500	500
	Max step	5000	5000	5000