


Article

SS-BERT: A Semantic Information Selecting Approach for Open-Domain Question Answering

Xuan Fu ^{1,†}, Jiangnan Du ², Hai-Tao Zheng ^{3,*} , Jianfeng Li ², Cuiqin Hou ², Qiyu Zhou ² and Hong-Gee Kim ⁴¹ Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China² Ping An Technology, Shenzhen 518000, China³ Pengcheng Laboratory, Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China⁴ Dental College, Seoul National University, Seoul 08826, Republic of Korea

* Correspondence: zheng.haitao@sz.tsinghua.edu.cn

† Work conducted during the first author's internship at Ping An Technology.

Abstract: Open-Domain Question Answering (Open-Domain QA) aims to answer any factoid questions from users. Recent progress in Open-Domain QA adopts the “retriever-reader” structure, which has proven effective. Retriever methods are mainly categorized as sparse retrievers and dense retrievers. In recent work, the dense retriever showed a stronger semantic interpretation than the sparse retriever. When training a dual-encoder dense retriever for document retrieval and reranking, there are two challenges: negative selection and a lack of training data. In this study, we make three major contributions to this topic: negative selection by query generation, data augmentation from negatives, and a passage evaluation method. We prove that the model performs better by focusing on false negatives and data augmentation in the Open-Domain QA passage rerank task. Our model outperforms other single dual-encoder rerankers over BERT-base and BM25 by 0.7 in MRR@10, achieving the highest Recall@50 and the max Recall@1000, which is restricted by the BM25 retrieval results.

Keywords: open-domain question answering; passage rerank; data augmentation; negative selection; BERT



Citation: Fu, X.; Du, J.; Zheng, H.-T.; Li, J.; Hou, C.; Zhou, Q.; Kim, H.-G. SS-BERT: A Semantic Information Selecting Approach for Open-Domain Question Answering. *Electronics* **2023**, *12*, 1692. <https://doi.org/10.3390/electronics12071692>

Academic Editor: Marcello Trovati, Umar F Khan

Received: 3 March 2023

Revised: 21 March 2023

Accepted: 27 March 2023

Published: 3 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the rapid development of the internet, people are facing the problem of obtaining the required response from a large amount of information. Search engines alleviate this issue; however, they can only provide a list of web page results rather than a direct answer. Compared with search engines, Question Answering (QA) systems are able to provide direct answers to satisfy users. The early question answering systems (e.g., [1,2]) are more like expert systems. As QA systems gradually develop, the knowledge source changes from an artificially encoded knowledge base to text collections. Restricted Domains Question Answering (RDQA [3]) aims to answer questions in restricted domains. Compared with RDQA, Open-Domain QA [4] aims to find the answers to any factoid questions using an unlimited knowledge base in any domain.

Due to the gradual maturity of associated techniques in natural language processing, deep learning has been used in almost every stage of Open-Domain QA. DrQA [5] is the first Open-Domain QA model to use Neural Machine Reading Comprehension (Neural MRC) in Open-Domain QA, resulting in a two-stage structure QA system which consists of a retriever and a reader. In a two-stage QA system, the retriever retrieves a list of passages from a large database, then the reader provides the final answer, the accuracy of which is not only decided by the reader itself but also by the performance of the retriever. Traditional retrievers are efficient, with an inverted index, but face difficulties (e.g., term mismatch [6]) in matching queries and passages, e.g., Term Frequency–Inverse Document Frequency

(TF-IDF) and Best Match 25 (BM25). Recently, based on Pre-trained Language Models (PLMs), the dual-encoder has been widely used to learn the relations between queries and passages. It outperforms sparse retrievers.

There are two major challenges when it comes to training a dual-encoder for document retrieval:

The first is the **hard negative selection problem**. A hard negative is a negative that is more likely to be regarded as a positive by the model and is more valuable than simple negatives (there may be lots of overlapping tokens between the hard negative and the positive). Since most negatives in Open-Domain QA are not labeled, selecting hard negatives for the model is problematic. There are two main training approaches used to select hard negatives for dual-encoders: list-wise and pair-wise. In the list-wise approach, the dual-encoder chooses in-batch negatives, facing the limit of the memory of the GPU (or other devices). When using the pair-wise approach, the dual-encoder uses triples of query, positive and negative as the training data, leading to issues related to the low quality of training data. Both methods face the hard negative selection problem.

The second is the **training data deficiency problem**. More training data usually improves the performance of the Open-Domain QA models, but it is expensive to acquire labeled data. In recent work, cross-encoders have been used to build labels for unlabeled data, especially in large-scale datasets (e.g., MS-MARCO [7]; Natural Questions [8]), but the method faces two problems: the cross-encoder is too expensive and the labeled data are limited since they only come from the original unlabeled passages (the total amount of training data is constant).

In this paper, we present several methods with which to alleviate these problems relating to dual-encoder training in Open-domain QA systems. First, we change the dataset distribution to enable the dual-encoder to learn more about the difference between false negatives and negatives using a fine-tuned BART [9] model. Second, to acquire more training data, we use the BART model to generate queries. We choose the generated queries which have low similarity scores with the positive queries as the new queries, the negatives as new positives, and the positives as new negatives. Third, we use the probability of generating the query from the negative as an evaluation of the negative.

Our contributions are as follows:

- We present a BERT-based semantic information selection method, named SS-BERT, to alleviate the **hard negative selection problem** and the **training data deficiency problem**.
- We prove that the dual-encoder performs better by focusing on hard negatives in the Open-Domain QA passage rerank task;
- The proposed methods outperform other passage rerankers on MRR@10 and Recall@50 with a single dual-encoder based on the BERT-based model and BM25 retriever.

2. Related Work

The retriever, the reranker, and the reader are the core modules of the two-stage Open-domain QA, in this section, we introduce how retrievers and rerankers have developed recently and give a brief review of the sequence-to-sequence method, which is applied in our proposed model.

2.1. Retriever Methods

Retriever methods are mainly divided into sparse retrievers and dense retrievers. In recent work, dense retrievers (e.g., [10–12]) outperformed traditional sparse retrievers (e.g., TF-IDF and BM25). They usually represent queries and passages as vectors in low-dimensional vector space (compared with the dictionary size) and calculate their similarity in terms of the dot product.

Retrieval models usually use dual encoders or cross-encoders, which are representation based and interaction based, respectively. Furthermore, the late interaction encoder is representation–interaction based (e.g., [13]). The three methods are shown in Figure 1.

Dual encoders use two encoders to represent queries and passages, respectively (e.g., [10]). Cross-encoders use one encoder to jointly represent queries and passages (e.g., [14–16]). The cross-encoder captures the semantic relationships between queries and passages more precisely than the dual encoder, but it requires much more computational resources than the dual encoder, and it is usually used as the passage reranker.

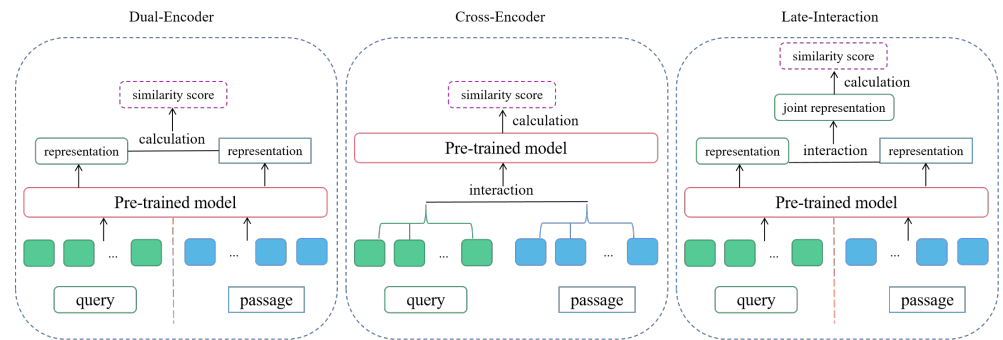


Figure 1. Retriever Methods. Among the methods, dual-encoder is more efficient, cross-encoder is more accurate, and late-interaction looks for a balance of efficiency and accuracy.

2.2. Passage Reranker

Recently, a two-stage retriever structure has been proved efficient, which further divides the retriever into two categories: retriever and passage reranker. For example, Rocket QA [14] con-cats the query and the passage as the input to train the reranker, then uses the reranker to drop the retrieval results with low confidence scores and generate labeled data with which to train the cross-encoder. PAIR [16] follows Rocket QA and presents a method for measuring the similarity of passages to drop false negatives. Rocket QA v2 [15] presents dynamic list-wise distillation to jointly train the retriever and reranker. ColBERT [13] presents a light late interaction model over BERT [17] as a reranker.

Since the cross-encoder is expensive, in order to balance efficiency and accuracy simultaneously, our method uses a dual-encoder as the reranker. We fine-tune a BART [9] model to improve the dual-encoder through negative selection, data augmentation, and passage evaluation, which is described in the Methodology and the Experimental Section.

2.3. Sequence-to-Sequence Method

Sequence-to-Sequence (Seq2Seq) is a method that generates a sequence with a given sequence. It was first presented in 2014 ([18,19]). GPT [20] uses a diverse corpus of unlabeled text to train a generative pre-train model, then fine-tunes the model on different tasks, e.g., text generation. BART [9] is one of the best seq2seq models for text generation, combining the advantages of BERT [17] and GPT [20], BART uses a cascade structure of a bidirectional encoder and an auto-regressive decoder to allow arbitrary noises to avoid model dependence on sequence information. DocT5query [21] follows DocTquery [22]; it uses T5 [23] to generate queries from given documents. The queries can be answered with the documents, then the generated queries are added to the documents. Recent work [24] used BART-large (374M parameters, 12 layers in the encoder and decoder) to generate queries from English Wikipedia in a zero-shot retrieval task, unsupervised, and found that for some datasets, the model training on the generated dataset outperformed the original one. Our proposed model uses a fine-tuned BART-base model to generate queries from negatives, which is described in the Methodology and the Experimental Section.

3. Methodology

In this section, we present a light passage reranker that selects semantic information including negatives via the BART [9] model and method over BERT [17] dual encoder (called SS-BERT). SS-BERT has the advantage of being a light model which can be used as a module in another retriever framework.

3.1. Task Description

To choose passages to obtain the answer to a natural question from a large corpus, a two-stage retriever begins by using a retriever to choose a list of passages from the corpus. Since we present our model as a flexible module for the retriever, we continue to use BM25 as the retriever and keep the top k results of BM25, then we use a passage reranker to rerank the top k results.

3.2. Negative Selection by Query Generation

The QA system faces the problem of insufficient training data since it is expensive to obtain labeled data. We try to improve the quality of training data to alleviate the problem. In the retrieval task, when using the pairwise training method, training data are in the form of triples consisting of query, positive, and negative (denoted as $T(q, p, n)$). However, there are more negatives than positives, which means the quality of negatives may not be very high because they are actually unlabeled. Furthermore, hard negatives are not labeled; thus, the model would regard them as simple negatives. Table 1 shows a hard negative case in the MS-MARCO dataset; our proposed negative selection method tries to judge the hard negatives.

Table 1. Hard Negative Case. In this case, the keywords ‘a master’s degree’ are mentioned many times both in the positive and the negative. The cosine similarity of the positive and the hard negative is 0.9, which is calculated as shown in Section 4.2. This kind of negative is called hard negative; however, a hard negative is always regarded as a common negative, since it is too expensive to label all negatives.

Data Type	Text
query	How long it takes to get a master’s degree
positive	In most cases, a master’s degree program takes two years to complete, although there are exceptions to the rule. If you’d like to know how long it would take to earn a master’s degree , you should consider how much time you could devote to school and the specific type of program you will be enrolling in.
hard negative	Normally you need to get a bachelor’s degree before you get a master’s degree . The bachelor’s degree is the basic college degree. it can be completed in three years including summers, or in four years if you take summers off. The master’s degree follows it. Any individuals may pursue a master’s degree in a field unrelated to their bachelor’s degree. My bachelor’s degree is in psychology. However, my master’s degree is in organizational management.

To alleviate the problem, we fine-tune the BART-base model with the MS-MARCO dataset (using the *Huggingface Transformers* library [25]) and use the model to generate queries from the negatives (denote the queries as generated queries q_G). Then, we compute the similarity score (denoted as $sim(q, q_G)$) between the queries q) and the generated queries:

$$sim(q, q_G) = E(q) \cdot E(q_G) \quad (1)$$

We set a threshold (denoted as t_G) and drop the negative if $sim(q, q_G)$ is smaller than t_G . After all the negatives have been selected, we obtain the negatives set (denoted as n_H), which represents the part of the original data which are more similar to the positives and thus more likely to be hard negatives; the pipeline is shown in Figure 2. Then, we mix the triples $T(q, p, n)$ and $T(q, p, n_H)$ in a 1:1 ratio and obtain new triples T_{hybrid} , which

have a data distribution that focuses more on the false negatives. The strategy is shown in Figure 3.

The method enables the dual encoder to learn more about the differences between false negatives and positives. As a result, it performs better, which indicates a novel method for data pre-processing for the task (e.g., training a data reranker to change the data distribution to improve the dataset quality). With the proposed method, the model performs better; the experimental results are shown in Section 4.3.

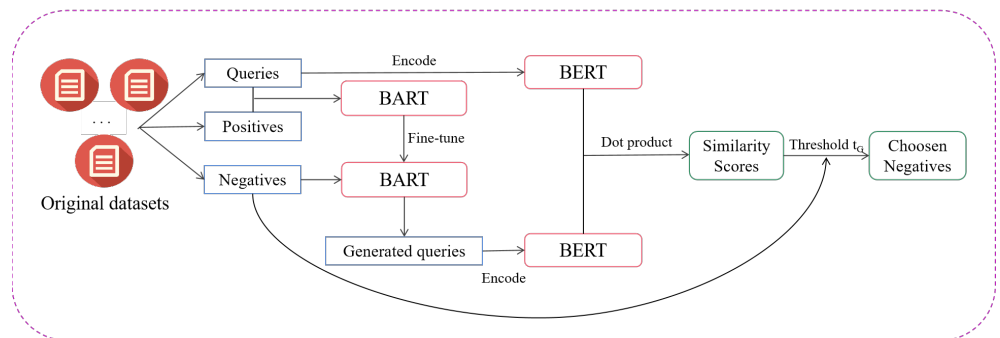


Figure 2. The pipeline of negative choices from the original dataset. (1) A BART model is fine-tuned with query-positive pairs; (2) the fine-tuned BART model generates queries from the negatives; (3) the similarity scores of queries and generated queries are calculated after being encoded by a BERT model; (4) choose negatives of which the similarity scores are higher than the threshold.

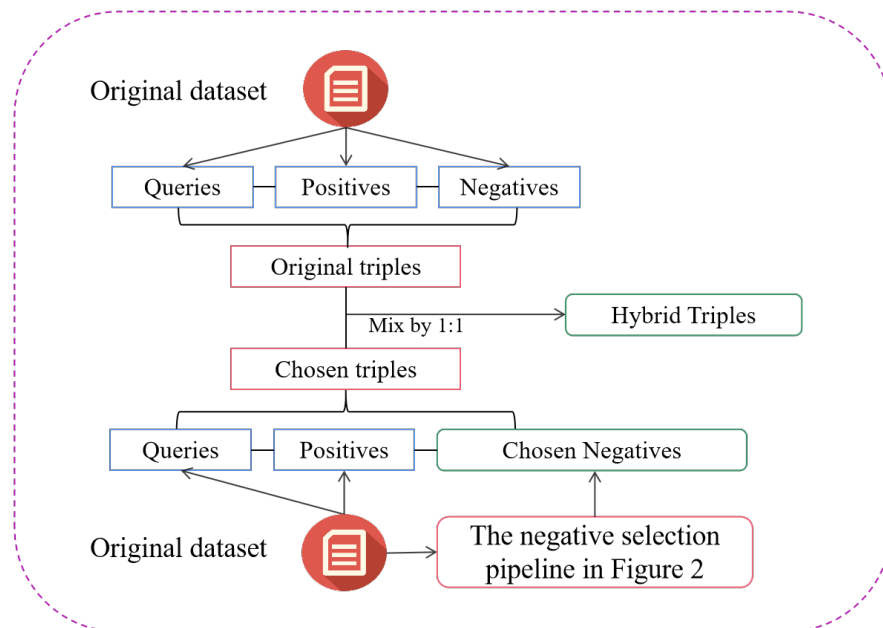


Figure 3. Acquire the hybrid triples. The chosen triples are more likely to be hard negatives.

3.3. Data Augmentation From Negatives

We set a threshold (denoted as t_A) using the generated query q_G , the negative, and the positive as a new triple $T(q_G, p_{new}, n_{new})$, of which the s_G is lower than t_A . This means the positive is not similar to the negative, so it is actually the negative of the generated query. Compared with the original triple $T(q, p, n)$, p is the same as n_{new} , and n is the same as p_{new} . We add the new triples $T(q_G, p_{new}, n_{new})$ into T_{hybrid} and the model performs better. The experimental results are shown in Section 4.3, and a case of data augmentation is shown in Section 4.4.

3.4. Passage Evaluation

In passage rerank, we set the query as the target and the passage as the input, denoting the probability of a BART model generating the target from the input as P_t . A high P_t means the passage is more likely positive. In Formula (2), n is the length of a query, tok_t is the t_{th} token of a query, and s_{query} is the score needed for BART to generate the query from the passage. A ratio (denoted as k_s) is used to limit the influence of the score s_{query} . In passage rerank, we denote the score of a passage as $s_{passage}$ ($s_{passage}$ is calculated as ColBERT [13]). It becomes $s'_{passage}$ with our method, as shown below. The experimental results are shown in Section 4.3:

$$s_{query} = - \sum_{t=1}^n \log p(tok_t | tok_1, \dots, tok_{t-1}, passage) \tag{2}$$

$$s'_{passage} = s_{passage} + k_s \cdot s_{query} \tag{3}$$

4. Experiments

In this section, we describe the experimental setting details, including how to make the generated query reusable for saving computing resources, evaluation metrics, experimental results, and case studies.

4.1. Experimental Settings

4.1.1. Dataset

This paper uses MS-MARCO [7] to evaluate the proposed model. MS-MARCO is one of the most popular Open-Domain QA datasets. In MS-MARCO, all queries are sampled from real anonymous user queries through Bing or Cortana. The passages are extracted from real web documents by the Bing search engine. The answers are human-generated from the context passages and are strongly encouraged to be in the form of a complete sentence.

The dataset has 502,939 queries in the training set; 6980 queries in the dev set; 6837 queries in the test set; 8,841,823 non-redundant passages and 39,769,172 triples (a triple consists of a query, a positive, and a negative; the negatives are from the 8,841,823 passages, which means each passage is used 4.5 times on average). Each query has an average of only 1.07 positives and 5.97 words; each passage has an average of 56.58 words.

In the MS-MARCO dataset, since the queries come from real anonymous users, only part of the queries contain keywords such as “what”, “how”, “where”, etc. As mentioned in Section 3.3, our proposed method generates queries from negatives with a fine-tuned BART-base [9] model, the distribution of queries’ keywords in the generated queries is compared to that in the MS-MARCO dataset in Table 2. The top two words in both datasets are “what” (42.2% and 43.8%) and “how” (15.3% and 19.5%), and the third word in MS-MARCO is “where” (4.4%, 4.3% in the generated dataset) and the third word in the generated dataset is “which” (4.9%, 1.7% in MS-MARCO). The similar keyword distribution between the two datasets indicates the generated queries are reliable.

Table 2. The distribution of queries’ keywords of MS-MARCO and SS-BERT generated queries.

Query Contains	Percentage of Queries (MS-MARCO)	Percentage of Queries (SS-BERT)
what	42.2%	43.8%
how	15.3%	19.5%
where	4.4%	4.3%
when	2.0%	3.4%
why	1.8%	1.1%
who	1.7%	4.9%
which	1.4%	0.3%
others ¹	31.2%	22.5%

¹ “others” means queries that do not contain the above words.

4.1.2. Evaluation Metrics

Following previous work, we use Mean Reciprocal Rank (MRR) and Recall at Top k (R@k) to evaluate the performance of the reranker.

MRR calculates the averaged reciprocal of the rank at which the first positive passage is retrieved. When using MRR we focus more on the EM (exact match) part than R@k. When we use MRR@k to evaluate the model on N queries, it is calculated as below, p_i represents the rank of the true positive:

$$MRR@k = \frac{1}{N} \sum_{i=1}^N \frac{1}{p_i} (p_i \leq k) \quad (4)$$

R@k calculates the proportion of queries to which the top k retrieved passages contain positives. We use R@50 and R@1000 to evaluate the reranker. When we use R@k to evaluate the model on N queries, it is calculated as below, $TP_j = 1$ when the j_{th} sample is true positive, $TP_j = 0$ when the j_{th} sample is negative (one true positive at most):

$$Recall@k = \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^k TP_j \quad (5)$$

Rouge Rouge-N [26] calculates the n-gram recall between a candidate text and a reference text, Rouge-L calculates the longest common sub-sequence between the candidate and the reference, we use Rouge-1, Rouge-2, and Rouge-L to evaluate the fine-tuned BART-base model mentioned in Section 3.2. Rouge-N is calculated as Formula (6), M is the number of candidates, $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in the candidate and the reference. Rouge-L is calculated as Formula (7), β is a large constant, R_{lcs} is obtained by dividing $LCS(R, C)$ (the longest common sub-sequence of the candidate and the reference) by the length of the reference, C_{lcs} is obtained by dividing $LCS(R, C)$ by the length of the candidate.

$$RougeN = \frac{1}{M} \sum_{i=1}^M \frac{\sum_{gram_n \in Reference} Count_{match}(gram_n)}{\sum_{gram_n \in Reference} Count(gram_n)} \quad (6)$$

$$RougeL = \frac{(1 + \beta^2) R_{lcs} C_{lcs}}{R_{lcs} + \beta^2 C_{lcs}} \quad (7)$$

4.1.3. Implementation Details

We choose the MS-MARCO triples dataset, which has a format of query, positive passage, and negative passage. The triples are disordered and the negatives are repeated. We build labels between each triple and non-redundant negative (a non-redundant negative means one of the 8,841,823 passages mentioned in Section 4.1.1), then build labels between each negative and its generated query. Once we acquire all generated queries from the non-redundant negatives, we use the labels to distinguish the relationships among queries, positives, negatives, and generated queries when we try new methods or other hyper-parameters.

4.2. Passage Rerank

Fine-tune PLM *Huggingface Transformers* library [25] provides the method to fine-tune BART [9] model on CNN-Dailymail dataset ([27,28]) of news articles paired. We follow the framework, preprocess the MS-MARCO query-positive pairs to suit *Huggingface Transformers*, and fine-tune BART-base on these data. In this paper, for efficiency, we choose BART-base (6 layers in the encoder and decoder) to generate queries from negatives. The fine-tuning process costs 5 h on 4 NVIDIA Tesla V100 GPUs (with 16G RAM). The version of the transformers is 4.17.0. The ratio of train: test: valid is 8:1:1, dropout is 0.1, the number of

epochs is 5, the batch size is 64, optimization steps is 26,000, and the gradient accumulation step is 1. Cases of the generated queries are shown in Table 3, and the performance of the fine-tuned BART-base model is shown in Table 4, which indicates the degree of confidence of the generated queries.

Table 3. Query generation case, in this case, we choose three query-negative pairs to visually show the quality of the generated queries.

Generated Queries	Negatives
How to apply for Schengen Visa	To apply for a Schengen Visa you should apply to the embassy of the first country you intend to visit. If you plan to visit Italy, France, and Spain (in that order) then you only need to apply to the Italian embassy and your visa is good for France and Spain too.
What diseases do finches carry	However, in terms of diseases that can be transmitted to other birds, finches can carry pretty much any avian disease, including influenza viruses, Newcastle's disease virus, many different bacterial diseases and both internal and external parasites.
What is the function of capillaries	Veins carry blood from the other parts of the body to the heart. They have valves to stop the backward flow of blood. Capillaries are only one cell thick in reality, and they help to diffuse substances from the blood to the cell through the cell membrane (which is selectively permeable).

Table 4. The performance of the fine-tuned BART-base model.

Model	Dataset	Rouge-1	Rouge-2	Rouge-L
Our Fine-tuned BART-base	MS-MARCO	55.66	31.28	52.67

Negative Selection We first process the MS-MARCO triples (query, positive, and negative) train set and change the train set into ordered. Then, we build maps from triples to non-redundant negatives and use the fine-tuned BART-base model to generate queries (denoted as q_G) from the non-redundant negatives. Then, we use sentence-BERT [29] to calculate the similarity score (denoted as s_G) of queries (denoted as q) and q_G by dot product (using cosine similarity as the metric). Then, we drop the negatives of which the s_G is lower than a threshold (denoted as t_G), the distribution of s_G is in Table 5. We set the threshold (denoted as t_G) as 0.3 which means we keep the top 50% of similar negatives for each positive since these parts of negatives are more likely to be hard negatives. Then, we hybrid the selected triples and original triples by 1:1. Finally, we train the dual encoder with the hybrid train set.

Table 5. The distribution of the similarity scores between queries and generated queries. The distribution is important evidence for the threshold selection in the negative selection method and the data augmentation method, which indicate the degree of confidence of the selected data. The similarity scores are calculated through sentence-BERT [29].

Similarity Score Range	Percentage	Negatives Amount
(−1.0, 0.1)	20.17%	8,022,848
(0.1, 0.3)	30.69%	12,204,995
(0.3, 0.6)	39.02%	15,519,172
(0.6, 1.0)	10.12%	4,022,157
	Total	39,769,172

Data Augmentation We set the threshold t_A as 0.1, which means we choose about 8 M generated queries to build new triples. The triples are the top 20% of credible generated data, seen from the distribution of similarity scores, approximately. The rationality of the chosen threshold is discussed in Section 4.3, and a case of the new triples is shown in Section 4.4.

Passage Evaluation The score of the passage is denoted as s_{query} , we add $(k_s \cdot s_{query})$ on $s_{passage}$ to re-evaluate the passage. The ratio k_s should not be too high since $s_{passage}$ and s_{query} are similar in size, a big k_s intuitively means we regard a model fed with about 500 k data as reliable as one fed with about 50 M data. When k_s is in [0.05, 0.15], the model performs best. The proposed method enhances the robustness of the model, thus improving its performance.

4.3. Experimental Results

Compared Methods

We report the results of the following baselines: BM25 (official), K-NRM [30], Duet ([31,32]), fastText+ConvK-NRM ([33]), ColBERT [13], and COIL [34]. Detailed descriptions of the baselines are given below: Duet and fastText+ConvK-NRM are representations of neural matching models that have been tested in MS-MARCO [7] passage rerank task. ColBERT and COIL are both BERT-based and have a single dual-encoder architecture. We do not show the result of BERT-large models, since the large model with more parameters and neural network layers surely improves the model performance. We do not compare the result with Multi-stage BERT [35], which has a tandem structure of a list-wise BERT and a pair-wise BERT because our model can be seen as a pair-wise BERT of the architecture.

- BM25 (official) is a traditional bag-of-words information retrieval method (a sparse retriever), the rerank task of the following compared methods is based on the BM25 retriever result, which limits the Recall@1000 to 0.814.
- Soft-match of queries to document is a weaker signal compared with an exact match, K-NRM focuses on soft-match features extraction through kernels, uses a kernel-pooling technique to build word embeddings, then uses a translation matrix to model word-level similarities.
- Duet uses two deep neural networks, a local sub-model to match the term space of the queries and documents, and a distributed sub-model to match the learned latent space of the queries and documents.
- fastText+ConvK-NRM conduct a set of experiments on K-NRM, ConvK-NRM [36] and MatchPyramid [37], present a method that adopts sub-word token embeddings to alleviate the absence of low-frequency words in the word embeddings list.
- COIL gives the matching scores between queries and documents through overlapping query document tokens' contextualized representations. COIL-tok uses the exact match of tokens, and COIL-full uses CLS matching in addition.
- ColBERT introduces a late interaction architecture to model the similarity between queries and documents.

The results are in Table 6, our SS-BERT outperforms other neural matching models and dual-encoders based on BERT-base and BM25 retriever, all our proposed methods can improve MRR@10 and achieve the maximum of Recall@1000, SS-BERT(d) has a higher Recall@50 than baselines. The result shows that changing the data distribution improves the model performance since the method enables the dual-encoder to focus more on hard negatives. Data augmentation from negatives improves the model performance since the method provides the dual-encoder with more reliable training data. Passage evaluation improves the model performance since it enhances the robustness of the model.

Table 6. The results in MS-MARCO passage rerank task, PLM means the pre-trained language models used in the dual-encoder. SS-BERT(t) means the model uses the negatives select method, SS-BERT(h) means the model uses the negatives select method and the data augmentation method, SS-BERT(d) means the model uses the negatives select method, the data augmentation method, and the passage evaluation method.

Methods	PLM	MRR@10	Recall@50	Recall@1000
BM25(official)	-	16.7	-	0.814
K-NRM	-	19.8	-	-
Duet	-	24.3	-	-
fastText+ConvK-NRM	-	29.0	-	-
COIL-tok	BERT _{base}	33.6	-	-
COIL-full	BERT _{base}	34.8	-	-
ColBERT	BERT _{base}	34.9	0.751	0.814
SS-BERT(t)	BERT _{base}	35.3	0.746	0.814
SS-BERT(h)	BERT _{base}	35.5	0.751	0.814
SS-BERT(d)	BERT _{base}	35.6	0.753	0.814

In order to prove the rationality of the threshold t_A ($t_A = 0.1$) in Section 4.2, when t_A changes, as shown in Table 5, the amount of new triples consisting of generated queries grows as t_A grows; however, a higher t_A means the generated queries has a lower degree of confidence.

Figure 4 shows how the model performs with different t_A , a higher t_A means the new triples have a lower degree of confidence, thus the model performs worse when the threshold is too high, which provides the model with too much “bad data”. Figure 5 shows how many training steps the model needs to perform best with different t_A , a higher t_A means more training data; thus, the model training cost is much higher. In both experiments, the chosen t_A are in [0.10, 0.16, 0.22, 0.28, 0.34, 0.40], according to a sequence of equal difference.

4.4. Case Study

We choose a case of the data augmentation shown in Table 7, the original training triple consists of a query, a positive, and a negative, and the generated query is generated from the original negative. From the table, we see that the original negative contains the answer to the generated query while the original positive does not. Therefore, we acquire a new triple that regards the generated query as the new query, the original negative as the new positive, and the original positive as the new negative. In the label, the main overlapping tokens are in bold, we see that although the positive and negative have lots of overlapping tokens, our method finds that they are different indeed, our method ensures the quality of the new triple.

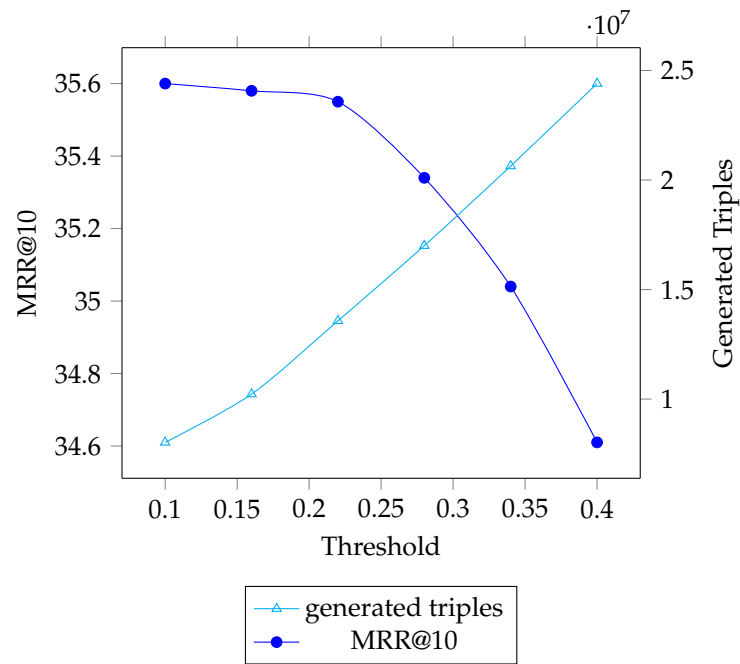


Figure 4. This figure shows the model performance with different thresholds t_A , which present different amounts of new triples. When the amount of new triples grows, the model performs worse since the data quality is worse.

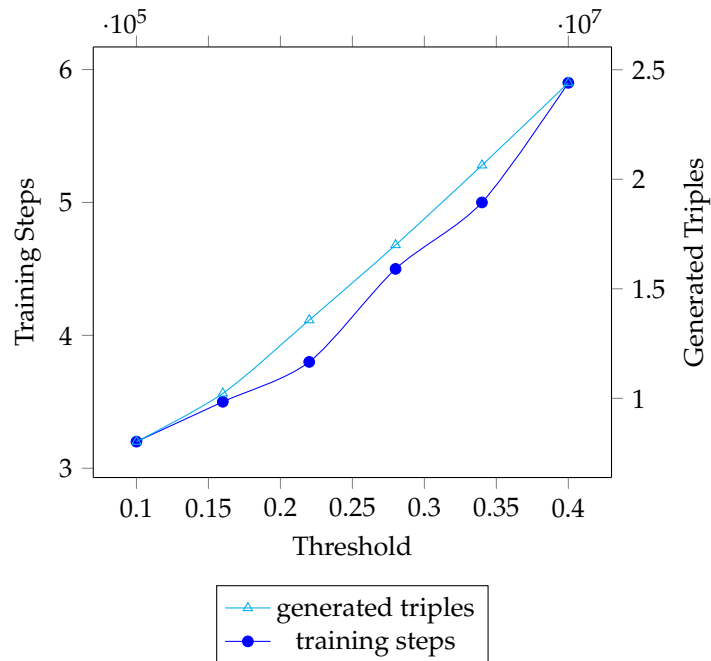


Figure 5. This figure shows the training steps for the model to perform best with different thresholds t_A , which present different amounts of new triples, the model costs much more computational resources when t_A is too big. Every 100,000 training steps cost about 17 h for two NVIDIA Tesla V100 GPUs with 16G RAM to train.

Table 7. Data Augmentation Case. In this case, since there are lots of overlapping tokens (**‘interest’**, **‘account’**, which are in bold type) between the original positive and negative, it is hard for the model to judge whether the positive and negative are similar through term frequency. We calculate the similarity between the generated query and the original query, and the similarity score is lower than 0.1 (using cosine similarity), so we regard the original positive as the negative of the generated query and build a new triple.

Data Type	Text
original query	\$10,000 at 5% interest term deposit how much interest
original positive (new negative)	With simple interest , interest is only paid at the end of a specified term. A term deposit is an example of an account that will earn simple interest not compound interest . If you invested \$10,000 at 5% per year, you would earn \$2500 in simple interest after 5 years, \$500 for each year. If you invested \$10,000 at 5%, you would earn \$2834 in compound interest after 5 years, giving you a total of \$12,834.
original negative (new positive)	A full offset means that 100% of the funds in your offset account will be deducted from what you owe on your home loan before interest is calculated. A partial offset gives you a reduced interest rate on the part of your home loan equal to the balance of your offset account and while your money is working hard to reduce the interest you pay, your offset account will also be every bit as an everyday transaction account. This means you pay less interest on your home loan.
generated query (new query)	What is a full offset

5. Conclusions

The recent work focus on the two-stage Open-Domain QA system, improvements in both the retriever and the reader can improve the performance of the system. Our work tries to improve the reranker module in a retriever-reranker structure.

In this paper, to alleviate two problems in Open-domain QA: high-quality negatives selection and insufficient high-quality training data, we propose SS-BERT, a semantic information selecting method for Open-Domain QA passage rerank. We propose three methods to re-evaluate labeled data and create new labeled data: Negative selection by query generation, Data augmentation from negatives, and Passage evaluation. The experiments show that our methods are effective: among all the dual-encoder rerankers based on BERT-base and BM25, our proposed model performs best on MRR@10 (all the methods are effective) and Recall@50 and achieved the highest Recall@1000, which is limited by the BM25 retrieval result.

SS-BERT is a light model used for training a dual encoder, we use light pre-trained models (e.g., BERT-base, BART-base) to finish the dual-encoder training and query generation. We believe the absolute value of the model performance will significantly improve if we replace the base models with other state-of-art pre-trained models, which may cost much more computational resources. We plan to apply our method to train a cross-encoder and apply our method on other datasets in the future, which ask for much more computing resources (we have only two NVIDIA Tesla V100 GPUs with 16G RAM most of the time).

Author Contributions: Conceptualization and methodology, X.F., J.D., H.-T.Z. and J.L.; software and validation, X.F. and J.D.; writing—original draft preparation, X.F. and J.D.; writing—review and editing, X.F., J.D., H.-T.Z., J.L., C.H., Q.Z. and H.-G.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the National Natural Science Foundation of China (Grant No.62276154), AMiner.Shenzhen SciBrain Fund, Research Center for Computer Network (Shenzhen) Ministry of Education, Beijing Academy of Artificial Intelligence (BAAI), the Natural Science Foundation of Guangdong Province (Grant No. 2021A1515012640), Basic Research Fund of Shenzhen City (Grant No.JCYJ20210324120012033, JCYJ20190813165003837 and JSGG20210802154402007), and Overseas Cooperation Research Fund of Tsinghua Shenzhen International Graduate School (Grant No. HW2021008).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The MS-MARCO [7] dataset is available at <http://www.msmarco.org/dataset.aspx> (accessed on 1 January 2016) since 2016.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Green, B.F., Jr.; Wolf, A.K.; Chomsky, C.; Laughery, K. Baseball: An automatic question-answerer. In Proceedings of the Western Joint IRE-AIEE-ACM Computer Conference, Los Angeles, CA, USA, 9–11 May 1961; pp. 219–244.
2. Woods, W.A. Progress in natural language understanding: An application to lunar geology. In Proceedings of the National Computer Conference and Exposition, New York, NY, USA, 4–8 June 1973; pp. 441–450.
3. Mollá, D.; Vicedo, J.L. Question Answering in Restricted Domains: An Overview. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL), Prague, Czech Republic, 23–30 June 2007; pp. 41–61.
4. Chen, D.; Yih, W.T. Open-domain question answering. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts, Online, 5–10 July 2020; pp. 34–37.
5. Chen, D.; Fisch, A.; Weston, J.; Bordes, A. Reading Wikipedia to Answer Open-Domain Questions. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1870–1879.
6. Zhao, L. *Modeling and Solving Term Mismatch for Full-Text Retrieval*; Carnegie Mellon University: Pittsburgh, PA, USA, 2012.
7. Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; Deng, L. MS MARCO: A human generated machine reading comprehension dataset. *Adv. Neural Inf. Process. Syst. (NIPS)* **2016**, *2640*, 660.
8. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. Natural Questions: A Benchmark for Question Answering Research. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 453–466. [CrossRef]
9. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), Online, 5–10 July 2020; pp. 7871–7880.
10. Karpukhin, V.; Oğuz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; Yih, W.T. Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 30 September 2020; pp. 6769–6781.
11. Chang, W.C.; Yu, F.X.; Chang, Y.W.; Yang, Y.; Kumar, S. Pre-Training tasks for embedding-based large-scale retrieval. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts, Online, 5–10 July 2020; pp. 34–37.
12. Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M.W. REALM: Retrieval-Augmented Language Model Pre-Training. In Proceedings of the 37th International Conference on Machine Learning (ICML), Virtual Event, 13–18 July 2020.
13. Khattab, O.; Zaharia, M. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 25–30 July 2020; pp. 39–47.
14. Qu, Y.; Ding, Y.; Liu, J.; Liu, K.; Ren, R.; Zhao, W.X.; Dong, D.; Wu, H.; Wang, H. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 6–11 June 2021; pp. 5835–5847.
15. Ren, R.; Qu, Y.; Liu, J.; Zhao, W.X.; She, Q.; Wu, H.; Wang, H.; Wen, J.R. Open-domain question answering. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online, 7–11 November 2021; pp. 2825–2835.
16. Chen, D.; Yih, W. PAIR: Leveraging Passage-Centric Similarity Relation for Improving Dense Passage Retrieval. *arXiv* **2021**, arXiv:2108.06027.
17. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
18. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. *Adv. Neural Inf. Process. Syst. NIPS* **2014**, *27*.
19. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1724–1734.

20. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. *OpenAI* **2018**.
21. Nogueira, R.; Lin, J.; Epistemic, A.I. From doc2query to docTTTTTquery. *Tech. Rep.* **2019**, *6*, online preprint.
22. Nogueira, R.; Yang, W.; Lin, J.; Cho, K. Document expansion by query prediction. *arXiv* **2019**, arXiv:1904.08375.
23. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv* **2016**, arXiv:1910.10683.
24. Liang, D.; Xu, P.; Shakeri, S.; Santos, C.N.D.; Nallapati, R.; Huang, Z.; Xiang, B. Embedding-based zero-shot retrieval through query generation. *arXiv* **2022**, arXiv:2009.10270.
25. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-art Natural Language Processing. In Proceedings of the 2020 EMNLP (Systems Demonstrations), Online, 16–20 November 2020; pp. 38–45.
26. Lin, C.Y. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*; Association for Computational Linguistics: Barcelona, Spain, 2004; pp. 74–81.
27. Hermann, K.M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching machines to read and comprehend. *Adv. Neural Inf. Process. Syst. (NIPS)* **2015**, *28*.
28. Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL), Berlin, Germany, 11–12 August 2016; pp. 280–290.
29. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019; pp. 3982–3992.
30. Xiong, C.; Dai, Z.; Callan, J.; Liu, Z.; Power, R. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017.
31. Mitra, B.; Craswell, N. An updated duet model for passage re-ranking. *arXiv* **2019**, arXiv:1903.07666.
32. Mitra, B.; Diaz, F.; Craswell, N. Learning to Match using Local and Distributed Representations of Text for Web Search. In Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, Perth, Australia, 3–7 April 2019; pp. 1291–1299.
33. Hofstätter, S.; Rekabsaz, N.; Eickhoff, C.; Hanbury, A. On the effect of low-frequency terms on neural-IR models. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019.
34. Gao, L.; Dai, Z.; Callan, J. COIL: Revisit Exact Lexical Match in Information Retrieval with Contextualized Inverted List. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 6–11 June 2021; pp. 3030–3042.
35. Nogueira, R.; Yang, W.; Cho, K.; Lin, J. Multi-Stage Document Ranking with BERT. *arXiv* **2019**, arXiv:1910.14424.
36. Dai, Z.; Xiong, C.; Callan, J.; Liu, Z. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, Los Angeles, CA, USA, 5–9 February 2018.
37. Pang, L.; Lan, Y.; Guo, J.; Xu, J.; Wan, S.; Cheng, X. Text Matching as Image Recognition. In Proceedings of the Conference on Artificial Intelligence, New York, NY, USA, 9–15 July 2016.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.