future internet

MDPI

Article

Contrastive Refinement for Dense Retrieval Inference in the Open-Domain Question Answering Task

Qiuhong Zhai [1], Wenhao Zhu [1], Xiaoyu Zhang [1] and Chenyun Liu [2,*]

[1] School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China
[2] Shanghai Municipal Big Data Center, Shanghai 200444, China
* Correspondence: liuchenyun189@163.com

Abstract: In recent years, dense retrieval has emerged as the primary method for open-domain question-answering (OpenQA). However, previous research often focused on the query side, neglecting the importance of the passage side. We believe that both the query and passage sides are equally important and should be considered for improved OpenQA performance. In this paper, we propose a contrastive pseudo-labeled data constructed around passages and queries separately. We employ an improved pseudo-relevance feedback (PRF) algorithm with a knowledge-filtering strategy to enrich the semantic information in dense representations. Additionally, we proposed an Auto Text Representation Optimization Model (AOpt) to iteratively update the dense representations. Experimental results demonstrate that our methods effectively optimize dense representations, making them more distinguishable in dense retrieval, thus improving the OpenQA system's overall performance.

Keywords: dense retrieval; pseudo-reference feedback; pseudo-labels; semi-supervised learning

check for updates

Citation: Zhai, Q.; Zhu, W.; Zhang, X.; Liu, C. Contrastive Refinement for Dense Retrieval Inference in the Open-Domain Question Answering Task. *Future Internet* **2023**, *15*, 137. https://doi.org/10.3390/fi15040137

Academic Editors: Wei Emma Zhang, Chenliang Li and Michael Sheng

Received: 23 February 2023
Revised: 26 March 2023
Accepted: 30 March 2023
Published: 31 March 2023

1. Introduction

Since 2019, there has been a significant development in pre-trained language models, making dense retrieval [1–6] (which projects text into a low-dimensional mathematical space) more effective than traditional sparse retrieval (which is based on term retrieval) in OpenQA. The retrieval module is a crucial step in OpenQA and typically adopts a dual-encoder model architecture to learn dense representations of queries and passages. The dot product operation is then used to calculate the similarity between query-passage pairs for sorting purposes. Currently, many studies aim to improve this dense retrieval architecture; however, they have mainly focused on improving the query side, such as the model-level improvement [4,7] or instance-level improvement [8]. Despite these efforts, Sciavolino et al. [9] has experimentally demonstrated that DPR [1], which is a dense retrieval baseline, performed significantly worse than sparse retrieval on the entity-centric dataset EntityQuestions (EQ), as shown in Table 1. Their study indicated that DPR [1] could only improve questions with common entities and specific sentence templates, and text augmentation of queries alone could not solve this problem. Therefore, there is a need to build a more robust passage encoder.

Table 1. Retrieval results (Top-k) on NQ and EQ datasets.

Dataset	DPR-Single [1]	DPR-Multi [2]	BM25
NQ	80.1	79.4	64.4
EQ	49.7	56.7	72.0

[1] DPR model trained on NQ only. [2] DPR model trained on 4 datasets (NQ, TQA, WebQ, TREC) combined.

Based on the findings of Sciavolino et al. [9], we contend that the refinement of passage encoding is a crucial problem that needs to be addressed in open-domain question-answering (OpenQA) research. Thus, to enhance retrieval performance in such tasks, we propose optimizing the dense representations of both queries and passages. However, before proceeding with a specific implementation, we encountered two major challenges. Firstly, encoding millions of passages in the entire training set with a dense encoder can be compute-intensive and time-consuming (8.8 h on 8 64 GB GPUs). Secondly, as demonstrated by Sciavolino et al. [9], there is currently no universal and robust passage encoder that can effectively encode long sequence texts from different domains and contain the required semantic information.

This paper proposes a method for optimizing text representations based on the pseudo-relevance feedback (PRF) algorithm to address potential issues in retrieval. The proposed method constructs pseudo-labeled data with the query or passage as the center and builds a text representation optimization model called AOpt using the gradient descent algorithm to optimize the dense representations of queries and passages at the instance level. This approach avoids repeated text encoding, thus improving efficiency. To improve the general PRF algorithm, we use a knowledge-filtering strategy and conduct binary classification on query-passage pairs to obtain positive and negative pseudo-labels. Our positive and negative sampling strategy is based solely on whether the answer is contained in the passage, which is the most direct strategy. We believe that using the most relevant labels is beneficial for the AOpt model, as the text representation will be optimized through multiple iterations. After constructing the pseudo-labeled data, we first experiment with improving dense retrieval performance using a simple linear weight calculation method under different hyperparameters of pseudo-labeled data. This verifies the effectiveness of pseudo-labeled data in optimizing text representation. We then build the text representation optimization model AOpt to iteratively optimize text-dense representations. AOpt makes the relevant query-passage pairs closer in the mathematical space while making the irrelevant ones farther. Experimental results show that the optimized dense text representation effectively improves the performance of dense retrieval and the entire QA system in OpenQA.

Overall, the proposed method contributes to the field of passage retrieval by addressing the potential issues of existing methods and providing a more efficient and effective way of optimizing text representations.

2. Materials and Methods

We found that previous research on retrieval modules in OpenQA systems has mostly focused on improving the performance of dense retrieval from the query side in the question-answering data. However, Ren et al. [10] have experimentally shown that the other end of the question-answering data, the passage side, still contains a lot of unused information and knowledge. Therefore, we propose a pseudo-labeled data construction method based on contrastive relevance from both the query and passage sides. Specifically, we use a knowledge-filtering strategy in the OpenQA retrieval process to perform binary classification on “query-passage” pairs to distinguish between positive and negative relevance. Next, we use a simple linear weighted calculation method to demonstrate the effectiveness of this pseudo-labeled data for improving the retrieval and question-answering performance of OpenQA.

We then separately design a query-centric text representation optimization model, AOpt-query, and a passage-centric text representation optimization model, AOpt-passage. Their loss functions both aim to pull positive “query-passage” pairs closer and push negative “query-passage” pairs further apart. During the optimization process, we iteratively update the dense representation of the query or passage dense representations using the gradient descent algorithm.

2.1. Constructing Pseudo-Labels

Although DPR [1] achieved SOTA results at that time, it had several shortcomings, including its negative sampling strategies: (1) Random: this approach randomly sampled from the entire QA training corpus; (2) BM25: this method selected passages that contained most of the query tokens but did not include correct answers from the BM25 retrieval results;

and (3) Gold: this method selected gold passages (passages containing answers) from other queries in the mini-batch. Xiong et al. [7] demonstrated that (1) random sampling resulted in indistinguishable sample vector distances; (2) BM25-based negative sampling resulted in biased sparse retrieval, which failed to effectively improve the retrieval performance of the model, and (3) mini-batch negative sampling resulted in fewer informative negative samples due to the smaller batch size and the lower probability of obtaining effective negative samples in each mini-batch, making it difficult to achieve improved contrastive learning performance. To achieve better contrastive learning performance, we propose a semi-supervised pseudo-labeled data construction method that improves the negative sampling strategy in DPR [1].

Firstly, we still follow the dense retrieval step proposed in DPR [1] for retrieval. Specifically, we use a pre-trained model to encode all the queries and passage texts in the question-answering dataset to obtain their corresponding dense representations. Then, we construct an index of dense representations for all passages and use a dot-product calculation to obtain similarity scores for all "query-passage" pairs. After sorting the similarity scores, we take the top-k passages (usually, k is set to 100) as the most relevant passages for the current query. The question-answering retrieval step is shown in Figure 1.

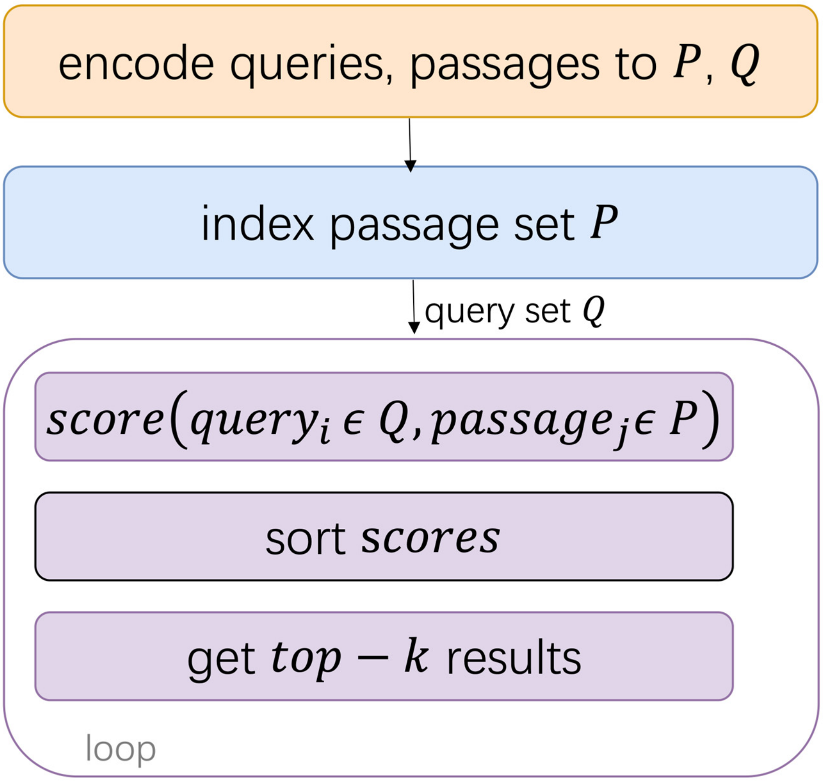

encode queries, passages to P, Q
index passage set P
query set Q
$score(query_i \in Q, passage_j \in P)$
sort $scores$
get $top-k$ results
loop

Figure 1. DPR Dense Retrieval Steps.

In the encoding stage, we used the pre-trained query encoder and passage encoder provided by DPR [1]. DPR [1] constructs positive and negative samples based on the negative sampling strategy mentioned in Section 2.1 and the traditional positive sampling strategy provided by the question-answering dataset. Two independent BERT-base-uncased pre-trained models are trained using these samples. Their training goal is to obtain a mathematical vector space that makes the distance between relevant "query-passage" pairs closer and the distance between irrelevant ones farther apart.

We used the query encoder E_Q and passage encoder E_P provided by DPR [1] to generate the dense representations of the queries and passages, which serve as the initial input for AOpt.

$$q = E_Q(query) \quad (1)$$

$$p = E_P(passage) \tag{2}$$

where *query* and *passage* denote the query text and the passage text, q and p are [CLS] token's hidden states in the last layer of BERT [11]. As shown in Figure 2, BERT [11] always adds a [CLS] token at the beginning of the input text before computation and encodes it together with the input text. Many previous studies [12] have demonstrated that the [CLS] token has a relatively effective representation in the final hidden state of pre-trained encoder models. This hidden state is a d-dimensional mathematical vector (usually d = 768), which can be used as a semantic representation of the entire input text. Therefore, the hidden state corresponding to the [CLS] token is commonly used as the final dense representation of the text.

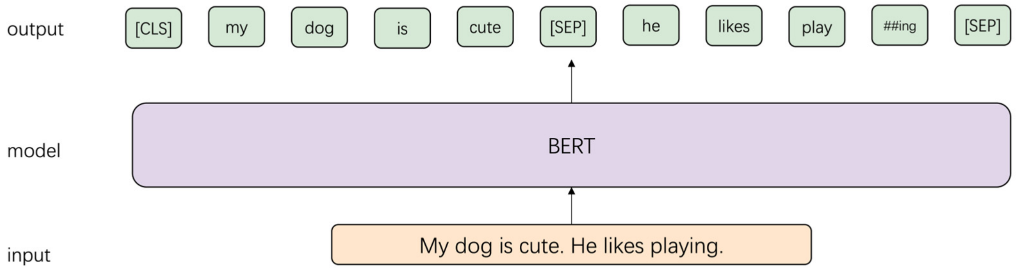


Figure 2. Embedding Generation Flow of BERT.

In the stage of computing the relevance scores, based on previous relevant experiments and considering both computational efficiency and effectiveness, we choose dot product operation to obtain the similarity scores of the "query-passage" pairs.

$$s(query, passage) = q{\cdot}p \tag{3}$$

To efficiently compute the relevance scores of "query-passage" pairs, we used the FAISS library to construct an index for all passage representations. The FAISS library is an efficient open-source library suitable for dense vector search and clustering, particularly for similarity search of millions of mathematical vectors. By building an index, the FAISS library greatly reduces the search time.

Specifically, given a query text q, we use its dense representation q to retrieve the top-k passage dense representations closest to it in the FAISS index. Finally, we rank the retrieval results obtained through FAISS in descending order of similarity scores to obtain the top-k passages most relevant to each query.

$$top - k_{p \in P}(query) = sort(s(query, context))[1:k] \tag{4}$$

Here, we introduce a PRF algorithm, which evolves from the relevance feedback (RF) algorithm. The RF algorithm is a classic and common algorithm in information retrieval task. It interacts with users to obtain feedback information on search results and uses this feedback information as auxiliary data to enhance the performance of the retrieval algorithm. The general step of the RF algorithm is to improve the query representation through some updating functions:

$$q_{t+1} = f(q_t) \tag{5}$$

q_t denotes the query representation after t updates; f denotes the update function.

Rocchio's algorithm is a classic RF algorithm, aiming to find the optimal query representation which maximizes the similarity of relevant query-passage pairs and minimizes the similarity of the irrelevant ones:

$$q_{opt} = \underset{q}{\text{argmax}}\left[sim\left(q, C^{+}\right) - sim\left(q, C^{-}\right)\right] \tag{6}$$

q denotes the previous query representation; C^+ denotes the relevant context of q; C^- denotes the irrelevant context of q, and q_{opt} denotes the optimal query representation after algorithm's updating.

However, the RF algorithm requires manual interaction and feedback, making it costly. In order to automate user interaction feedback, the PRF algorithm was proposed. A common method of PRF is to select the top k' passages from the top k retrieval results as the relevant passage set and consider the rest as the irrelevant set. The final query representation update is calculated in a weight calculation as Equation (7):

$$f(q_t) = \alpha * q_t + \beta * \frac{1}{|C^+|} * \sum\nolimits_{c^+ \in C^+} c^+ - \gamma * \frac{1}{|C^-|} * \sum\nolimits_{c^- \in C^-} c^- \tag{7}$$

C^+ and C^- respectively denote the relevant and irrelevant context of q_t, α, β, and γ are hyperparameters that determine different components.

Overall, the PRF algorithm is suitable for training based on large-scale unsupervised corpora, as it can easily achieve the automatic generation of feedback labels through certain settings. Therefore, we design an improved PRF algorithm with knowledge filtering based on answers, which divides the retrieval results into two contrastive labels, as shown in Equation (8).

$$\begin{cases} relevant,\ if\ a\ in\ passage_i \\ irrelevant,\ if\ a\ not\ in\ passage_i \end{cases} \tag{8}$$

a denotes the answer of the present query; $passage_i$ denotes the $i-th$ passage text in the Wikipedia dataset.

Equation (8) is the feedback generation step in our improved PRF algorithm, which refers to the findings of Xiong et al. [7] and employs a more practical method for determining relevance relations to generate pseudo-labels. In addition, considering that the AOpt model needs to iteratively optimize text representations with more reliable and strongly correlated sample pairs, we did not introduce other non-strongly correlated negative sample strategies, such as random sampling. The element in the passage-centric pseudo-labeled data we constructed can be referred to in Table 2, including the passage ID, passage text, passage title, positive query samples of the passage (positive section), and negative query samples of the passage (negative section).

Table 2. Example of constructed pseudo-labeled data.

Id: wiki: 14572616
Passage: The American Film Institute ranked season three one of the ten best television seasons of 2009. The Big Bang Theory (season 3). The third season of the American sitcom "The Big Bang Theory" was originally aired on CBS from 21 September 2009, to 24 May 2010, with 23 episodes. It received higher ratings than the previous two seasons with over 15 million viewers. Season three started three months after the end of season two when the guys left for the North Pole. The third season saw the first appearances of future main cast members Melissa Rauch as Bernadette
Title: The Big Bang Theory (season 3)
Positive Section: when do amy and bernadette come into the big bang theory {third season} when does amy come in big bang theory {The third season}
Negative Section: what is the cast of big bang theory paid {$1 million} when is the new episode of big bang theory airing {25 September 2017} where is the big bang theory show based {Pasadena, California}

2.2. *A Simple Linear Weight Calculation Method*

In this section, we propose a simple linear weight calculation method based on the pseudo-labeled data constructed in Section 2.1 to improve text representation by adjusting the weights of positive and negative samples. It should be noted that because of the simplicity and directness of this method, we only apply it to optimize passage representations in the experiments. The main purpose of the experiments is to demonstrate the effectiveness of the constructed pseudo-labeled data in improving text representation.

First, following the above steps, we construct passage-centric pseudo-labeled data for the QA dataset. Next, we use the passage encoder provided by DPR [1] to encode all the passage texts in the QA dataset and use the output dense representations as the initial input for the text representation calculation process, as shown in Equation (9).

$$p' = \alpha * p + \beta * pos\_q + \gamma * neg\_q \tag{9}$$

p denotes the passage representation before updating; p' denotes the passage representation after updating; pos_q is the averaged value of all relevant query representation of p; neg_q is the averaged value of all irrelevant query representation of p; α, β, and γ are hyperparameters that control the weights of the different components mentioned above. The whole updating process is shown in Figure 3.

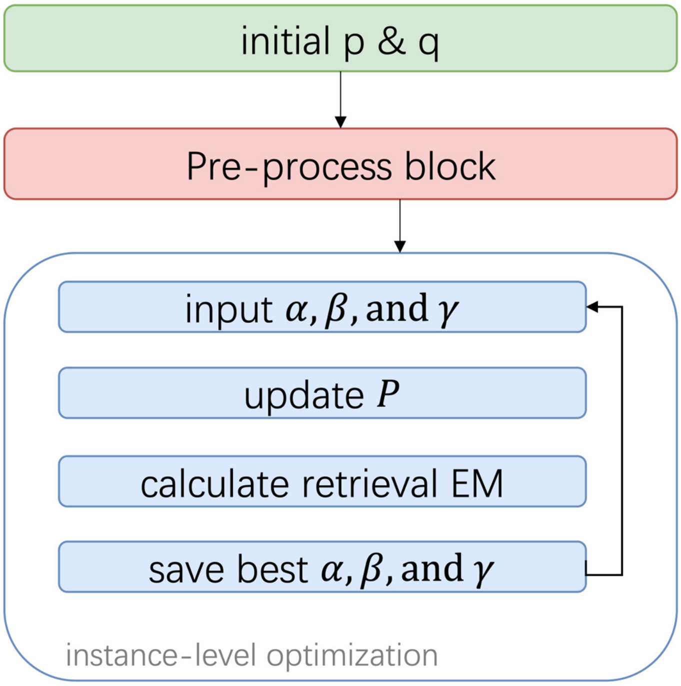

initial p & q
Pre-process block
input α, β, and γ
update P
calculate retrieval EM
save best α, β, and γ
instance-level optimization

Figure 3. Flowchart for Updating Text Representations Based on Linear Weight Calculation.

The "Pre-process block" shown in Figure 3 is detailed in Figure 4. This process is based on the retrieval and construction of pseudo-labeled data obtained from the current query representation and passage representation. Additionally, the "Pre-process block" also appears in the execution steps of the AOpt model (Figure 5).

We set different ranges for the hyperparameters α, β, and γ, then we test the retrieval improvement brought by the updated dense representations under all possible hyperparameter combinations. The results will be shown in Section 3.

Previous related studies based on PRF tended to construct relevance relations centered on queries and then use it to optimize query representations [8]. Differently, we research the relations of both query and passage by leveraging retrieval results and optimizing both of them. Theoretically, our method pays more attention to solving the above problems to make it more practical and explicable.

2.3. Text Representation Optimization Model AOpt

In this section, we have designed two text representation optimization models: AOpt-query and AOpt-passage. These models focus on the query side and passage side, respectively, and automatically refine their dense representations. Both models use the same logic to calculate loss, aiming to bring positive "query-passage" pairs closer and push negative pairs farther apart. Specifically, the AOpt-query model adjusts the proportion of positive and negative passage representations in the optimization process of query representations, while the AOpt-passage model performs corresponding operations based on passage representations.

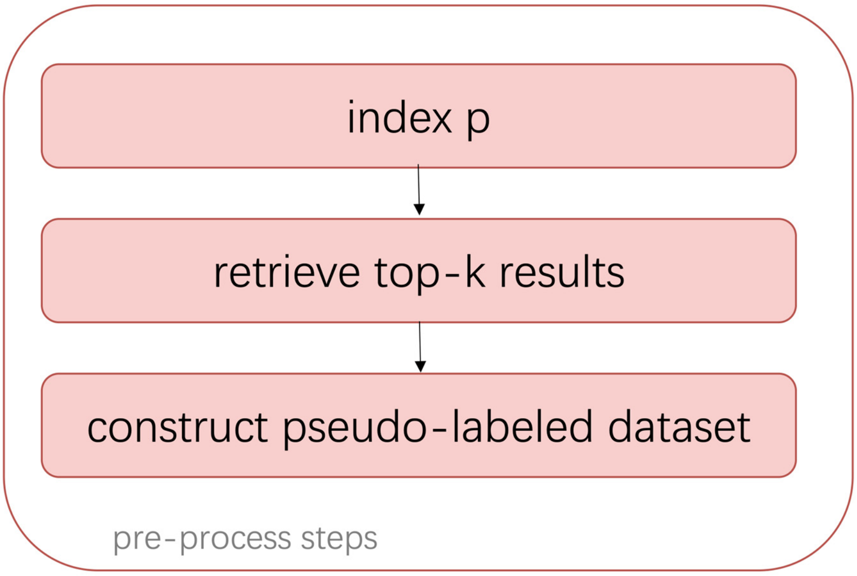


Figure 4. Pre-process Steps.

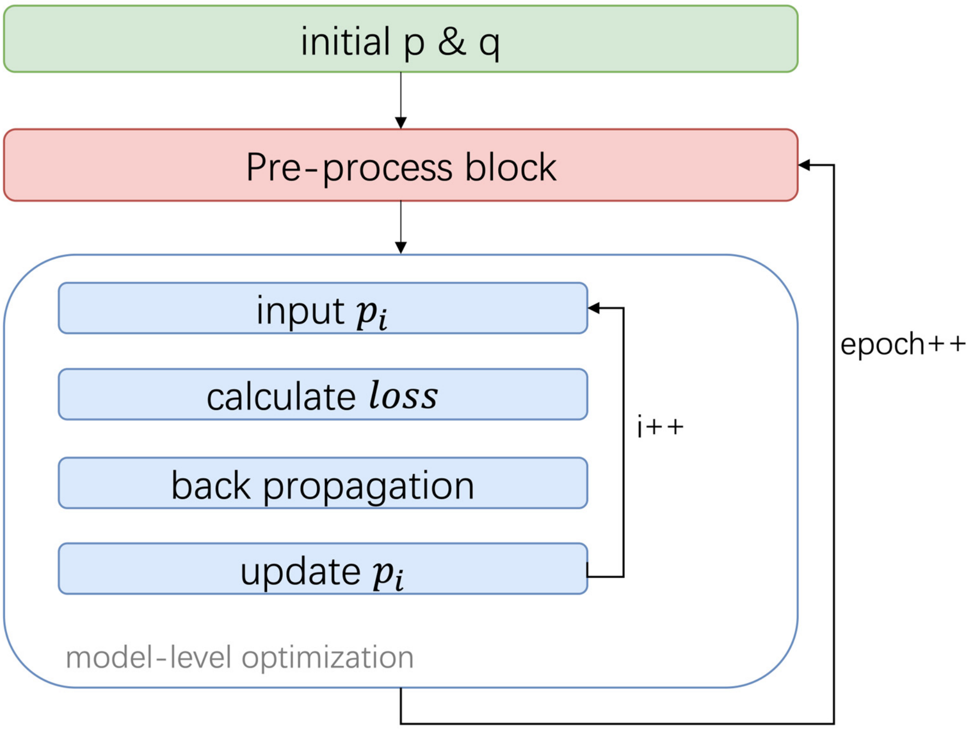


Figure 5. Flowchart for AOpt Model.

2.3.1. AOpt-Query LOSS

The query-centric loss considers the query representation q as the center and pulls the relevant passage representations pos_p closer while pushing the irrelevant passage representations neg_p farther:

$$s_q(q, pos\_\mathrm{p}) > s_q(q, neg\_\mathrm{p}), \tag{10}$$

where $s_q(q, pos\_\mathrm{p})$ denotes the similarity for relevant passages representations pos_p to the query representation q; $s_q(q, neg\_\mathrm{p})$ denotes the similarity for irrelevant passages representations neg_p to query representation q. We learn each query-centric similarity relation by backpropagating its loss with the gradient descent algorithm:

$$l_{q-centric} = -\log \sum\nolimits_i^{|pos\_\mathrm{p}|} \frac{s(q, p_i)}{\sum_j^{|\mathrm{pos\_p}\ +neg\_\mathrm{p}|} s(q, p_j)}, \tag{11}$$

p_i denotes the $i - th$ relevant passage representation for the current query; p_j denotes the $j - th$ query representation in the union of relevant and irrelevant passage representation set for the current passage; $\mathrm{s}(q, p_i)$ denotes the similarity score of q and p_i calculated with the dot product operation.

2.3.2. AOpt-Passage Loss

The passage-centric loss considers the passage representation p as the center and pulls the relevant query representations pos_q closer while pushing the irrelevant query representations neg_q farther:

$$s_q(p, pos\_\mathrm{q}) > s_q(p, neg\_\mathrm{q}) \tag{12}$$

Similarly, we learn each passage-centric similarity relation as follows:

$$l_{p-centric} = -\log \sum\nolimits_i^{|pos\_q|} \frac{s(p, q_i)}{\sum_j^{|\mathrm{pos\_}q + neg\_q|} s(p, q_j)}, \tag{13}$$

We can find that the query-centric and passage-centric losses are similar in logic but different in the choice of the center component.

2.3.3. Dense Representations Update

After performing backpropagation, we can get the gradient of the current center component. Taking AOpt-passage as an example, we use $\frac{\partial l_{p-centric}}{\partial \mathrm{p}}$, which is the gradient of the current passage representation p to update itself:

$$p_{i+1} = p_i - \eta * \frac{\partial l}{\partial p}, \tag{14}$$

p_i denotes the passage representation before the $i - th$ update; p_{i+1} denotes the passage representation after the $i - th$ update; η denotes the learning rate.

In AOpt-passage, after updating all the dense representations of passages one by one (according to the design of the loss function, the batch size is set to 1, so each batch updates one passage representation), we use the FAISS library to rebuild the index for the updated passage representation set and re-execute the retrieval process, then reconstruct the pseudo-labels centered on the passage, and finally, optimize all passage representations again. The above process will be repeated n times (where n is the set number of rounds), as shown in Figure 5. The optimization of text representations will end when the loss function value does not decrease for five consecutive epochs, triggering an early stopping mechanism.

3. Results

In this section, we describe our experimental settings, experimental results, and related analysis.

3.1. Experimental Preparations

3.1.1. Dataset

1. Wikipedia Dataset

As Karpukhin et al. [1], we used Wikipedia as the knowledge source for OpenQA. They used the preprocessing code provided in DrQA [13] to remove semi-structured data, such as tables, infoboxes, and lists, from the documents and divided each document into multiple disjoint 100-token passages (a total of 21,015,324 passages), which served as the basic retrieval unit. In our retrieval experiments, we used the pre-processed passage set provided by Karpukhin et al. [1].

2. QA Dataset

Similar to GAR [14], we conducted experiments on the test sets of two common OpenQA datasets: NQ [15] and TriviaQA [16].

- NQ

NQ is designed for end-to-end question-answering tasks, where the questions are from real Google search logs, and the answers are manually annotated in Wikipedia documents as text spans.

- TriviaQA

The TriviaQA dataset is a challenging dataset with complex questions that require more cross-sentence reasoning to obtain the answers. The questions and corresponding answer sentences in this dataset have a significant amount of syntactic or lexical variation, making answering the questions more difficult.

3.1.2. Evaluation Metrics

Similar to DPR [1], we use top-k accuracy to evaluate the retrieval performance in OpenQA and use exact match to measure the reader performance.

- Top-k Accuracy

Top-k accuracy refers to the proportion of queries that have at least one answer passage in the top-k relevant passages returned by the retriever. This value represents an upper limit on the number of queries that can be answered by subsequent extractive readers.

- Exact Match

Exact match refers to the proportion of predicted answers that are identical to the reference answers provided in the question-answering dataset. Before comparing, the answers need to be normalized by removing articles (such as "the", "a", and "an") and punctuation marks, etc.

3.1.3. Experimental Details

1. Device

We conducted experiments on 4 NVIDIA Tesla V100 GPUs with 32 GB RAM each.

2. Main Libraries
    - We used the PyTorch deep learning framework for our experiments.
    - Similar to DPR [1], we used the HNSW index from the FAISS-cpu library for retrieval experiments, with 512 neighbors stored for each node.

3. Retriever and Reader

Similar to Ma et al. [17], we used Retriever$_{\text{NQ}}$ and Reader$_{\text{NQ-Single}}$ for dense encoding and answer extraction on NQ and Retriever$_{\text{Multi}}$ and Reader$_{\text{TQA-Multi}}$ on TriviaQA.

3.1.4. Hyperparameters

- Weight-Based Calculation

α, β, and, γ respectively denote the weight of the original passage representation, the weight of the averaged relevant query representations and the weight of the averaged irrelevant query representations. Following the previous conclusion, we set the relevant component weight β from 0 to 0.9 with a search interval of 0.1 and set the irrelevant component weight γ from 0 to −0.9 with a search interval of −0.1. As the basis, we fix the weight of the original passage representation to 1.

- AOpt Model

We set the training batch size to 1; namely, every single dense representation will be processed alone for a forward and backward propagation step during every epoch. We set the number of training epochs to a maximum of 100 and set the early stop to be triggered if the loss does not decrease for up to five consecutive epochs. We set the learning rate η to 0.1 to obtain effective results.

3.2. Experimental Results

We first verified the effectiveness of the proposed pseudo-labeled data by conducting a simple linear weighted calculation on the NQ dataset. Next, we conducted experiments on both the NQ and TriviaQA datasets to compare the performance of the AOpt-query and AOpt-passage text representation optimization models proposed in this paper, as well as their combination, in the retrieval module and the entire QA system.

3.2.1. Linear Weight Calculation Result

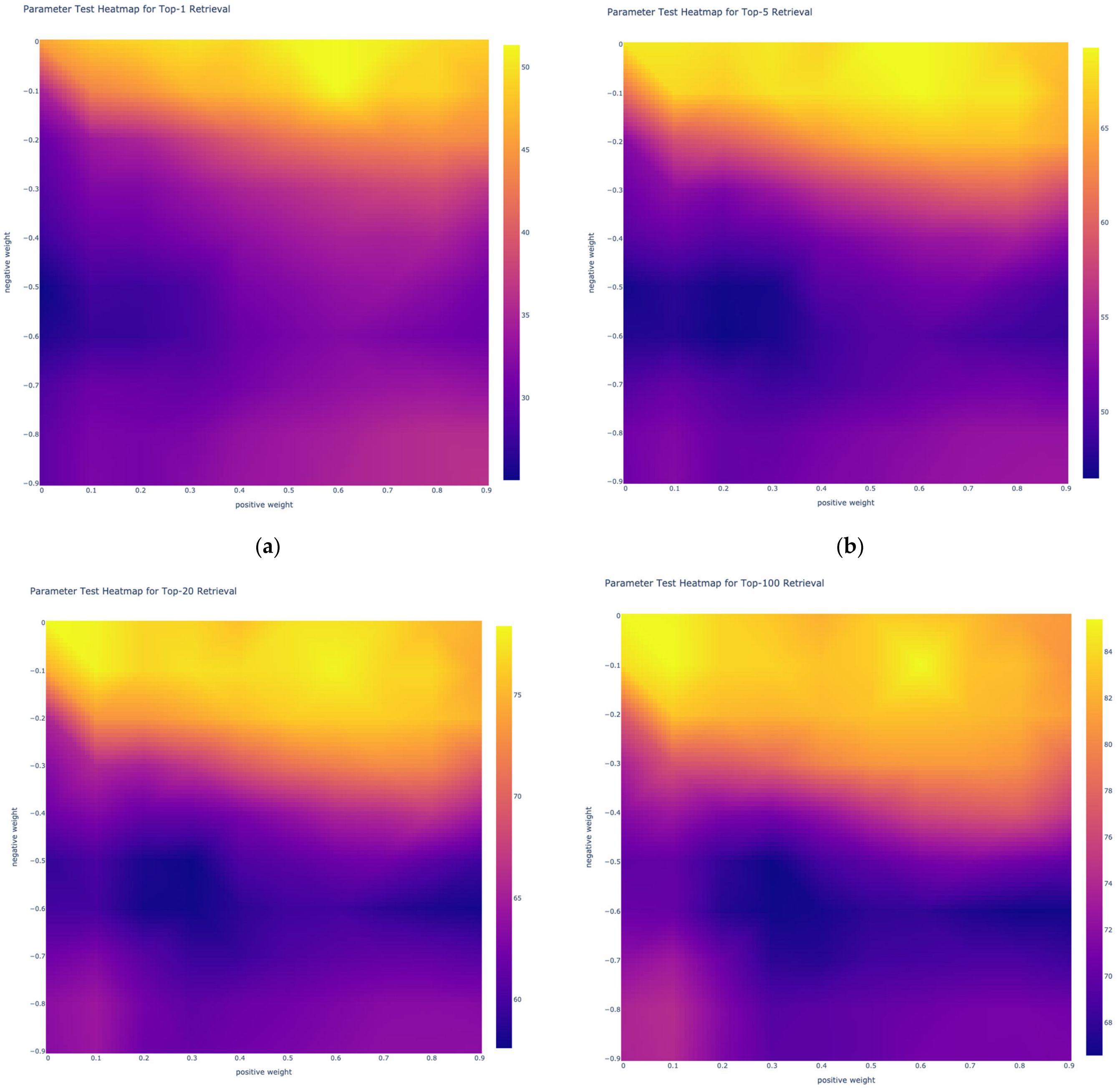
For the experiment of linear weight calculation of pseudo-labels, we first need to find out the best hyperparameters through retrieval experiments with weight combinations within the specified range. Following the above hyperparameter settings, we calculated the retrieval performance for all possible combinations of β and γ on NQ and plotted the heat map shown in Figure 6 for retrieval accuracy from Top-1 to Top-100.

In Figure 6, the lighter the color, the higher the top-k accuracy. With Table 3, we found that the optimal weight combination for the proposed pseudo-labeled data construction method in OpenQA task retrieval experiments is β = 0.6, γ = −0.1, which can improve the Top-1 retrieval performance by about 5.9% on the NQ dataset.

3.2.2. Retriever Performance

After demonstrating the effectiveness of our proposed pseudo-labeled data in improving OpenQA retrieval performance through simple linear weighting, we applied this data to our text representation optimization model AOpt. The AOpt-query model is used to optimize the dense representation of queries, while the AOpt-passage model is used to optimize the dense representation of passages. We conducted retrieval experiments on the NQ and TriviaQA datasets for three cases: "optimizing query representations only"; "optimizing passage representations only"; and "optimizing query and passage representations sequentially".

As shown in Table 3, we compared our proposed AOpt model with baseline models and representative models in recent years for OpenQA tasks and found that the AOpt model improved the retrieval performance on the NQ dataset, especially on the Top-1 accuracy, achieving a significant improvement of 11.4% compared to the baseline model DPR. In addition, the AOpt model achieved the best retrieval performance on the TriviaQA dataset, with a 12.1% improvement in the Top-1 accuracy compared to the DPR baseline model.



(**a**)

(**b**)

(**c**)

(**d**)

Figure 6. These are weight combinations experimental results (on NQ dataset) heat map, whose x-axis denotes relevant query representations' weight β and y-axis denotes irrelevant query representations' weight γ. The retrieval performance (EM) results are as follows: (**a**) Top-1; (**b**) Top-5; (**c**) Top-20; (**d**) Top-100.

Specifically, we found that the retrieval performance of AOpt-query was slightly higher than AOpt-passage, with a maximum difference of 3.0% (Top-5 accuracy on the NQ dataset). We believe this indicates that optimizing query representations is more effective than optimizing passage representations under the same scale.

Table 3. Retrieval of Top-k Results.

	NQ				TriviaQA			
Model	**Top1**	**Top5**	**Top20**	**Top100**	**Top1**	**Top5**	**Top20**	**Top100**
BM25	-	-	59.1	73.7	-	-	66.9	76.7
DPR(single [1])	-	-	78.4	85.4	-	-	79.4	85.0
DPR(multi [2])	-	-	79.4	86.0	-	-	78.8	84.7
Hybrid [3](single)	-	-	76.6	83.8	-	-	79.8	84.5
Hybrid(multi)	-	-	78.0	83.9	-	-	79.9	84.4
GAR	-	60.9	74.4	85.3	-	73.1	80.4	85.7
ANCE(single)	-	-	81.9	87.5	-	-	80.3	85.3
ANCE(multi)	-	-	82.1	87.9	-	-	80.3	85.2
PAIR	-	74.9	83.5	89.1	-	-	-	-
DPR *	45.7	68.3	-	-	47.2	72.7	-	-
Weighted-based ($\beta = 0.6$, $\gamma = -0.1$)	51.6	69.6	78.2	84.3	-	-	-	-
DPR + AOpt(query)	52.9	75.1	82.1	86.2	55.4	80.3	83.3	86.0
DPR + AOpt(passage)	51.3	72.1	80.4	85.9	53.0	77.3	81.1	85.6
DPR + AOpt(hybrid) [4]	57.1	74.5	81.2	85.4	59.3	79.9	82.3	84.9

[1] trained on an individual training dataset. [2] trained on combined training datasets (all except SQuAD). [3] retrieval with both BM25 and DPR. [4] text representation optimization with both AOpt(query) and AOpt(passage). * our replication result under DPR [1] experimental settings.

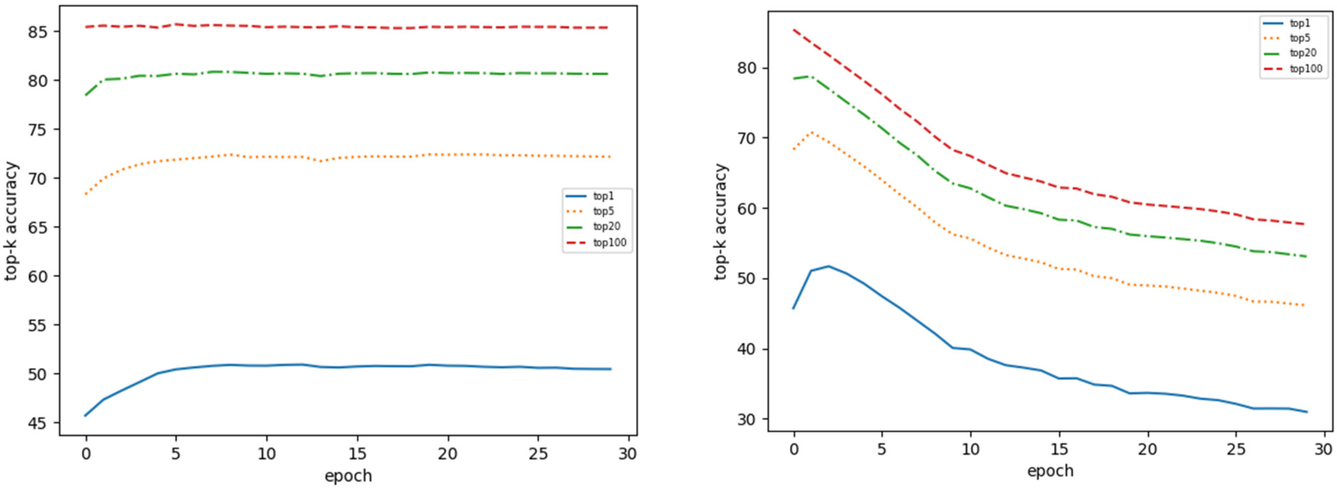
Although optimizing queries can achieve relatively higher significant results, the data in Table 3 for "DPR + AOpt(hybrid)" shows that optimizing both query and passage representations can achieve even bigger improvement. On the NQ dataset, it improved by 4.2% compared to only optimizing query representations and 5.8% compared to only optimizing passage representations. On the TriviaQA dataset, it improved by 3.9% compared to only optimizing query representations and 6.3% compared to only optimizing passage representations.

Furthermore, according to Table 3, we found that DPR with AOpt led to a significant improvement in retrieval performance, especially in Top-1 accuracy. As the k value increases, the retrieval improvement starts to decrease. We believe that this phenomenon is caused by our pseudo-labeled data construction strategy, which strengthens the mathematical correlation between query representation and passage representation through several iterations. When retrieving from a large-scale set of passages, our dense representation optimization strategy makes the retriever focus more on the most recent relevant passages that contain the current query answer. Therefore, in the iterative retrieval process, relevant passages will obtain higher-ranking results. In other words, this method can achieve results comparable to adding an additional re-ranker module.

During the iterative epochs of the AOpt model, we found that AOpt-passage and AOpt-query resulted in different trends in retrieval performance, as shown in Figure 7.

Based on the two trends shown in Figure 7, we conduct the following observations: (1) the retrieval performance change curve using the AOpt-passage model is stable and gradually tends to flatten; (2) the retrieval performance change curve using the AOpt-query model quickly reaches its peak and then rapidly declines.

We believe that this is due to the significant difference in sequence length between queries and passages. In the QA dataset, the query sequence length is generally less than 15 tokens, while the passage sequence length is around 100 tokens. Therefore, after the same computation, the AOpt model will update query representation more thoroughly over a larger range, which explains why query representation can show relatively large improvements and steeper trends in retrieval recall. However, due to the relatively longer text length of passage sequences, it is more difficult to achieve precisely encoded text, and they often lack some semantic information. Therefore, passage representation can absorb additional relevant semantic information from relevant and irrelevant query representations during the AOpt optimization process while still retaining most of its original encoding information. In other words, if query representations are continually updated, similar to passage representations, the original information in query representations will be completely covered, thereby losing its own representational ability. Therefore, for query representation optimization, we need to determine the best training rounds and stop optimizing query representation in advance.



(**a**) (**b**)

Figure 7. AOpt Retrieval Performance Trends, which are (**a**) passage-centric and (**b**)query-centric.

3.2.3. Reader Performance

We evaluated the final question-answering exact match of the reader by inputting retrieval results from NQ and TriviaQA datasets. Our AOpt model was compared to the baseline and representative models in OpenQA, and the results are presented in Table 4. The analysis shows that our AOpt model achieved the best performance on the NQ dataset, with a maximum improvement of 5.7% when compared to the baseline model DPR [1]. Additionally, the exact match on TriviaQA also demonstrated an improvement of 4.3% when compared to DPR [1].

Table 4. QA Exact Match Results.

Model	**NQ**	**TriviaQA**
BM25	32.6	52.4
DPR	41.5	56.8
Hybrid	39.0	57.9
GAR	45.3	62.7
ANCE	46.0	57.5
DPR + AOpt(query)	43.9	58.9
DPR + AOpt(passage)	43.3	58.5
DPR + AOpt(hybrid)	47.2	61.1

4. Conclusions and Future Work

In this paper, we begin by constructing contrastive pseudo-labeled data using question-answering retrieval results. Next, we propose two models for optimizing dense representations: AOpt-query and AOpt-passage. They focused on queries and passages, respectively, and used a gradient descent algorithm to refine dense representations at the instance level. Experimental results show that the AOpt model can effectively optimize the dense representations, resulting in improved retrieval and question-answering performance of the OpenQA system. The highest improvements observed are 12.1% and 5.7%, respectively. In future work, we plan to research and design a specific reader model to further enhance the performance of the entire OpenQA system.

Author Contributions: Conceptualization, Q.Z. and W.Z.; methodology, Q.Z. and C.L.; software, Q.Z. and C.L.; validation, Q.Z. and X.Z.; formal analysis, W.Z. and C.L.; investigation, X.Z.; resources, Q.Z.; data curation, Q.Z.; writing—original draft preparation, Q.Z.; writing—review and editing, Q.Z.; visualization, Q.Z.; supervision, W.Z.; project administration, C.L.; funding acquisition, W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 61572434 and No. 91630206), the Shanghai Science and Technology Committee (No. 19DZ2204800) and the National Key R&D Program of China (No. 2017YFB0701501).

Data Availability Statement: Data sharing is not applicable to this article due to privacy.

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

References

1. 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 (Volume 1: Long Papers), Vancouver, QC, Canada, 30 July–4 August 2017; Association for Computational Linguistics: Stroudsburg, PA, USA, 2017; pp. 1870–1879.
2. Robertson, S.; Zaragoza, H. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends® Inf. Retr.* **2009**, *3*, 333–389. [CrossRef]
3. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2018**, arXiv:1810.04805.
4. Karpukhin, V.; Oğuz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; Yih, W. Dense Passage Retrieval for Open-Domain Question Answering. *arXiv* **2020**, arXiv:2004.04906.
5. Zhu, W.; Liu, S.; Liu, C. Learning multimodal word representation with graph convolutional networks. *Inf. Process. Manag.* **2021**, *58*, 102709. [CrossRef]
6. Zhu, W.; Jin, X.; Liu, S.; Lu, Z.; Zhang, W.; Yan, K.; Wei, B. Enhanced double-carrier word embedding via phonetics and writing. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2020**, *19*, 1–18. [CrossRef]
7. Xiong, L.; Xiong, C.; Li, Y.; Tang, K.-F.; Liu, J.; Bennett, P.; Ahmed, J.; Overwijk, A. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. *arXiv* **2020**, arXiv:2007.00808.
8. Ren, R.; Lv, S.; Qu, Y.; Liu, J.; Zhao, W.X.; She, Q.; Wu, H.; Wang, H.; Wen, J.-R. PAIR: Leveraging Passage-Centric Similarity Relation for Improving Dense Passage Retrieval. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online, 1–6 August 2021; pp. 2173–2183.
9. Sung, M.; Park, J.; Kang, J.; Chen, D.; Lee, J. Optimizing Test-Time Query Representations for Dense Retrieval. *arXiv* **2022**, arXiv:2205.12680.
10. Lee, K.; Chang, M.-W.; Toutanova, K. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 6086–6096.
11. Lee, J.; Sung, M.; Kang, J.; Chen, D. Learning Dense Representations of Phrases at Scale. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 5–6 August 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 6634–6647.
12. Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M.-W. REALM: Retrieval-Augmented Language Model Pre-Training. *arXiv* **2020**, arXiv:2002.08909.
13. Liu, C.; Zhu, W.; Zhang, X.; Zhai, Q. Sentence part-enhanced BERT with respect to downstream tasks. *Complex Intell. Syst.* **2023**, *9*, 463–474. [CrossRef]
14. Sciavolino, C.; Zhong, Z.; Lee, J.; Chen, D. Simple Entity-Centric Questions Challenge Dense Retrievers. *arXiv* **2022**, arXiv:2109.08535.
15. Lee, D.-H. Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. In Proceedings of the ICML 2013 Workshop on Challenges in Representation Learning (WREPL), Atlanta, GA, USA, 17–19 June 2013.
16. 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]
17. Ma, X.; Sun, K.; Pradeep, R.; Lin, J. A replication study of dense passage retriever. *arXiv Prepr.* **2021**, arXiv:2104.05740.

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.