# Improving the Accuracy of Dense Retrieval on the Quantized Indexes via Gradient Optimization of the Target Embeddings

**Cong Tan, Yongqi Shao, Hong Huo, Tao Fang**

Shanghai Jiao Tong University
coign@sjtu.edu.cn, cici_syq@sjtu.edu.cn, huohong@sjtu.edu.cn, tfang@sjtu.edu.cn

## Abstract

Dense retrieval models commonly use flat indexes to achieve high-precision retrieval by computing exact distances between embedding vectors. However, flat indexes are memory-intensive and inefficient, limiting their scalability in large-scale retrieval tasks. In contrast, quantized index enables faster retrieval with significantly lower memory usage, but their accuracy tends to decrease. Therefore, we propose a scalable and efficient training method for the dual-encoder models to improves the retrieval accuracy on quantized index. Our approach combines the direct gradient update to the cached target embeddings with large scale negative sampling based on similarity, significantly reducing computational overhead and GPU memory usage. Target embeddings are initialized with a pre-trained encoder and stored in a memory buffer, which is directly updated via backpropagation, thus avoiding the repeated re-encoding of the full corpus. To build a rich set of negatives, we retrieve the top-k most similar targets for each query from cached embeddings using the quantized index, including both query-specific and cross-batch top-k results. This design effectively approximates the softmax distribution. The experiments show that our method achieves performs exceptionally well on quantized index, providing a practical and scalable solution for real-world retrieval systems.

**Code** — https://github.com/cgmta/dback

## Introduction

Efficient retrieval of relevant information from large document corpora is a critical task in modern retrieval-based systems, underpinning applications such as open-domain question answering, search ranking, and retrieval-augmented generation (RAG) (Kwiatkowski et al. 2019; Xu et al. 2024; Zhao et al. 2024a). The quality of retrieved results directly influences downstream performance, so the accuracy and efficiency of retrieval are crucial for large-scale deployments. Dense retrieval has gained increasing attention as it leverages continuous vector embeddings learned by deep neural networks, enabling more flexible and semantically rich document matching (Zhao et al. 2024b).

In this context, Karpukhin et al. (Karpukhin et al. 2020) introduced dual-encoder models, where the query and the candidate documents are encoded independently, and their relevance is determined by the dot product between their embeddings. Since computing softmax logits over all possible targets is computationally infeasible, training typically relies on approximations such as truncated softmax and negative sampling (Reddi et al. 2019; Lindgren et al. 2021). Updating the cache of target embeddings in a fixed or periodic manner may become stale and degrade the training effectiveness. Moreover, most dense retrievers are trained and evaluated with flat indexes (Douze et al. 2024; Bornea et al. 2024), which compute precise embedding distances and achieve high accuracy. However, these approaches incurs substantial memory and computational costs, making them unsuitable for large-scale retrieval.In contrast, the quantized index (Douze et al. 2024; Danopoulos, Kachris, and Soudris

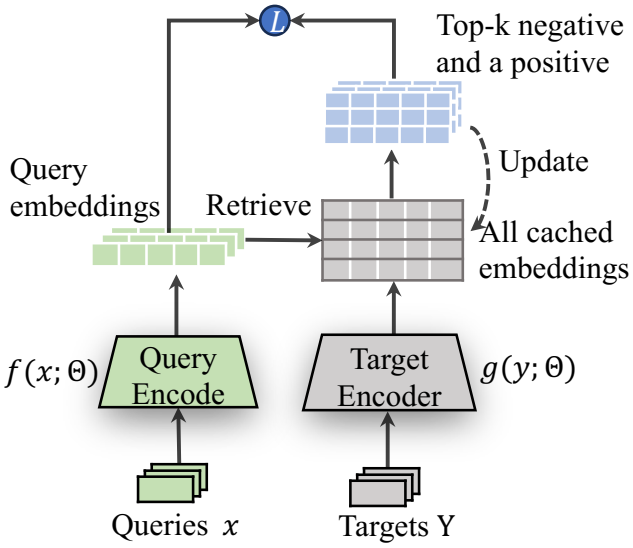


Figure 1: The proposed overall training method. We use a pre-trained $g$ to embed all target texts $\mathcal{Y}$. For any given query $x$, the $f$ produces a query embedding. From the cached target embeddings, the top-$k$ most similar negative samples are selected. During training, the target embeddings are directly updated based on the gradients.

2019) significantly reduces memory usage and latency, but suffers from degraded accuracy if the model is not trained under the condition of quantized retrieval.

To address this trade-off, we propose a scalable and efficient training method for dual-encoder models, which can significantly improve the retrieval accuracy on quantized index. We initialize a memory buffer containing all the target embeddings using a pre-trained encoder, and these cached embeddings are directly updated via gradient descent during training. Then, we construct and periodically refresh the quantized index over the cached embeddings. At each training step, we leverage the quantized index to select the top-$k$ most similar negative targets for each query. These negative targets are combined with in-batch top-$k$ negatives to form a comprehensive set of negative samples, where $k$ can be large to better approximate the actual softmax distribution.

Our main contributions include: 1. Direct gradient updates to the cached target embeddings avoids redundant computations while maintaining the up-to-date representations. 2. A large-scale negative sampling strategy based on similarity combines top-$k$ nearest targets from both the current query and in-batch samples to better approximate the softmax distribution. 3. Empirical gains in retrieval accuracy under quantized index enable faster retrieval and reduced memory usage in large-scale retrieval tasks.

## Related Work

**Dense Retrieval Methods** Recent advances in dense retrieval have significantly improved the effectiveness of information retrieval systems. Dense Passage Retrieval (DPR) (Karpukhin et al. 2020) pioneered the use of dense vector embeddings for query and passage matching. Subsequent works have focused on improving the efficiency and effectiveness of dense retrievers through better training methods. For instance, RocketQA (Qu et al. 2021) and co-Condenser (Gao and Callan 2021) introduced more efficient training approaches, which can reduce computational costs while maintaining strong retrieval performance. Despite these advances, training dense retrieval models on large-scale corpora remains challenging due to computational requirements (Arabzadeh, Yan, and Clarke 2021; Monath et al. 2024).

**Negative Sampling** The negative sampling is vital for training dense retrieval models. Traditional sampling methods, such as random and batch-based sampling, may limit the diversity of negative examples. Approaches such as using BM25 (Robertson, Zaragoza et al. 2009) to select "hard" negatives (Karpukhin et al. 2020) can improve the sample quality, but still have certain limitations. Approximate Nearest-Neighbor Contrastive Learning (ANCE) (Xiong et al. 2020) enhances this approach by selecting hard negatives based on similarity, thereby improving the informativeness of negative samples. Dynamic Indexes (Monath et al. 2023) utilize tree structures for more efficient negative mining, while methods such as TriSampler (Yang et al. 2024) further improve performance by generating more informative negatives.

**Target Embedding Caching and Indexing** During training, in order to avoid recalculating embeddings for every query, methods such as ANCE (Xiong et al. 2020) and Corrector Networks (Monath et al. 2024) are often used to construct target embedding caches. In the passage retrieval tasks (Sachan et al. 2022), the cached target embeddings can also accelerate retrieval by enabling fast lookup during query processing. Various indexing methods are employed to balance retrieval speed and memory usage, including exact search for L2 distances (Norouzi, Punjani, and Fleet 2013), Hierarchical Navigable Small World (HNSW) graphs (Lin and Zhao 2019), and inverted file indexes with post-verification (Kukreja et al. 2023). Product Quantization (PQ) (Thakur, Reimers, and Lin 2022) is also commonly used to compress index size at the expense of retrieval accuracy, offering a scalable solution for large datasets with limited memory.

## Preliminaries

This section introduces the dual-encoder architectures in dense retrieval models, the softmax function used for ranking relevance, as well as techniques such as truncated softmax and top-$k$ sampling that make large-scale training more efficient.

**Dual-Encoder Architecture** In dense retrieval models, a dual-encoder architecture is typically used to compute the unnormalized logits, $s_{x,y}$, by factorizing the embeddings of the query and the target. Specifically, each input (the query $x$ and the target $y$) is encoded into a $D$-dimensional embedding through deep neural networks. The query $x$ is encoded by a function $f(x;\Theta)$, and the target $y$ is encoded by a function $g(y;\Theta)$. The logits are then computed as the dot product between the query and target embeddings:

$$s_{x,y} = \langle f(x;\Theta), g(y;\Theta) \rangle, \tag{1}$$

where $\Theta$ denotes the parameters of the neural networks and $\langle \cdot, \cdot \rangle$ denotes the dot product operation. This architecture is highly effective in querying and learning the semantic embeddings of targets. It helps to accomplish retrieval tasks by ranking the relevance of candidate passages based on the similarity between their embeddings.

**Softmax Function** Given the dual-encoder model, we define a probability distribution over a set of $N$ targets, denoted as $\mathcal{Y}$, based on the similarity between the query $x$ and the candidate target $y$. The set $\mathcal{Y}$ represents the full collection of all possible candidate targets that the model considers during inference or training—typically, this could be a large corpus such as all passages in Wikipedia. The softmax function is commonly used to compute this distribution in the following manner:

$$\begin{aligned} P(y|x) &= \frac{\exp(\beta s_{x,y})}{Z_x} \\ &= \frac{\exp(\beta s_{x,y})}{\sum_{y' \in \mathcal{Y}} \exp(\beta s_{x,y'})}, \end{aligned} \tag{2}$$

where $\beta$ is inverse temperature that controls the smoothness of the distribution, and $s_{x,y}$ is the unnormalized logit

of the target $y$ given the query $x$. This formula is widely used in retrieval tasks, where $x$ is a query and the targets $y$ are candidate passages. For example, in the Natural Questions dataset (Kwiatkowski et al. 2019), $x$ is a question, and $\mathcal{Y}$ refers to the entire corpus of Wikipedia passages that may contain relevant answers. The softmax assigns a probability to each target based on its relevance to the query. This is very useful in tasks such as information retrieval, where the goal is to rank passages according to the relevance to the given query.

**Training with Truncated Softmax** In the dual-encoder model, training typically involves optimizing the loss function for a specific task, such as cross-entropy, through the gradient descent (Rawat et al. 2019). This process requires computing the softmax distribution over all possible targets. However, this is computationally expensive due to the massive number of candidate targets (usually in the millions or more). The exact computation of the normalization constant $Z_x$—which requires summing over all targets—becomes intractable during training.

To address this problem, the softmax function is approximated by a truncated version, $\tilde{P}(y|x)$, which considers only a subset of targets $S(\mathcal{Y}) \subset \mathcal{Y}$:

$$\tilde{P}(y|x) = \frac{\exp(\beta s_{x,y})}{\sum_{y' \in S(\mathcal{Y})} \exp(\beta s_{x,y'})}. \tag{3}$$

By truncating the softmax, the computational cost of the normalization term is significantly reduced, thus making it feasible to efficiently optimize the dual-encoder parameters. Although this approximation introduces a bias—since only a subset of the full candidate set is considered—it enables scalable training while maintaining the model's ability to learn meaningful semantic representations.

**Top-K Sampling Approximations** Efficient selection of the subset $S(\mathcal{Y})$ is critical for the effectiveness of truncated softmax. Traditional methods, such as BM25, select the top-$k$ candidates based on lexical similarity. For dense retrieval, the top-$k$ targets are selected based on the embedding similarity, which is measured by the dot product between the query embedding and the target embedding. More advanced techniques, such as Gumbel-Max sampling (Lindgren et al. 2021) and large-scale sampling algorithms (Xiong et al. 2020), offer more efficient top-$k$ selection strategies, improving training efficiency and retrieval performance on large datasets.

## Method

In this section, we propose a method for training a dual-encoder model in dense retrieval. The main workflow is shown in Figure 1. This training method mainly consists of two primary objectives: enhancing ranking quality on quantized index by leveraging top-$k$ sampling based on similarity during training, and optimizing the update of target embeddings through a gradient-based caching mechanism.

**Target embedding Initialization** To address the computational challenges in the large-scale training, we first use the pre-trained target encoder $g(y; \Theta)$ to initialize the target embeddings. For each target $y_i$, the corresponding embedding is computed as follows:

$$b_{y_i} = g(y_i; \Theta), \tag{4}$$

where $b_{y_i} \in \mathcal{R}^D$ represents the $D$-dimensional vector of target $y_i$. These initial embeddings are stored in a buffer, $B \in \mathcal{R}^{\mathcal{Y} \times D}$, which contains the cached embeddings for all targets in the dataset.

The interaction between the query embedding and any cached target embedding is computed thought the dot product:

$$s_{x,b_y} = \langle f(x; \Theta), b_y \rangle, \tag{5}$$

where $b_y$ represents the cached embedding of a target stored in the buffer.

Using these scores, the truncated softmax distribution is expressed as:

$$\tilde{P}(b_y|x) = \frac{\exp(\beta s_{x,b_y})}{\sum_{b_{y'} \in S(B)} \exp(\beta s_{x,b_{y'}})}, \tag{6}$$

where $S(B)$ denotes the subset of cached target embeddings selected from the buffer.

By directly sampling a subset of the target embeddings from the cached buffer, we can avoid recalculating target embeddings using the target encoder. This approach drastically reduces memory usage and computational load of the GPU, allowing us to efficiently sample a large number of highly similar negative samples. Therefore, this method accelerates the training process without weakening the model's ability to learn high-quality semantic embeddings.

**Negative Sample Construction** One of the critical points of this method is how to construct negative samples, which are essential for the effective training of the dual-encoder model. In large-scale retrieval tasks, the goal is to select negative samples that are highly similar to the query embedding using the quantized index, as these "hard" negative samples are crucial for optimizing the training objective. This enables the model to learn from more challenging examples that are difficult to distinguish from the true positives, thereby enhancing its discriminative capability.

For each query, we use a hybrid strategy composed of the following two components to construct the negative sample set $S_{\text{neg}}(B)$:

- **Top-$k$ hard negatives from the query:** We exclude the corresponding positive target embedding for the query, and then select the top-k embeddings that are most similar to the target embeddings from the buffer.
- **Top-$k$ hard negatives from other queries in the batch:** For each of the remaining queries in the batch, we select their own top-$k$ most similar target embeddings from the buffer (while excluding their respective positives).

Thus, for each query, the negative sample set $S_{\text{neg}}(B) \in \mathbb{R}^{\mathcal{B} \times k \times D}$ comprises $k \times \mathcal{B}$ negative samples, where $\mathcal{B}$ denotes the batch size. For example, with a batch size $\mathcal{B} = 32$ and $k = 500$, the total number of negative samples amounts to 16,000. This scale is significantly larger than those typically adopted in previous work, making training more robust training with a richer set of negative samples.

**Cached Target Embedding Update** To effectively update the cached target embeddings so that its distribution better aligns with the retrieval objective, while maintaining computational efficiency, we adopt a gradient-based update mechanism. This mechanism operates on a subset of the buffer in each training batch. The overall process is illustrated in Figure 1.

For each training batch, in order to obtain the corresponding query embedding, we first use the pre-trained target encoder to encode the input query. Then, we retrieve the top-k most similar target embeddings from the buffer $B$ based on vector similarity to construct the negative sample set $S_{\text{neg}}(B)$. This negative sample set is combined with the positive target embedding corresponding to each query to form the final subset of cached target embeddings $S(B) \in \mathbb{R}^{\mathcal{B}\times(k+1)\times D}$, which is used for a single training step.

Given the batch of query embeddings and the corresponding subset of cached target embeddings $S(B)$, we compute the similarity scores between each query embedding and each target embeddings in $S(B)$. These scores are then transformed into a probability distribution using the truncated softmax defined in Equation (6).

The loss function is defined as the negative log-probability of the truncated softmax distribution:

$$\mathcal{L} = -\sum_{x\in\mathcal{B}} \log \tilde{P}(b_y|x), \tag{7}$$

Based on the gradient of the loss $\mathcal{L}$, both the query encoder and the subset of cached target embeddings $S(B)$ are updated. Each cached target embedding $b_y \in S(B)$ is updated as $b_y \leftarrow b_y - \eta\frac{\partial\mathcal{L}}{\partial b_y}$, where $\eta$ denotes the learning rate. The updated subset $S(B)$ then replaces the corresponding entries in the original buffer $B$, and is used in subsequent training iterations.

This method can efficiently and continuously optimize the cached target embeddings in a retrieval-related manner, without repeatedly invoking the target encoder. As a result, it significantly reduces both GPU memory consumption and computational overhead.

In order to achieve direct gradient-based optimization of the cached target embeddings $B$, we implement the following update procedure in each training step:

Algorithm 1: Gradient-based Update of Cached Target Embeddings

1: Initialize a learnable tensor $W \in \mathbb{R}^{\mathcal{B}\times(k+1)\times D}$ within the neural network model.
2: For each training batch, a subset $S(B) \subset B$ is sampled from the cached target embeddings.
3: The sampled buffer values is assigned to the tensor: $W \leftarrow S(B)$.
4: $W$ is updated based on gradient through the backpropagation of the loss $\mathcal{L}$.
5: The updated values are written back to the subset of cached target embeddings $S(B) \leftarrow W$.
6: The corresponding entries in the cached buffer $B$ are replaced with the updated embeddings $S(B)$.

**Faiss Index Construction and Update** To efficiently retrieve the top-$k$ most similar samples during similarity-based sampling, we leverage the Faiss library to construct an index that supports fast nearest neighbor search. The most basic index is the flat index, which stores full vectors and performs exhaustive search. This index achieves the highest accuracy and is often used in passage retrieval tasks to report the optimal results. However, it has high memory consumption and low computational efficiency, and is thus inefficient for large-scale datasets.

To address these limitations, we adopt a quantized Hierarchical Navigable Small World (HNSW) index, which was constructed by combining HNSW with an Inverted File with Product Quantization (IVFPQ) index. The IVFPQ significantly compresses the index size while maintaining high retrieval speed, making it highly suitable for large-scale training and retrieval scenarios. However, this compression comes at the cost of reduced retrieval accuracy.

During the training, the cached target embeddings $B$ is continuously updated, so the Faiss index must also be refreshed to reflect the latest representation. Rebuilding the index after each training step would incurs considerable computational overhead. To strike a balance between retrieval accuracy and training efficiency, we adopt a periodic index update strategy. Specifically, the Faiss index is rebuilt every $C$ training steps. In our experiments, we set the update interval to $C = 800$ steps, which achieves a favorable trade-off between up-to-date representations and computational cost.

**Training Process** The overall training process, including query encoding, retrieval of negative samples, loss calculation, and periodic updates to the Faiss index, is all outlined in Algorithm 2.

Algorithm 2: Improving Dense Retrieval Accuracy on quantized index via Gradient-Optimized Target Embeddings

**Require:** Pre-trained query encoder $f$, target encoder $g$; target dataset $\mathcal{Y}$; hyperparameters: learning rate $\eta$, batch size $\mathcal{B}$, number of top-$k$ similar negative samples per query, index update interval $\mathcal{C}$, temperature $\tau$
**Ensure:** Optimized query encoder $f$ and updated cached target embeddings $B$
1: Compute initial target embeddings: $B = \{g(y_i) \mid \forall y_i \in \mathcal{Y}\}$
2: Construct Faiss index for $B$
3: **for** each training step **do**
4: Sample a batch of queries $\mathcal{B} = \{x_j\}_{j=1}^{\text{batch}}$
5: Compute query embeddings: $e_{x_j} = f(x_j)$
6: Retrieve top-$k$ hard negative samples from $B$ for each query
7: Compute loss $\mathcal{L}$
8: Update query encoder parameters
9: Update $B$ with modified target embeddings according to Algorithm 1
10: **if** training step $\mod \mathcal{C} == 0$ **then**
11: Rebuild Faiss index with updated $B$
12: **end if**
13: **end for**

## Experiments

In this section, we will evaluate the effectiveness and efficiency of our proposed method in large-scale retrieval tasks on three widely used benchmark datasets: the **Natural Questions** (Kwiatkowski et al. 2019), **TriviaQA** (Joshi et al. 2017), and **MSMARCO Passage Ranking** (Nguyen et al. 2016). Our primary goals are to demonstrate the following key improvements:

- Optimizing the training of the query encoder through large negative sampling based on similarity, where negatives are dynamically retrieved using the quantized index.
- Achieving scalable and efficient updates to cached target embeddings through direct gradient-based refinement.

The Natural Questions consists of the actual queries issued by users to Google Search. These queries are annotated by human experts, and the labeled content consists of answer fragments from relevant Wikipedia passages. TriviaQA contains questions presented in the form of trivia and their corresponding answers, which are collected from a variety of web sources. Both Natural Questions and TriviaQA use a shared corpus of over 21 million Wikipedia passages as the target retrieval pool.

The MSMARCO Passage Ranking was developed based on the MSMARCO Question Answering dataset and was specifically designed for passage retrieval. It includes approximately 8.8 million passages and is constructed through real Bing search queries. The aim is to retrieve passages that contain the correct answers to the questions.

For the evaluation, we primarily use the **Recall@k (R@k)** metric, which measures the proportion of queries that can retrieve at least one relevant passage within the top $k$ candidates. In addition, for MSMARCO, we report **MRR@10** (Mean Reciprocal Rank at 10), which reflects the quality of the ranking as it takes into account the position of the first relevant passage among the top 10 results.

The pre-trained target encoder is used to embed all target texts in advance, generating an initial embedding buffer $B \in \mathbb{R}^{\mathcal{Y} \times D}$, where $\mathcal{Y} = 21$M for Wikipedia passages (used in NQ and TriviaQA), and $\mathcal{Y} = 8.8$M for MSMARCO. The embedding dimension $D$ is set to 768. In our experiments, to efficiently initialize this buffer, we directly reused the precomputed dense embeddings provided by previous work. Specifically, we loaded the publicly released FAISS FlatIP indexes from DPR and ANCE, which store full-precision passage embeddings.

We trained the models for 10 epochs on NQ and TriviaQA, and for 1 epoch on MSMARCO due to its significantly larger number of training set. The learning rate used by the query encoder was $2 \times 10^{-6}$, while the selected subset of cached target embeddings $S(B)$ was updated with a smaller learning rate of $1 \times 10^{-6}$ to ensure the stable embedding distribution. We used a batch size of $\mathcal{B} = 32$, a similarity-based sampling of top-$k = 500$ and a temperature $\tau = 8$. The Faiss index used for efficient retrieval was refreshed every $\mathcal{C} = 800$ training steps.

All the experiments were conducted on a single workstation, which was equipped with two NVIDIA RTX 3090 GPUs (each with 24GB of memory), an Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50GHz, and 125GB RAM.

### Experimental Results

We assessed the effectiveness of our proposed method, considering it as an advanced improvement over the dual-encoder retrieval architectures, with a particular focus on improving retrieval accuracy under the condition of quantized index. To validate its performance, we applied the gradient-based embedding update method to two representative dual-encoder models: DPR (Karpukhin et al. 2020), which is a widely adopted dual-encoder retrieval system, and ANCE (Xiong et al. 2020), which extends DPR by asynchronously updating the FAISS index and using outdated target embeddings to mine hard negatives. In our experiments, we initialize from the pretrained checkpoints of these models and continue training using our proposed gradient-based embedding update method.

For Natural Questions, we start with publicly released DPR checkpoints trained on the latest versions of the datasets, and then further fine-tune them on the original training data to better evaluate the improvement effect brought by our approach. For the MSMARCO Passage Ranking task and TriviaQA, we continue to train from a publicly available ANCE-trained model and report the results under our method. It was also compared with Corrector Networks (Monath et al. 2024), which address the problem of stale embeddings by learning a lightweight parametric network to adjust the cached target vectors, thereby achieving more accurate softmax approximations and efficient hard negative sampling to solve the problem of outdated embeddings.

The index size and retrieval speed of different indexes constructed based on the same retrieval corpus were compared, as shown in Table 1. The Quantized HNSW (QHNSW) index is more than 25 times smaller than the FlatIP index, and the retrieval speed per batch is more than 300 times faster than the latter. Although the FlatIP index achieves the highest retrieval accuracy due to exhaustive search, its large memory footprint and low query efficiency make it unsuitable for actual large-scale applications.

This observation result indicates that QHNSW performs exceptionally well in terms of memory consumption and retrieval latency, making it highly suitable for scenarios requiring high-speed response under limited memory. However, this efficiency is typically achieved at the expense of reduced retrieval accuracy, as it employs approximate search and vector quantization techniques.

To address this limitation, our method aims to significantly enhance the retrieval accuracy of the QHNSW index. As shown in Table 2 and Table 3, our method has improved performance on multiple datasets and baseline models.

For the MS MARCO Dev and the TriviaQA Test, we first trained models using ANCE, and then continued to train it using our proposed method. As seen in Table 2, although our method did not further improve the performance of the FlatIP index—in fact, it even led to a slight decrease in accuracy —it brought a substantial improvement in the QHNSW index. Specifically, our method improved the MRR@10 of

| Dataset | Passages | Index | Size (GB) | Speed (ms) |
|---|---|---|---|---|
| Wiki | 21M | FlatIP | 61 | 16264 |
| | | QHNSW | **2.7** | **46** |
| MSMARCO | 8.8M | FlatIP | 26 | 15559 |
| | | QHNSW | **1.2** | **40** |

Table 1. Comparison of index size and batch query latency under different datasets and index types. Speed (ms) denotes the average retrieval latency per batch.

| Index | Method | MS-MARCO Dev | | TriviaQA Test | | |
|---|---|---|---|---|---|---|
| | | MRR@10 | R@1000 | R@1 | R@5 | R@10 |
| FlatIP | DPR | - | - | 44.85 | 64.32 | 70.05 |
| | ANCE | **32.26** | **95.82** | **56.48** | **72.26** | **76.65** |
| | Ours | 29.71 | 94.97 | 55.20 | 71.10 | 75.45 |
| QHNSW | DPR | - | - | 38.68 | 60.12 | 66.69 |
| | ANCE | 12.74 | 74.18 | 51.22 | 69.59 | **74.80** |
| | Ours | **26.32** | **89.30** | **53.91** | **69.81** | 74.39 |

Table 2. Comparison of retrieval performance using flat and quantized index on MS MARCO Dev and TriviaQA Test. We report the MRR@10 and R@1000 of MS MARCO, as well as recall rates of TriviaQA at different top-*k* thresholds.

the MS MARCO Dev by 14 percentage points, and increased the R@1 of the TriviaQA Test by 2.5 percentage points. This demonstrates that it is very effective in improving the quality of the quantitative index.

For the Natural Questions Test dataset, whose results are presented in Table 3, we continued the training from the publicly released DPR model. Our method has improved the retrieval accuracy on both FlatIP and QHNSW indexes, achieving consistent gains across all evaluation metrics (R@1, R@5, and R@10) on the FlatIP index, and significantly boosting R@1 and R@5 on QHNSW. Notably, our method improves R@1 on the QHNSW index by 5 percentage points over the baseline DPR model, narrowing the gap with the FlatIP index to just one percentage point—an impressive result considering the substantial difference in index size and speed.

Taken together, the results on MS MARCO Dev, TriviaQA Test, and Natural Questions Test demonstrate that our method substantially improves the retrieval accuracy of the QHNSW index, particularly in terms of the MRR@10 and R@1 metrics. These improvements are critical for downstream question answering tasks, as these tasks depend heavily on the quality of top-ranked retrieval results.

Our approach combines large-scale hard negative sampling based on similarity with efficient updates of cached target embeddings based on gradients. This design not only improves the accuracy of retrieval but also accelerates the convergence speed. Specifically, for Natural Questions and TriviaQA, we can complete training in just 10 epochs, and for MS MARCO, it only takes only 1 epoch , thereby substantially reducing the overall training time and computational cost.

In summary, our method effectively bridges the gap between accuracy and efficiency in retrieval systems based on quantized index, offering a practical and scalable solution for real-world, large-scale retrieval tasks.

## Effect of Temperature on Training Performance

In the truncated softmax distribution (Equation 6), the temperature parameter $\tau$ plays a crucial role in modulating the sharpness of the output probability distribution:

$$\tilde{P}(b_y|x) = \frac{\exp(\beta s_{x,b_y})}{\sum_{b_{y'} \in S(B)} \exp(\beta s_{x,b_{y'}})},$$

A larger value of $\tau$ leads to a more peaked distribution, assigning higher probabilities to the most similar candidates. In contrast, a smaller $\tau$ results in a flatter distribution, promoting a more uniform allocation of attention across all candidates.

To investigate its effect, we conducted experiments on the Natural Questions (NQ) training set. By gradually adjusting the $\tau$ values within the range from 0.5 to 16, and evaluating the R@1 retrieval accuracy on the NQ test set, all the retrieval evaluations were carried out using the QHNSW index.

As shown in Figure 2, the best retrieval accuracy is achieved at $\tau = 8$, with the R@1 score of approximately 51.52%. This indicates that a moderately softened distribution helps the model better distinguish positive samples from hard negatives during training, without making the distribution overly concentrated and focusing only on a few candidate samples with high scores.

Based on these observations, we set $\tau = 8$ as the default value for all subsequent experiments, as under the quantized index setting, it consistently yields the best retrieval performance.

## Ablation Study on Model Training

To better understand the specific roles of each component in our proposed method, we conducted an ablation study using the quantized index on the NQ dataset. Specifically, we evaluated the impact of large negative sampling based on

| Index | Method | R@1 | R@5 | R@10 | R@20 | R@100 |
|---|---|---|---|---|---|---|
| FlatIP | DPR | 52.13 | 71.80 | 77.20 | 81.16 | 87.06 |
| | ANCE | 51.19 | 77.78 | 77.78 | 81.9 | 87.5 |
| | Corrector | 50.61 | 71.00 | 77.73 | **82.66** | **88.39** |
| | Ours | **52.49** | **71.85** | **77.84** | 81.91 | 87.40 |
| QHNSW | DPR | 46.06 | 67.42 | 73.76 | 77.83 | 84.65 |
| | ANCE | 44.96 | 68.48 | 75.24 | **79.89** | 86.09 |
| | Ours | **51.52** | **70.55** | **76.01** | 79.81 | **86.15** |

Table 3. Comparison of retrieval performance using flat and quantized index on Natural Questions Test. We report the recall rates at different top-*k* thresholds: R@1, R@5, R@10, R@20, and R@100.

| Configuration | R@1 | R@5 | R@10 | R@20 | R@100 |
|---|---|---|---|---|---|
| Baseline | 46.06 | 67.42 | 73.76 | 77.83 | 84.65 |
| Large-Batch Negatives | 49.72 | 68.28 | 74.01 | 78.31 | 84.12 |
| Full Model | **51.52** | **70.55** | **76.01** | **79.81** | **86.15** |

Table 4. Ablation study results on the NQ dataset. The retrieval performance is evaluated using the QHNSW index.

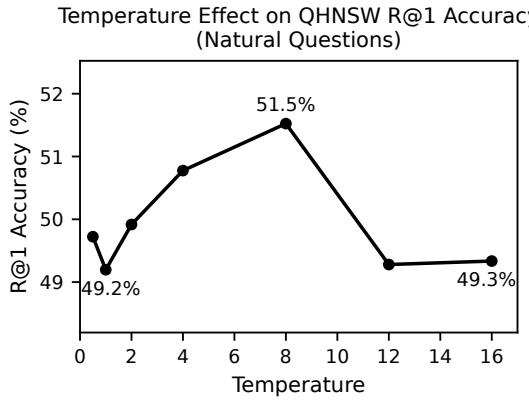


Figure 2: Retrieval performance under different temperature values on the Natural Questions testing set.

similarity and gradient-based updates to the cached target embeddings.

First, we adopted the baseline configuration, which used the original dual-encoder model without any further training or enhancements. The following configurations were then tested:

- **Large-Batch Negative samples**: Enhanced query encoder training by incorporating similarity-based large negative samples retrieved dynamically from the quantized index.
- **Full Method**: Combined both quantized-index-based negative sampling and gradient-optimized target embedding updates, representing our complete proposed method.

The ablation study results, shown in Table 4, demonstrate the effectiveness of the proposed components. The large negative sampling based on similarity has improved retrieval performance across all Recall@k metrics, particularly enhancing the ability of the query embedding to distinguish between relevant and irrelevant passages. The update of the cached target embeddings based on gradients further boost the performance, especially at higher recall rates, by ensuring that the embeddings remain accurate and aligned with the evolving query representations. Combining these two strategies in the Full Method led to the best overall performance, outperforming both the Baseline and Large-Batch Negatives configurations. This highlights the complementary benefits of these components in improving retrieval accuracy and efficiency.

Overall, the ablation study confirms that large negative sampling based on similarity can enhance the discriminative ability of the query encoder, while gradient-based updates ensure that the cached embeddings remain consistent with the constantly query representations. These strategies significantly improve retrieval performance, making the model more efficient for large-scale retrieval tasks.

## Conclusion

In this paper, we propose a scalable and efficient training method that significantly improves the retrieval accuracy of dense dual-encoder models on quantized index. By directly updating cached target embeddings via gradient backpropagation, our method reduces memory usage and eliminates the need for repeated target encoding. To better approximate the full softmax distribution, we introduce a similarity-guided negative sampling strategy that dynamically retrieves top-k candidates from the quantized index across and within batches. Experimental results on Natural Questions, TriviaQA, and MSMARCO show that our approach significantly improves QHNSW retrieval accuracy, particularly in MRR@10 and R@1. These gains lead to more effective and efficient retrieval, benefiting downstream open-domain QA tasks that rely on high-quality top-ranked candidates.

## References


Arabzadeh, N.; Yan, X.; and Clarke, C. L. 2021. Predicting efficiency/effectiveness trade-offs for dense vs. sparse retrieval strategy selection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2862–2866.

Bornea, A.-L.; Ayed, F.; De Domenico, A.; Piovesan, N.; and Maatouk, A. 2024. Telco-RAG: Navigating the Challenges of Retrieval Augmented Language Models for Telecommunications. In *GLOBECOM 2024-2024 IEEE Global Communications Conference*, 2359–2364. IEEE.

Danopoulos, D.; Kachris, C.; and Soudris, D. 2019. Approximate similarity search with faiss framework using fpgas on the cloud. In *International Conference on Embedded Computer Systems*, 373–386. Springer.

Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.-E.; Lomeli, M.; Hosseini, L.; and Jégou, H. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Gao, L.; and Callan, J. 2021. Unsupervised Corpus Aware Language Model Pre-training for Dense Passage Retrieval. arXiv:2108.05540.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Karpukhin, V.; Oğuz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Kukreja, S.; Kumar, T.; Bharate, V.; Purohit, A.; Dasgupta, A.; and Guha, D. 2023. Vector Databases and Vector Embeddings-Review. In *2023 International Workshop on Artificial Intelligence and Image Processing (IWAIIP)*, 231–236. IEEE.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Lin, P.-C.; and Zhao, W.-L. 2019. A comparative study on hierarchical navigable small world graphs. *Computing Research Repository (CoRR) abs/1904.02077*.

Lindgren, E.; Reddi, S.; Guo, R.; and Kumar, S. 2021. Efficient training of retrieval models using negative cache. *Advances in Neural Information Processing Systems*, 34: 4134–4146.

Monath, N.; Grathwohl, W.; Boratko, M.; Fergus, R.; McCallum, A.; and Zaheer, M. 2024. A fresh take on stale embeddings: improving dense retriever training with corrector networks. *arXiv preprint arXiv:2409.01890*.

Monath, N.; Zaheer, M.; Allen, K.; and McCallum, A. 2023. Improving dual-encoder training through dynamic indexes for negative mining. In *International Conference on Artificial Intelligence and Statistics*, 9308–9330. PMLR.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. Ms marco: A human-generated machine reading comprehension dataset. *https://openreview.net/forum?id=Hk1iOLcle*.

Norouzi, M.; Punjani, A.; and Fleet, D. J. 2013. Fast exact search in hamming space with multi-index hashing. *IEEE transactions on pattern analysis and machine intelligence*, 36(6): 1107–1119.

Qu, Y.; Ding, Y.; Liu, J.; Liu, K.; Ren, R.; Zhao, W. X.; Dong, D.; Wu, H.; and Wang, H. 2021. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. arXiv:2010.08191.

Rawat, A. S.; Chen, J.; Yu, F. X. X.; Suresh, A. T.; and Kumar, S. 2019. Sampled softmax with random fourier features. *Advances in Neural Information Processing Systems*, 32.

Reddi, S. J.; Kale, S.; Yu, F.; Holtmann-Rice, D.; Chen, J.; and Kumar, S. 2019. Stochastic negative mining for learning with large output spaces. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 1940–1949. PMLR.

Robertson, S.; Zaragoza, H.; et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4): 333–389.

Sachan, D. S.; Lewis, M.; Joshi, M.; Aghajanyan, A.; Yih, W.-t.; Pineau, J.; and Zettlemoyer, L. 2022. Improving passage retrieval with zero-shot question generation. *arXiv preprint arXiv:2204.07496*.

Thakur, N.; Reimers, N.; and Lin, J. 2022. Domain adaptation for memory-efficient dense retrieval. *arXiv preprint arXiv:2205.11498*.

Xiong, L.; Xiong, C.; Li, Y.; Tang, K.-F.; Liu, J.; Bennett, P.; Ahmed, J.; and Overwijk, A. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.

Xu, S.; Pang, L.; Xu, J.; Shen, H.; and Cheng, X. 2024. List-aware reranking-truncation joint model for search and retrieval-augmented generation. In *Proceedings of the ACM on Web Conference 2024*, 1330–1340.

Yang, Z.; Shao, Z.; Dong, Y.; and Tang, J. 2024. TriSampler: A Better Negative Sampling Principle for Dense Retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 9269–9277.

Zhao, P.; Zhang, H.; Yu, Q.; Wang, Z.; Geng, Y.; Fu, F.; Yang, L.; Zhang, W.; Jiang, J.; and Cui, B. 2024a. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473*.

Zhao, W. X.; Liu, J.; Ren, R.; and Wen, J.-R. 2024b. Dense text retrieval based on pretrained language models: A survey. *ACM Transactions on Information Systems*, 42(4): 1–60.