# Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling

Sebastian Hofstätter[1], Sheng-Chieh Lin[2], Jheng-Hong Yang[2], Jimmy Lin[2], Allan Hanbury[1]
[1] TU Wien, [2] University of Waterloo

## ABSTRACT

A vital step towards the widespread adoption of neural retrieval models is their resource efficiency throughout the training, indexing and query workflows. The neural IR community made great advancements in training effective dual-encoder dense retrieval (DR) models recently. A dense text retrieval model uses a single vector representation per query and passage to score a match, which enables low-latency first-stage retrieval with a nearest neighbor search. Increasingly common, training approaches require enormous compute power, as they either conduct negative passage sampling out of a continuously updating refreshing index or require very large batch sizes. Instead of relying on more compute capability, we introduce an efficient topic-aware query and balanced margin sampling technique, called TAS-Balanced. We cluster queries once before training and sample queries out of a cluster per batch. We train our lightweight 6-layer DR model with a novel dual-teacher supervision that combines pairwise and in-batch negative teachers. Our method is trainable on a single consumer-grade GPU in under 48 hours. We show that our TAS-Balanced training method achieves state-of-the-art low-latency (64ms per query) results on two TREC Deep Learning Track query sets. Evaluated on NDCG@10, we outperform BM25 by 44%, a plainly trained DR by 19%, docT5query by 11%, and the previous best DR model by 5%. Additionally, TAS-Balanced produces the first dense retriever that outperforms every other method on recall at any cutoff on TREC-DL and allows more resource intensive re-ranking models to operate on fewer passages to improve results further.

## CCS CONCEPTS

• **Information systems** → **Learning to rank**;

## KEYWORDS

Dense Retrieval; Knowledge Distillation; Batch Sampling

**ACM Reference Format:**
Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, Allan Hanbury. 2021. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462891


*SIGIR '21, July 11–15, 2021, Virtual Event, Canada*

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462891

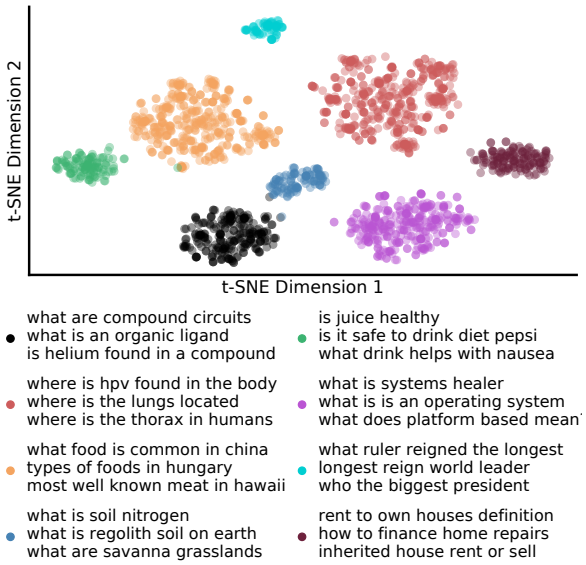



**Figure 1: T-SNE plot of 8 randomly sampled topic clusters and example queries. Our Topic Aware Sampling (TAS) composes queries from a single cluster per batch.**

## 1 INTRODUCTION

Having a well prepared teacher in life makes learning easier and more efficient. Training dense text retrieval models with more experienced and capable teacher models follows the same path. Dense retrieval models – such as the BERT-based [10] dual-encoder $\text{BERT}_{\text{DOT}}$ – offer the great potential of low-latency query times, vastly better accuracy and recall than traditional first-stage retrieval methods, and moving most computational cost into offline indexing and training. The unifying $\text{BERT}_{\text{DOT}}$ architecture is already supported by many open source search engines. $\text{BERT}_{\text{DOT}}$ can be used as a standalone retriever or as part of a re-ranking pipeline. The problem, when further improving the result quality, becomes the affordability in terms of hardware resources and requirements for training and indexing. A recent trend to improve retrieval result quality is to augment the $\text{BERT}_{\text{DOT}}$ training procedure, which leads to increased hardware requirements. Examples of this include conducting negative passage sampling out of a continuously updating refreshing index (ANCE [42]), generations of models (LTRe [44]), or requiring large batch sizes (RocketQA [11]).

A concurrent line of inquiry is the use of knowledge distillation from more effective, but less efficient architectures as teachers either in pairwise [14, 16, 25] or in-batch negatives [24] settings. In-batch negatives reuse the encoded representations per sample and compute interactions between all samples in a batch. We combine these two knowledge distillation paradigms into a novel dual-supervision

with a pairwise concatenated $\text{BERT}_{\text{CAT}}$ and a ColBERT teacher for in-batch negatives. These approaches, while already working well, are constrained by the information gain a single random batch can deliver for training. The training data available to dense retrieval training consists of a pool of queries, and associated with each query is typically a set of passage pairs with a teacher score margin. Each pair consists of a relevant and non-relevant passage, with the margin set by subtracting the non-relevant sampled passage teacher score from the relevant passage teacher score.

The main contribution of this work is to improve both pairwise and in-batch teacher signals. We propose Balanced Topic Aware Sampling (TAS-Balanced) to compose dense retrieval training batches. This sampling technique has two components: (1) we compose batches based on queries clustered in topics (TAS); and (2) we then select passage pairs so as to balance pairwise teacher score margins (TAS-Balanced). We cluster the topics once before training based on a baseline representation by semantic dot product similarity (which allows grouping queries without lexical overlap) – a one time cost of under 10 minutes for all 400K training queries of MSMARCO. An example selection of topic clusters is shown in Figure 1. Previously, a batch would be composed of random queries from the training set, leaving little information gain for in-batch negatives. By selecting queries from a single cluster, we concentrate information about a topic in a single batch, which after in-batch negative teaching, leads to higher quality retrieval results.

We show that with TAS-Balanced batches and dual-supervision we can train a very effective dense retrieval model on a single consumer-grade (11GB memory) GPU in under 48 hours, as opposed to a common configuration of 8x V100s, because our method does not rely on repeated indexing [42] or large batch size training [11]. Specifically, we study the following research questions:

**RQ1** How effective are TAS and TAS-Balanced batch sampling techniques with single and dual-teacher supervision?

We find TAS improving both in-batch negative teaching alone as well as our dual-supervision teachers. The TAS-Balanced sampling improves pairwise training, in-batch negatives, and the dual-supervision training, which represents the best overall configuration across our three query sets. The dual-teacher supervision has an especially big positive impact on recall using a Margin-MSE loss. We study different losses for the dual-supervision and find Margin-MSE improves the results consistently over other loss functions.

A common problem in machine learning research is inadvertent overfitting on a specific combination of hyperparameters, random seed, and collection. To gain confidence in our results, we study:

**RQ2** How robust is TAS-Balanced to different randomization?

To show that TAS-Balanced is robust against random overfitting, we conduct a randomization study of 5 instances with different random orderings of selected clusters and queries. We find only small standard deviations across the metrics of our query sets ($< .01$ nDCG change on TREC-DL; $< .001$ MRR on MSMARCO-DEV). This gives us great confidence in the efficacy and robustness of our technique. To set our results in context to related work, we answer:

**RQ3** How does our TAS-Balanced approach compare to other dense retrieval training methods?

We evaluate our models on two TREC-DL ('19 & '20) query sets and the MSMARCO-DEV set using the MSMARCO passage collection. The two TREC sets are especially suited to study recall quality of the dense retrievers with hundreds of judged passages per query. Our TAS-Balanced & dual-supervision trained $\text{BERT}_{\text{DOT}}$ model shows state-of-the-art low-latency results on both TREC-DL'19 and '20 query sets using a batch size as small as 32. Our $\text{BERT}_{\text{DOT}}$ model, evaluated on NDCG@10, outperforms BM25 by 44%, a plainly trained DR by 19%, docT5query by 11%, and the previous best DR model by 5%. On the sparse labelled MSMARCO-DEV queries, TAS-Balanced shows the best results for methods using a single consumer-grade GPU and outperforms most approaches that require 20x more resources to train. Finally, while TAS-Balanced is an effective standalone low-latency retriever, we also study the impact of our TAS-trained model in a larger search system:

**RQ4** How well suited is our TAS-trained dense retriever as a first-stage module in terms of recall and re-ranking gains?

We find that TAS-Balanced results in the first $\text{BERT}_{\text{DOT}}$ model that outperforms BM25 and docT5query consistently on every recall cutoff in TREC-DL densely judged query sets. Fused together with docT5query results we see another increase in recall, showing that dense and sparse solutions still benefit from each other at an already high recall level. Furthermore, we stack the state-of-the-art re-ranking system mono-duo-T5 on top of our first-stage retrieval. Because TAS-trained $\text{BERT}_{\text{DOT}}$ increases the recall and accuracy for small cutoffs, we can reduce the number of passages an expensive re-ranking system processes and still receive considerable benefits. However, we also find a limitation in re-ranking quality for higher cutoffs: Even though TAS-Balanced continues to improve the recall at higher cutoffs, the re-ranking does not take advantage of that. Future work may improve re-rankers on top of TAS-Balanced.

The aim of this work is to produce a very effective $\text{BERT}_{\text{DOT}}$ retrieval model and minimize the training resources necessary. Our contributions are as follows:

- We propose an efficient Topic Aware Sampling (TAS-Balanced) for composing informative dense retrieval training batches
- We show that TAS-Balanced in combination with a dual-teacher supervision achieves state-of-the-art DR results on TREC-DL
- We study our training robustness and how TAS-Balanced improves a larger (re-)ranking system
- We publish our source code at: https://github.com/sebastian-hofstaetter/tas-balanced-dense-retrieval

## 2 RETRIEVAL MODEL BACKGROUND

We employ three different Transformer based [38] & BERT pre-trained [10] architectures in our work. We use two teacher architectures for the best combination of pairwise ($\text{BERT}_{\text{CAT}}$) and in-batch negative teaching (ColBERT) to train our our main dense retrieval model: the dual-encoder $\text{BERT}_{\text{DOT}}$ architecture. In the following we present the characteristics of each model architecture, our dual-teacher supervision, as well as related training methods.

### 2.1 BERT Teacher Models

The common way of utilizing the BERT pre-trained Transformer model in a re-ranking scenario is by concatenating query and passage input sequences [1, 28, 30]. We refer to the architecture as

$\text{BERT}_{\text{CAT}}$. We use it in this work as our strong pairwise teacher model. In the $\text{BERT}_{\text{CAT}}$ ranking model, the query $q_{1:m}$ and passage $p_{1:n}$ sequences are concatenated with special tokens (using the ; operator), encoded with BERT, the CLS token representation pooled, and scored with single linear layer $W_s$:

$$\text{BERT}_{\text{CAT}}(q_{1:m}, p_{1:n}) = W_s \, \text{BERT}(\text{CLS}; q_{1:m}; \text{SEP}; p_{1:n})_{\text{CLS}} \quad (1)$$

This architecture is easy to train and provides very strong results in terms of effectiveness, especially when used in an ensemble [14]. However, it requires candidate selection prior to re-ranking, has no ability to pre-compute and index passage representations, and is therefore slow in practice [15].

The ColBERT model [22] tries to overcome the time-efficiency problem of $\text{BERT}_{\text{CAT}}$ by delaying the interactions between every query and document term representation after BERT encoding:

$$\begin{aligned} \hat{q}_{1:m} &= \text{BERT}(\text{CLS}; q_{1:m}; \text{SEP}) \\ \hat{p}_{1:n} &= \text{BERT}(\text{CLS}; p_{1:n}; \text{SEP}) \end{aligned} \quad (2)$$

The interactions in the ColBERT model are aggregated with a max-pooling per query term and sum of query-term scores as follows:

$$\text{ColBERT}(q_{1:m}, p_{1:n}) = \sum_{1}^{m} \max_{1..n} \hat{q}_{1:m}^{T} \cdot \hat{p}_{1:n} \quad (3)$$

This decoupling of query and passage encoding allows the passage representations to be indexed in theory. However, the storage cost of pre-computing passage representations is much higher and scales in the total number of terms in the collection. Because of the storage increase and increased complexity for the scoring aggregation we refrain from using ColBERT as a dense retrieval model, and rather use it as an efficient teacher for in-batch negatives.

## 2.2 $\text{BERT}_{\text{DOT}}$ Dense Retrieval Model

The $\text{BERT}_{\text{DOT}}$ model encodes query $q_{1:m}$ and passage $p_{1:n}$ sequences independently from each other and matches only a single representation vector of the query with a single representation vector of a passage [25, 26, 42]. It pools each CLS token output for query $\hat{q}$ and passage $\hat{p}$ representations as follows:

$$\hat{q} = \text{BERT}(\text{CLS}; q_{1:m}; \text{SEP})_{\text{CLS}}, \; \hat{p} = \text{BERT}(\text{CLS}; p_{1:n}; \text{SEP})_{\text{CLS}} \quad (4)$$

Potentially after storing all representations in an index, the model computes the final scores as the dot product $\cdot$ of $\hat{q}$ and $\hat{p}$:

$$\text{BERT}_{\text{DOT}}(q_{1:m}, p_{1:n}) = \hat{q} \cdot \hat{p} \quad (5)$$

The independence of query and document encoding as well as the dot product scoring enables two crucial operations for this work. First, we encode all queries once and use their representation for clustering in our TAS approach and second we deploy $\text{BERT}_{\text{DOT}}$ with a simple maximum-inner product retrieval workflow: After training, we encode and index every passage once in a nearest neighbor search index and retrieve the top results at query time for a single encoded query.

In Table 1 we give a training-agnostic latency analysis of our $\text{BERT}_{\text{DOT}}$ retrieval setup. We use both the DistilBERT encoder and a brute-force Faiss nearest neighbor index (FlatIP) on a single TITAN RTX GPU with a total of 24 GB memory. We can fit a batch size of up to 2,000 queries on this single GPU. We measure that a single query can be responded to in 64ms, batching up to 10 queries (for example in a high load system) only reduces the latency to 162ms. The practical effect of always computing roughly the same number of operations reduces the volatility of the latency to a very small margin as the 99$^{\text{th}}$ percentile of the measured latency is very close to the mean. For our one-time clustering we utilize the query encoding only with a batch size of 2,000, shown in Table 1. The fast processing allows us to encode all 400K MSMARCO training queries in one minute.

**Table 1: Latency analysis of Top-1000 retrieval using our $\text{BERT}_{\text{DOT}}$ retrieval setup for all MSMARCO passages using DistilBERT and Faiss (FlatIP) on a single Titan RTX GPU**

| Batch Size | Q. Encoding Avg. | Q. Encoding 99$^{\text{th}}$ Per. | Faiss Retrieval Avg. | Faiss Retrieval 99$^{\text{th}}$ Per. | Total Avg. | Total 99$^{\text{th}}$ Per. |
|---|---|---|---|---|---|---|
| **1** | 8 ms | 11 ms | 54 ms | 55 ms | 64 ms | 68 ms |
| **10** | 8 ms | 9 ms | 141 ms | 144 ms | 162 ms | 176 ms |
| **2,000** | 273 ms | 329 ms | 2,515 ms | 2,524 ms | 4,780 ms | 4,877 ms |

## 2.3 Dual-Teacher Supervision

The community produces mounting evidence that knowledge distillation is essential for effective dense retrieval training [11, 14, 24]. Hofstätter et al. [14] showed the benefits of an ensemble of pairwise $\text{BERT}_{\text{CAT}}$ teachers; concurrently, Lin et al. [24] showed the benefits of using a ColBERT teacher model for in-batch negative sampling. Both possess unique strengths: $\text{BERT}_{\text{CAT}}$ is the more effective teacher, but prohibitively expensive to use for in-batch negatives as it requires quadratic scaling in the batch size, because we need to encode concatenated pairs; ColBERT only requires a linear runtime (in the batch size) for in-batch negative scores.

In this work we combine these two approaches into a novel dual-teacher supervision paradigm that provides the best trade-off between effective teaching and efficient training.

We utilize the published $\text{BERT}_{\text{CAT}}$ ensemble scores from Hofstätter et al. [14] for every official training triple of the MSMARCO-Passage collection. Using this data allows us to use these teacher model $M_t$ scores as a teacher signal for our $M_s$ student model ($\text{BERT}_{\text{DOT}}$) without computational cost. Any pairwise loss function is applicable here, we use the very effective Margin-MSE loss [14], formalized as follows:

$$\begin{aligned} \mathcal{L}_{Pair}(Q, P^+, P^-) = \text{MSE}(&M_s(Q, P^+) - M_s(Q, P^-), \\ &M_t(Q, P^+) - M_t(Q, P^-)) \end{aligned} \quad (6)$$

For the in-batch negative signal, we use the fact that both $\text{BERT}_{\text{DOT}}$ student and ColBERT teacher can independently compute the representation vectors, that is then scored via a dot-product. To create in-batch pairings we cross the representation and pair each positive passage with all other passages in the batch and compute the loss:

$$\begin{aligned} \mathcal{L}_{InB}(Q, P^+, P^-) = \frac{1}{2|Q|} \Big( &\sum_{i}^{|Q|} \sum_{p^-}^{P^-} \mathcal{L}_{Pair}(Q_i, P_i^+, p^-) \\ &+ \sum_{i}^{|Q|} \sum_{p^+}^{P^+} \mathcal{L}_{Pair}(Q_i, P_i^+, p^+) \Big) \end{aligned} \quad (7)$$

**Table 2: Comparison of the computational cost of dense retrieval training methods. The GPUs refer to classes: V100 stands for a server-grade GPU with ≥ 32 GB memory; GTX refers to a consumer-grade GPU ≥ 11GB memory (GTX 1080Ti or better).**

| | Training | Min. GPU | Batch Size | KD Teacher | Added Cost (per Sample) | ↻ Passage Sampling | Index Refresh | Misc. Costs |
|---|---|---|---|---|---|---|---|---|
| | **Standalone** | 1× GTX | 32 | – | – | – | – | – |
| [42] | **ANCE** | 8× V100 | 32 | – | – | ✓ | 10K batches | +1 BM25-trained checkpoint |
| [44] | **LTRe** | 1× GTX | 32 | – | – | ✓ | 1× | +1 ANCE checkpoint |
| [14] | **Margin-MSE** | 1× GTX | 32 | $\text{BERT}_{\text{CAT}}$ | × 1-3 | – | – | – |
| [24] | **TCT** | 1× V100 | 96 | ColBERT | × 1 | – | – | +1 BM25-trained checkpoint |
| [11] | **RocketQA** | 8× V100 | 4,000 | $\text{BERT}_{\text{CAT}}$ | × > 13 | ✓ | 2× | 4× cycles of training |
| | **TAS-Balanced** | 1× GTX | 32 | $\text{BERT}_{\text{CAT}}$ + ColBERT | × 1-4 | – | – | 1× query clustering |

Here, for simplicity and harmony between the dual-teacher signals we re-use the $\mathcal{L}_{Pair}$ loss, but the teacher model $M_t$ is now ColBERT. Additionally, we studied list-based losses that score each query with the full set of all passages in the batch, as shown in Section 5.1, and found Margin-MSE to be the best choice. We compute the total loss as the weighted combination (where $\alpha$ is a hyperparameter to steer the influence) of the pairwise loss $\mathcal{L}_{Pair}$ and the in-batch loss $\mathcal{L}_{InB}$ as follows:

$$\mathcal{L}_{DS}(Q, P^+, P^-) = \mathcal{L}_{Pair}(Q, P^+, P^-) + \mathcal{L}_{InB}(Q, P^+, P^-) \times \alpha \quad (8)$$

Following, the findings of Hofstätter et al. [14] and Ding et al. [11] we do not use the binary relevance labels provided by MSMARCO directly and only rely on the teacher supervision signal, which confirms in almost all cases the binary relevance ordering. This allows our method to be applied to unsupervised scenarios, where training data is generated and scored by trained teacher models without human assessments.

## 2.4 Other Dense Retrieval Training Methods

Improving the training of the $\text{BERT}_{\text{DOT}}$ model is a rapidly evolving field in neural IR with a variety of approaches with different training costs. To give a structured overview of the state of the field, we summarize and compare the most related work for dense passage retrieval training with our TAS-Balanced approach in Table 2. We identify three main drivers of computational cost per training method that lead to a minimum GPU requirement per method. First, the recommended batch size; second, whether knowledge distillation is used; and third, if a dynamic index refresh is needed during training. The standalone training of the $\text{BERT}_{\text{DOT}}$ model only uses binary relevance labels and BM25-sampled negative passages [21]. While it offers the lowest cost training, its results are inferior to the other methods, as we show in Table 4.

The ANCE [42] training swapped BM25-generated negative samples for negative samples retrieved from an index that needs to be refreshed fully every 10K batches, which according to Xiong et al. [42] requires 8 GPU-hours every 10K batches for MSMARCO. Zhan et al. [44] built upon ANCE with LTRe training by continuing to train the query encoder with a fixed passage encoder module.

The pairwise Margin-MSE training [14] showed how pairwise knowledge distillation benefits from an ensemble of $\text{BERT}_{\text{CAT}}$ teachers. With tightly coupled teachers (TCT), Lin et al. [24] showed the benefit of utilizing a ColBERT teacher model for in-batch negative signals. Both approaches add teacher inference overhead to the training. However, this can be mitigated by computing the teacher output once and re-using it.

Ding et al. [11] showed with RocketQA a multi-generational process of training $\text{BERT}_{\text{DOT}}$ student and $\text{BERT}_{\text{CAT}}$ filtered negative passage sampling. They also showed how a very large batch size of 4,000 leads to large gains in accuracy on the sparse MSMARCO-DEV labels. Combined they require an enormous compute capacity (as the batch size has to fit into the GPU memory simultaneously) and time requirement for training a single instance.

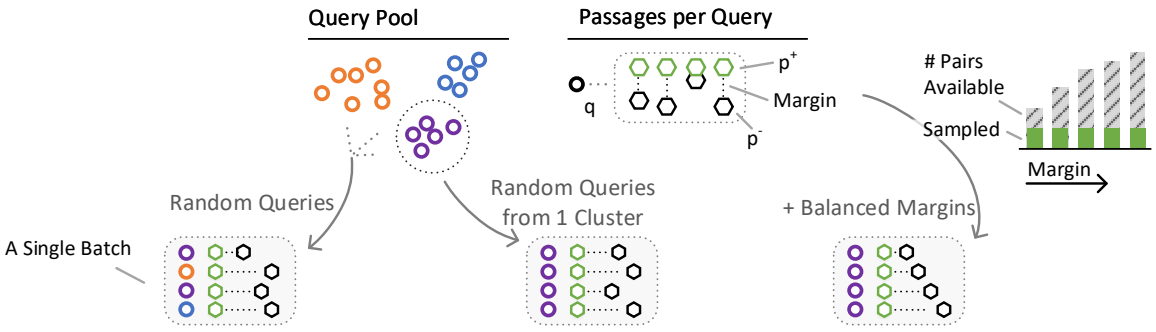
Apart from specifically training dense retrieval models, knowledge distillation has gained popularity, with general-purpose BERT-style models [18, 34] as well as a range of applications in IR: from sequential recommendation models [36], BERT-based retrieval chatbots [37], BERT-based Question Answering [16], reducing the size of the $\text{BERT}_{\text{CAT}}$ passage re-ranking model [5, 12], to dense keyword matching in sponsored search [25].

The composition or sampling of training batches spans all machine learning application fields. Many advances were made especially in computer vision: whether to create synthetic negative samples for contrastive learning [20], unsupervised image cluster learning [4], or changing the image mixture of batches for self-supervised representation learning [35]. In IR, Cohen et al. [6] demonstrated that the sampling policy for negative samples plays an important role in the stability of the training, and MacAvaney et al. [27] adapted the training procedure by shifting samples to the beginning which are estimated to be easy.

# 3 TOPIC AWARE SAMPLING

Clustering data has a long history in IR, for example in the cluster hypothesis concerned with retrieving clustered documents [17, 39]. Inspired by these fundamental findings, we turn to clustering queries, as it is much more efficient than clustering passages, because we have fewer queries than passages available in the MSMARCO training data and each query is more rapidly encoded. We cluster queries to sample out of clusters for our topic aware training batches. We balance the passage pair selection to cover close and distant passage pairs uniformly. This reduces the amount of high-margin (low information) passage pairs. We combine the sampling with an efficient dual-teacher supervision method that combines pairwise and in-batch negative teachers.

Typically, neural networks trained using gradient descent methods consider a collection of training samples together as a batch, for a single update to the network parameters. The commonly used




**(a) Random Sampling** **(b) TAS: Topic Aware Sampling** **(c) TAS-Balanced**

**Figure 2: Comparison of batch sampling strategies. Each strategy has access to a pool of (clustered) queries; where each query has a set of relevant and non-relevant passage pairs with $\text{BERT}_{\text{CAT}}$ score margins.**

approach to compose such a batch $\mathcal{B}$ with size $b$ in the retrieval task is to take a sample of a query $q$, a relevant $p^+$ and a non-relevant $p^-$ passage randomly from the training data of all queries $Q$ and passage pairs per query $P_q$, as shown in Figure 2 (a). Formally:

$$\mathcal{B} = \{(q, p^+, p^-) \mid q \in rand(Q, b), \; p^+, p^- \in rand(P_q)\} \tag{9}$$

where $rand(X, y)$ is a random sampling method of $y$ samples (1 if omitted) from the population $X$ without replacement. With hundreds of thousands of possible training queries, this random sampling produces a collection of queries per batch that cover completely different topics. Using an in-batch negative loss, where each query interacts not only with its own passages, but all others in the batch as well, has very little information gain from those random in-batch interactions. In-batch negatives offer a great promise of "re-using" already computed representations in the loss function.

**TAS.** To fulfill the promise of improved training with in-batch negatives, we propose a Topic Aware Sampling (TAS) strategy, as depicted in Figure 2 (b). Before training, we group all training queries into $k$ clusters with k-means clustering [29], using their baseline representation vectors and minimize:

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{q \in C_i} ||q - v_i||^2 \tag{10}$$

where $v_i$ is the centroid vector of the group $C_i$. The results of this one-time and very efficient procedure are topically related clusters, as shown in the example Figure 1. Now, instead of randomly selecting queries out of the full pool of queries, we randomly select $\lfloor b/n \rfloor$ queries out of $n$ random clusters from $C$ to create a batch:

$$\mathcal{B} = \{(q, p^+, p^-) \mid \overbrace{q \in rand(rand(C, n), \lfloor b/n \rfloor)}^{\text{Topic Aware Sampling}} \; p^+, p^- \in rand(P_q)\} \tag{11}$$

**TAS-Balanced.** As a further refinement, we augment TAS with the need to balance the pairwise margins. Naturally, most queries have fewer relevant passages than non-relevant ones [45]. We define negative passages to be *easy* when they are further away from the positive passage in terms of the teacher model margin. To create a balanced sampling, based on the static margins of the pairwise teacher model $M_t$, as shown in Figure 2 (c), we define a method $H$ that filters passage pairs based on $h$ ranges of size $m$, that uniformly cover the minimum $m_{min}$ to the maximum margin per query:

$$H(P_q, i) = \{(p^+, p^-) \mid m_{min} + i \times m \geq M_t(q, p^+) - M_t(q, p^-) < m_{min} + (i+1) \times m\} \tag{12}$$

Similar to sampling clusters first, and then queries, we sample a range first and then sample from the filtered pairs to unskew the distribution of sample passage pairs. Together, this yields our TAS-Balanced batch sampling strategy:

$$\mathcal{B} = \{(q, p^+, p^-) \mid \overbrace{q \in rand(rand(C, n), b),}^{\text{Topic Aware Sampling}} \underbrace{p^+, p^- \in rand(H(P_q, rand(0..h)))}_{\text{Balanced Margin Sampling}}\} \tag{13}$$

The random sampling does not slow down our training loop as we conduct this batch composition concurrently in a sub-process and queue batches. For training we continuously sample new batches and do not repeat the same batch in multiple epochs. Rather than training for a certain number of epochs, our early-stopping approach detailed in Section 4.3 decides when to stop training.

## 4 EXPERIMENT DESIGN

Our main training and inference dependencies are PyTorch [32], HuggingFace Transformers [41], and Faiss [19], which we use for query clustering as well as brute-force nearest neighbor retrieval.

### 4.1 Passage Collection & Query Sets

We use the MSMARCO-Passage [2] collection with the sparsely-judged MSMARCO-DEV query set of 6,980 queries (used in the leaderboard) as well as the densely-judged query sets of 43 and 54 queries derived from TREC-DL '19 [7] and '20 [8]. For TREC graded relevance (0 = non relevant to 3 = perfect) we use the recommended binarization point of 2 for MRR, MAP, and recall. MSMARCO is based on sampled Bing queries and contains 8.8 million passages. We use the official BM25-based 40 million training passage-pair samples. We cap the query length at 30 tokens and the passage length at 200 tokens; both values represent generous bounds with few outliers that have more tokens.

## 4.2 Parameter Settings

For our TAS clustering we use a pairwise trained $\text{BERT}_{\text{DOT}}$ model baseline as our source for the query representations. We create 2K clusters from the 400K training queries. A pilot study did not find any noticeable difference in the number of clusters. We set the number $n$ of clusters to sample from to 1 due to our relatively low batch size $b$ of 32 (if not further specified). We balance the margin ranges into 10 bins ($h$). After a pilot study we set the dual-teacher combination hyperparameter $\alpha$ to 0.75 to bring both losses into the same range, as the in-batch loss, taking into account more data points, is consistently higher than the pairwise loss. We use the Adam [23] optimizer with a learning rate of $7 \times 10^{-6}$.

As a basis for all our $\text{BERT}_{\text{DOT}}$ and ColBERT instances we use a 6-layer DistilBERT [34] encoder as their initialization starting point. Each instance starts fresh from this DistilBERT checkpoint; we do not use generational retrieval-trained model checkpoints. We trained our ColBERT teacher model with the teacher pairwise signals. While we always used the static pairwise signals, due to studying many different batch compositions we implemented the ColBERT in-batch teacher as a dynamic sub-process running either on the same GPU for a batch size of 32 or an independent GPU for batch size 96 and 256. For the BM25 baseline we use Anserini [43].

## 4.3 Approximate Retrieval Early Stopping

We aim to train our neural retrieval models for as long as training improves the model, to facilitate the fairest comparison of our baselines and novel contributions. This is to not disadvantage methods that might take longer to train, but eventually catch up. We created an approximated early stopping set for all our experiments by indexing a pairwise trained baseline model and retrieving the top 100 passages for 3,200 queries uniformly sampled from the larger DEV-49K set, which are distinct from the DEV-7K and TREC evaluation sets. Additionally, we added all relevant passages if they have not been retrieved by the baseline already. Evaluating our early stopping set takes 5 minutes and we evaluate it every 4K steps; we stop training a model after 30 evaluations have not improved the nDCG@10 metric, which usually stops after 700-800K steps.

# 5 RESULTS

In this section we discuss our research questions and present our results: We compare to internal baselines of different teacher and sampling modes; compare our results to external baselines; study the robustness of our TAS method; and finally provide insights into the use of TAS in a broader re-ranking system. Except for the last point, we present results of the trained $\text{BERT}_{\text{DOT}}$ model using a nearest neighbor search across all MSMARCO passages without re-ranking or fusion of results.

## 5.1 Source of Effectiveness

With our proposal to change the batch sampling on one hand and the teacher supervision on the other, we carefully study the effects of each change in:

**RQ1** How effective are TAS and TAS-Balanced batch sampling techniques with single and dual-teacher supervision?

**Table 3: Analysis of TAS-Balanced & dual-supervision using different loss methods for in-batch negative signals. nDCG & MRR cutoff 10. *Stat. sig. difference w/ paired t-test ($p < 0.05$)***

| Loss | TREC-DL'19 | | TREC-DL'20 | | MSM. DEV | |
|---|---|---|---|---|---|---|
| | nDCG | R@1K | nDCG | R@1K | MRR | R@1K |
| **KLDiv** | .681 | .783 | .673 | .831 | .334 | .964 |
| **ListNet** | .687 | .788 | .668 | .829 | $.338^{k}$ | $.966^{k}$ |
| **Lambdarank** | .704 | $.812^{kn}$ | .682 | .840 | $\mathbf{.342}^{kn}$ | $.971^{kn}$ |
| **Margin-MSE** | **.712** | $\mathbf{.845}^{kln}$ | **.693** | $\mathbf{.865}^{kln}$ | $.340^{k}$ | $\mathbf{.975}^{kln}$ |

For pairwise knowledge distillation, Hofstätter et al. [14] showed their proposed Margin-MSE loss to outperform other options, therefore we fix the use of the Margin-MSE loss for the pairwise teaching part and examine the effect of different losses for additional in-batch negative training for our TAS-Balanced strategy in Table 3.

We study two different types of loss functions: First are list-based losses (KL Divergence, ListNet [3], and the LambdaLoss version nDCG2 [40]) where we build a ranked list of in-batch and pairwise negatives per query and second the pairwise Margin-MSE loss that repeats the relevant passage per query to pair with all other passages from the in-batch negative pool of a batch.

We find in Table 3 that the Margin-MSE loss outperforms other list-based loss variants in most metrics across our three query sets. The change is especially noticeable in the recall metric on the two TREC-DL query sets. The Margin-MSE loss in comparison to the list-based losses optimizes the $\text{BERT}_{\text{DOT}}$ model to follow the teacher score distribution and not just the general ordering of in-batch negatives. We hypothesize the reason for the better Margin-MSE results is because it is advantageous to use a homogeneous loss between both teachers and, because list-based losses only observe ordering and the in-batch negatives are still an incomplete set of all available orderings, the score optimization is more precise.

Our main ablation results in Table 4 investigate two axes: the type of teacher supervision and the type of sampling with all possible combinations between our proposed methods. We also provide a baseline of a standalone-trained $\text{BERT}_{\text{DOT}}$ model with random batch sampling and binary relevance labels only.

The first teacher scenario uses only the pairwise teacher ensemble scores. Comparing the pairwise teacher with the standalone model, we already see significant gains over all metrics. TAS sampling alone does not change the results much and even decreases TREC results slightly. This is an expected result, as the TAS sampling is geared towards in-batch negative training, and should not strongly influence training on queries independently. The TAS-Balanced procedure, on the other hand, improves most metrics for pairwise teaching by 1 percentage point or more, as the balanced margin sampling influences the pairwise supervision.

Using in-batch negatives and a single ColBERT teacher model for supervision with the Margin-MSE loss shows worse results for the original random sampling than the pairwise teacher on the same setting. Here, the TAS strategy provides a strong boost for the results, across all three collections. The TAS-Balanced strategy again improves results for two of the three collections.

Finally, using our novel dual-supervision strategy we observe the same pattern again: TAS improves over random sampling, and TAS-Balance improves over TAS for the best results on almost any

**Table 4: Ablation results of random, TAS, and TAS-Balanced sampling strategies. *(paired t-test; $p < 0.05$)***

| Teacher | Sampling | TREC-DL'19 | | | TREC-DL'20 | | | MSMARCO DEV | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | nDCG@10 | MRR@10 | R@1K | nDCG@10 | MRR@10 | R@1K | nDCG@10 | MRR@10 | R@1K |
| None | Random | .602 | .781 | .714 | .602 | .782 | .757 | .353 | .298 | .935 |
| Pairwise ($BERT_{CAT}$) | Random | .687 | .851 | .767 | .654 | .812 | .801 | .385 | .326 | .958 |
| | TAS | .677 | .851 | .769 | .650 | .820 | .819 | .385 | .325 | .957 |
| | TAS-Balanced | .686 | .866 | .783 | .665 | .823 | $.825^{r}$ | $.393^{rt}$ | $.334^{rt}$ | $.963^{r}$ |
| In-Batch Neg. (ColBERT) | Random | .680 | .857 | .745 | .631 | .773 | .792 | .372 | .315 | .951 |
| | TAS | .706 | .886 | .799 | $.667^{r}$ | .821 | $.826^{r}$ | $.396^{r}$ | $.336^{r}$ | $.968^{r}$ |
| | TAS-Balanced | **.716** | **.910** | .800 | $.677^{r}$ | .810 | $.820^{r}$ | $.397^{r}$ | $.338^{r}$ | $.968^{r}$ |
| Pairwise + In-Batch | Random | .695 | .891 | .787 | .673 | .812 | .839 | .391 | .331 | .968 |
| | TAS | .713 | .878 | .831 | .689 | .815 | $.862^{r}$ | $.401^{r}$ | $.338^{r}$ | $.973^{r}$ |
| | TAS-Balanced | .712 | .892 | **.845** | **.693** | **.843** | $\mathbf{.865}^{r}$ | $\mathbf{.402}^{r}$ | $\mathbf{.340}^{r}$ | $\mathbf{.975}^{rt}$ |

evaluated metric. When we look at the differences between the in-batch teaching and the dual-teacher we see that, especially on the recall metrics, the dual-teacher outperforms the single teacher by a large margin on all three query sets. The nDCG and MRR results are improved for dual-supervision on two out of the three query sets and the remaining TREC-DL'19 results are tied. Because of these results, we recommend using the dual-supervision and TAS-Balanced sampling as the main configuration and we use it throughout the paper for our analysis.

TAS-Balanced uses randomized sampling out of cluster, query, and passage-pair populations extensively. To be confident in our results, we need to investigate if we did inadvertently overfit our approach to a certain setting and study:

**RQ2** How robust is TAS-Balanced to different randomization?

In Table 5 we present the results of our robustness analysis for TAS-Balanced and dual-supervision with different random seeds that guide the ordering and selection of the training samples. Every instance had access to the same data, training configuration, teacher models – the only difference is the random ordering of clusters, queries and passage-pairs. We find overall low variability in our results, especially on the 6,980 test queries of MSMARCO-DEV. For the TREC-DL sets we have many fewer queries – 43 and 53 for TREC-DL'19 and '20 respectively – and still our robustness analysis shows a standard deviation of the results under a single point change in both nDCG@10 and recall. The biggest variation is on the nDCG@10 metric of TREC-DL'20, however the recall shows a lower variance than the recall on TREC-DL'19. This result gives us great confidence in the efficacy of our TAS-Balanced training.

## 5.2 Comparing to Baselines

In this section we focus on standalone $BERT_{DOT}$ retrieval results from different training methods and compare our results with related work to answer:

**RQ3** How does our TAS-Balanced approach compare to other dense retrieval training methods?

We present the dense retrieval results for models trained on the MSMARCO collection in Table 6, first the baselines and then our TAS-Balanced results using different training batch size settings. Important for the comparison of different $BERT_{DOT}$ training techniques is the number of Transformer encoder layers, which linearly determines the indexing throughput and query encoding latency, as well as the training batch size which influences the GPU memory requirements. The TREC-DL'20 query set was recently released, therefore most related work is missing results on these queries. We observe that the methods not using knowledge distillation and larger encoders (ANCE, LTRe) are outperformed on TREC-DL'19 by those that do use teachers (TCT, Margin-MSE), however on the sparse MSMARCO-DEV the result trend turns around. RocketQA on MSMARCO-DEV only outperforms all other approaches when using a batch size of 4,000; RocketQA using 128 samples per batch – more than any other method, but the lowest published by the authors – is outperformed by all other methods.

**Table 5: Random-robustness analysis of five instances of TAS-Balanced dual-supervision each using different sampling orders across clusters, queries, and passage pairs. *Stat. sig. difference w/ paired t-test ($p < 0.05$)***

| Inst. | TREC-DL'19 | | TREC-DL'20 | | MSMARCO DEV | |
|---|---|---|---|---|---|---|
| | nDCG@10 | R@1K | nDCG@10 | R@1K | MRR@10 | R@1K |
| **A** | .712 | .845 | .693 | $.865^{bc}$ | .340 | .975 |
| **B** | .713 | .833 | .684 | .859 | .341 | .974 |
| **C** | .716 | .844 | .679 | .859 | .341 | $.975^{b}$ |
| **D** | .712 | .838 | .688 | .861 | .339 | .974 |
| **E** | .705 | .841 | $.701^{bc}$ | .862 | .339 | .974 |
| **Avg.** | .712 | .840 | .689 | .861 | .340 | .975 |
| **StdDev.** | .004 | .005 | .008 | .003 | .001 | .001 |

Our TAS-Balanced results are in the last section of Table 6. We evaluated our 6 layer encoder model on three different training batch sizes (32, 96, and 256). Between the different batch sizes, we only see a clear trend of improvement on MSMARCO DEV, but not on the TREC collections, there the results are inconsistent, albeit with small differences, that fall in the standard deviation of our robustness analysis in Table 5. This leads us to believe that increasing the batch size is a source of overfitting on the sparse MSMARCO labels. Our TAS-Balanced models outperform all other dense retrieval training methods on both TREC-DL query sets, which show very similar trends: nDCG@10 by at least 4%; MRR@10 by 3%; and Recall@1K, where the margin is the highest with at least 9% improvement over the respectively best related work baseline. TAS-Balanced also shows consistently strong results on the

**Table 6: Dense retrieval results of $\text{BERT}_{\text{DOT}}$ for baseline training and our TAS-Balanced training. *L# = Transformer layers Stat. sig. difference w/ paired t-test ($p < 0.05$) b=BM25; T=TCT; M=Margin-MSE; 3=TAS-B 32; 9=TAS-B 96; 2=TAS-B 256***

| Training Type | Encoder | L# | Batch Size | TREC-DL'19 nDCG@10 | TREC-DL'19 MRR@10 | TREC-DL'19 R@1K | TREC-DL'20 nDCG@10 | TREC-DL'20 MRR@10 | TREC-DL'20 R@1K | MSMARCO DEV nDCG@10 | MSMARCO DEV MRR@10 | MSMARCO DEV R@1K |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Baselines** | | | | | | | | | | | | |
| BM25 | – | – | – | .501 | .689 | .739 | .475 | .649 | .806 | .241 | .194 | .868 |
| [42] ANCE | BERT-Base | 12 | 32 | .648 | – | – | – | – | – | – | .330 | .959 |
| [44] LTRe | BERT-Base | 12 | 32 | .661 | – | – | – | – | – | – | .329 | .955 |
| [44] ANCE + LTRe | BERT-Base | 12 | 32 | .675 | – | – | – | – | – | – | .341 | .962 |
| [11] RocketQA | ERNIE-Base | 12 | 4,000 | – | – | – | – | – | – | – | **.364** | – |
| | | | 128 | – | – | – | – | – | – | – | .309 | – |
| [24] TCT | BERT-Base | 12 | 96 | .670 | – | .720 | – | – | – | – | .335 | .964 |
| [24] TCT (ours) | DistilBERT | 6 | 32 | .680$^{b}$ | .857$^{b}$ | .745 | .631$^{b}$ | .773$^{b}$ | .792 | .372$^{b}$ | .315$^{b}$ | .951$^{b}$ |
| [14] Margin-MSE | DistilBERT | 6 | 32 | .697 | .868 | .769 | – | – | – | .381 | .323 | .957 |
| [14] Margin-MSE (ours) | DistilBERT | 6 | 32 | .687$^{b}$ | .851$^{b}$ | .767 | .654$^{b}$ | .812$^{b}$ | .801 | .385$^{bt}$ | .326$^{bt}$ | .958$^{bt}$ |
| **Ours** | | | | | | | | | | | | |
| TAS-Balanced | DistilBERT | 6 | 32 | .712$^{b}$ | .892$^{b}$ | **.845**$^{btm}$ | **.693**$^{btm}$ | **.843**$^{b}$ | .865$^{btm}$ | .402$^{btm}$ | .340$^{btm}$ | .975$^{btm}$ |
| | | | 96 | **.722**$^{btm}$ | **.895**$^{b}$ | .842$^{tm}$ | .692$^{btm}$ | .841$^{b}$ | .864$^{btm}$ | .406$^{btm}$ | .343$^{btm}$ | .976$^{btm}$ |
| | | | 256 | .717$^{btm}$ | .883$^{b}$ | .843$^{tm}$ | .686$^{btm}$ | .843$^{b}$ | **.875**$^{btm}$ | **.410**$^{btm39}$ | .347$^{btm39}$ | **.978**$^{btm3}$ |

sparse MSMARCO DEV, where we outperform all other baselines, especially on Recall@1K. The only stronger baseline is RocketQA's 4,000 batch size instance; however, as we discussed, this is only due to the larger batch size and not because of their approach, as we strongly outperform (+10% on MRR@10) their 128 batch size instance with a batch size as low as 32.

At this point we want to take a step back and examine the results from a perspective before the *neural revolution*: Our TAS-Balanced trained $\text{BERT}_{\text{DOT}}$ dense retriever, which has comparable query latency with BM25, outperforms BM25 by 44% on nDCG@10 and 9-14% on Recall@1K on TREC'19 & '20. Our work is only the latest in an enormous progress the community made the last few years.

## 5.3 TAS-Balanced Retrieval in a Pipeline

Although the quality of our top-10 results allows use of our $\text{BERT}_{\text{DOT}}$ model as a standalone retriever, usually a ranking system is a hybrid combining different relevance signals. Thus, we investigate:

**RQ4** How well suited is our TAS-trained dense retriever as a first-stage module in terms of recall and re-ranking gains?

We create two pipelines: First, we fuse our TAS-Balanced retriever with docT5query, a sparse passage expansion based retriever, following the setup of Lin et al. [24]. Second, we re-rank our results with the state-of-the-art mono-duo-T5 re-ranking model following Pradeep et al. [33].

As a first step, we examine the usability of different first-stage retrievers in terms of their recall at different cutoffs in Figure 3. These candidates can then be further used by re-ranking models. We find that on TREC-DL our dense retriever is the first dense retriever to consistently outperform BM25 and docT5query at all examined cutoffs. The fused TAS-Balanced + docT5query results offer another boost of recall, showing us that those two diametrical methods bring different strengths that fit together very well.

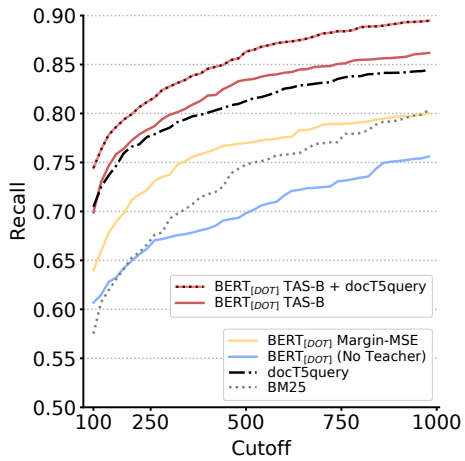



**Figure 3: Recall at different cutoffs for TREC-DL'20**

In Table 7 we compare our multi-stage pipelines grouped by latency. Naturally, the higher the re-ranking depth the higher the full system latency. Analyzing the baselines, we see a large spread in terms of per query latency and effectiveness results. Low-latency results are generally inferior in terms of quality compared to the slower re-ranking results. The powerful re-ranking models are able to improve results when the re-ranking depth is as small as 10 candidates (especially on TREC'20 and MSMARCO-DEV), albeit they show a larger improvement for 1,000 candidates.

Turning to our results, we use the TAS-Balanced with dual-supervision trained on a batch size of 96 for all pipeline experiments.

**Low-Latency.** As with dense retrieval models (in Table 6), TAS-Balanced outperforms other low-latency (<70 ms) systems BM25, DeepCT, and docT5query by large margins in Table 7. Fusing TAS-Balanced together with docT5query further improves Recall@1K, as shown in Figure 3, as well as almost all results across query sets,

**Table 7: Full system results using retrieval and re-ranking pipelines. *Stat. sig. difference w/ paired t-test ($p < 0.05$)***

| Retrieval-Stage | Re-ranking Model | # | Latency (ms) | TREC-DL'19 nDCG@10 | TREC-DL'19 MRR@10 | TREC-DL'19 R@1K | TREC-DL'20 nDCG@10 | TREC-DL'20 MRR@10 | TREC-DL'20 R@1K | MSMARCO DEV nDCG@10 | MSMARCO DEV MRR@10 | MSMARCO DEV R@1K |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Low Latency Systems (<70ms)** | | | | | | | | | | | | |
| [43] BM25 | – | – | 55 | .501 | .689 | .745 | .475 | .649 | .803 | .241 | .194 | .857 |
| [9] DeepCT | – | – | 55 | .551 | – | .756 | – | – | – | – | .243 | .913 |
| [31] docT5query | – | – | 64 | .648$^{b}$ | .799 | .827 | .619$^{b}$ | .742 | .844$^{b}$ | .338$^{b}$ | .277$^{b}$ | .947$^{b}$ |
| TAS-B | – | – | 64 | .722$^{bd}$ | .895$^{b}$ | .842 | .692$^{bd}$ | **.841**$^{bd}$ | .864$^{b}$ | .406$^{bd}$ | .343$^{bd}$ | .976$^{bd}$ |
| TAS-B + docT5query | – | – | 67 | **.753**$^{bdt}$ | **.920**$^{bd}$ | **.882**$^{bdt}$ | **.708**$^{bd}$ | .832$^{b}$ | **.895**$^{bdt}$ | **.425**$^{bdt}$ | **.360**$^{bdt}$ | **.979**$^{bdt}$ |
| **Medium Latency Systems (< 500ms)** | | | | | | | | | | | | |
| [24] TCT + docT5query | – | – | 106 | .739 | – | .832 | – | – | – | – | .364 | .973 |
| [22] ColBERT | – | – | 458 | – | – | – | – | – | – | – | .360 | .968 |
| [44] LTRe | BERT-Large | 10 | 148 | – | – | – | – | – | – | – | .362 | .962 |
| [33] BM25 | duo-T5 | 10 | 388 | .553 | .839 | .745 | .544 | .793 | .803 | .310 | .287 | .857 |
| [33] docT5query | duo-T5 | 10 | 397 | .696$^{b}$ | **.913** | .827 | .658$^{b}$ | .839 | .844$^{b}$ | .411$^{b}$ | .371$^{b}$ | .947$^{b}$ |
| TAS-B | duo-T5 | 10 | 397 | .727$^{b}$ | .877 | .842 | .710$^{b}$ | .864 | .864$^{b}$ | .449$^{bd}$ | .399$^{bd}$ | .976$^{bd}$ |
| TAS-B + docT5query | duo-T5 | 10 | 400 | **.755**$^{bdt}$ | .877 | **.882**$^{bdt}$ | **.726**$^{bdt}$ | **.870** | **.895**$^{bdt}$ | **.463**$^{bdt}$ | **.409**$^{bdt}$ | **.979**$^{bdt}$ |
| **High Latency Systems (> 500ms)** | | | | | | | | | | | | |
| [30] BM25 | BERT-Large | 1K | 3,500 | .736 | – | – | – | – | – | – | .365 | – |
| [33] BM25 | mono-duo-T5 | 1K | 12,800 | .760 | .852 | .745 | .774 | **.888** | .803 | .471 | .409 | .857 |
| [33] docT5query | mono-duo-T5 | 1K | 12,800 | **.773** | **.864** | .827 | **.784**$^{b}$ | .880 | .844$^{b}$ | .488$^{b}$ | .420 | .947$^{b}$ |
| TAS-B | mono-duo-T5 | 1K | 12,800 | .759 | .846 | .842 | .782 | .881 | .864$^{b}$ | .488$^{b}$ | .420 | .976$^{bd}$ |
| TAS-B + docT5query | mono-duo-T5 | 1K | 12,800 | .759 | .848 | **.882**$^{bdt}$ | .783 | .880 | **.895**$^{bdt}$ | **.489**$^{bt}$ | **.421**$^{d}$ | **.979**$^{bdt}$ |

at virtually no latency cost other than merging the two result lists. Across every query set we show the highest Recall@1K with this fused first-stage, followed by our standalone TAS-Balanced retriever. The recall of these first-stage models naturally determines the recall for the re-ranking pipelines. Comparing our low-latency TAS-Balanced (+ docT5query fusion) results with the medium-latency baselines, we observe that in many instances we already outperform or tie methods that are 2-6x slower.

**Medium-Latency.** As soon as we incorporate re-ranking models into a pipeline, we have an explosion of potential options, including the re-ranking depth. For medium-latency systems we re-rank only the top-10 candidates with the duo-T5 re-ranking model. While this top-10 approach only shows modest gains for TREC'19 on baselines and TAS-Balanced retrievers, the gains are much stronger on TREC'20 and MSMARCO-DEV. Following the low-latency pattern, our TAS-Balanced (+ docT5query fusion) re-ranked with duo-T5 outperform other duo-T5 re-ranking pipelines as well as other related systems such as ColBERT or a BERT-large re-ranking system.

**High-Latency.** Our final results employ the full mono-duo-T5 re-ranker at a depth of 1K, where mono-T5 re-ranks the 1K results and duo-T5 then scores the top-50. This pipeline is hardly practical in a production scenario, with 13 seconds latency per query, but gives us a ceiling for the best achievable metrics with current state-of-the-art re-rankers. For MSMARCO-DEV our increased first-stage recall leads to slightly better re-ranking results than the first-stage baselines. However, for the TREC query sets, even though TAS-Balanced shows a higher recall, the mono-duo-T5 re-ranker is (non significantly) better using BM25 & docT5query as retriever. We believe that the mono-duo-T5 re-ranker is not able to take advantage of the increased recall because it has been trained on a BM25 candidate distribution and with dense retrieval we create a shift in the candidate distribution. It is out of the scope of this work to re-train the mono-duo-T5 re-rankers, albeit the importance of training the re-ranker on the first-stage retriever distribution is shown by Gao et al. [13] and Ding et al. [11]. Overall, these are encouraging results to spark future pipeline work based on TAS-Balanced trained $\text{BERT}_{\text{DOT}}$ as a first-stage retrieval model.

## 6 CONCLUSION

We proposed to improve dense passage retrieval training with a cost-neutral topic aware (query) and balanced margin (passage pairs) sampling strategy, called TAS-Balanced. We train the dual-encoder $\text{BERT}_{\text{DOT}}$ model with a dual-supervision of pairwise and in-batch teacher models. Our training only requires under 48 hours on a single consumer-grade GPU and outperforms most other approaches that depend on large server infrastructures, especially on two densely judged TREC-DL query sets. We showed TAS-Balanced works consistently with different random orderings and different teacher supervisions. Additionally, to our standalone retriever, we show how TAS-Balanced interacts with other models in a larger search pipeline. Using TAS-Balanced fused with docT5query results outperforms many systems with 2-6x higher latency. Furthermore, TAS-Balanced is also beneficial for low re-ranking depths. We purposefully set out to design a training technique for dense retrieval that does not depend on large compute servers. We want to give the community the techniques necessary to train strong dense retrieval models with modest hardware, so that the largest possible number of researchers and practitioners can benefit from the neural first-stage improvements and build upon them.

## REFERENCES


[1] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Cross-Domain Modeling of Sentence-Level Evidence for Document Retrieval. In *Proc. of EMNLP-IJCNLP*.

[2] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, and Tri Nguyen. 2016. MS MARCO : A Human Generated MAchine Reading COmprehension Dataset. In *Proc. of NIPS*.

[3] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to Rank: From Pairwise Approach to Listwise Approach. In *Proc. of ICML*.

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2021. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. *arXiv:2006.09882* (2021).

[5] Xuanang Chen, Ben He, Kai Hui, Le Sun, and Yingfei Sun. 2020. Simplified TinyBERT: Knowledge Distillation for Document Retrieval. *arXiv:2009.07531* (2020).

[6] Daniel Cohen, Scott M. Jordan, and W. Bruce Croft. 2019. Learning a Better Negative Sampling Policy with Deep Neural Networks for Search. In *Proc. of ICTIR*.

[7] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2019. Overview of the TREC 2019 Deep Learning Track. In *TREC*.

[8] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2020. Overview of the TREC 2020 Deep Learning Track. In *TREC*.

[9] Zhuyun Dai and Jamie Callan. 2019. Context-Aware Sentence/Passage Term Importance Estimation for First Stage Retrieval. *arXiv:1910.10687* (2019).

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of NAACL*.

[11] Yingqi Qu Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2020. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. *arXiv:2010.08191* (2020).

[12] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2020. Understanding BERT Rankers Under Distillation. *arXiv:2007.11088* (2020).

[13] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. Rethink Training of BERT Rerankers in Multi-Stage Retrieval Pipeline. *arXiv:2101.08751* (2021).

[14] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation. *arXiv:2010.02666* (2020).

[15] Sebastian Hofstätter and Allan Hanbury. 2019. Let's Measure Run Time! Extending the IR Replicability Infrastructure to Include Performance Aspects. In *Proc. of OSIRRC*.

[16] Gautier Izacard and Edouard Grave. 2020. Distilling Knowledge from Reader to Retriever for Question Answering. *arXiv:2012.04584* (2020).

[17] Nick Jardine and Cornelis Joost van Rijsbergen. 1971. The Use of Hierarchic Clustering in Information Retrieval. *Information Storage and Retrieval* 7, 5 (1971), 217–240.

[18] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. TinyBERT: Distilling BERT for Natural Language Understanding. *arXiv:1909.10351* (2019).

[19] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-Scale Similarity Search with GPUs. *arXiv:1702.08734* (2017).

[20] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. 2020. Hard Negative Mixing for Contrastive Learning. *Advances in Neural Information Processing Systems* 33 (2020).

[21] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. *arXiv:2004.04906* (2020).

[22] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proc. of SIGIR*.

[23] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980* (2014).

[24] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2020. Distilling Dense Representations for Ranking using Tightly-Coupled Teachers. *arXiv:2010.11386* (2020).

[25] Wenhao Lu, Jian Jiao, and Ruofei Zhang. 2020. TwinBERT: Distilling Knowledge to Twin-Structured BERT Models for Efficient Retrieval. *arXiv:2002.06275* (2020).

[26] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2020. Sparse, Dense, and Attentional Representations for Text Retrieval. *arXiv:2005.00181* (2020).

[27] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Training Curricula for Open Domain Answer Re-Ranking. In *Proc. of SIGIR*.

[28] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized Embeddings for Document Ranking. In *Proc. of SIGIR*.

[29] James MacQueen. 1967. Some Methods for Classification and Analysis of Multivariate observations. In *Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1. 281–297.

[30] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv:1901.04085* (2019).

[31] Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[32] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic Differentiation in PyTorch. In *Proc. of NIPS-W*.

[33] Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. 2021. The Expando-Mono-Duo Design Pattern for Text Ranking with Pretrained Sequence-to-Sequence Models. *arXiv:2101.05667* (2021).

[34] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, A Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. *arXiv:1910.01108* (2019).

[35] Zhiqiang Shen, Zechun Liu, Zhuang Liu, Marios Savvides, and Trevor Darrell. 2020. Rethinking Image Mixture for Unsupervised Visual Representation Learning. *arXiv:2003.05438* (2020).

[36] Jiaxi Tang and Ke Wang. 2018. Ranking Distillation: Learning Compact Ranking Models with High Performance for Recommender System. In *Proc. of SIGKDD*.

[37] Amir Vakili Tahami, Kamyar Ghajar, and Azadeh Shakery. 2020. Distilling Knowledge for Fast Retrieval-based Chat-bots. In *Proc. of SIGIR*.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proc. of NIPS*.

[39] Ellen M. Voorhees. 1985. The Cluster Hypothesis Revisited. In *Proc. of SIGIR*.

[40] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The LambdaLoss Framework for Ranking Metric Optimization. In *Proc. of CIKM*.

[41] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proc. EMNLP: System Demonstrations*. 38–45.

[42] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. *arXiv:2007.00808* (2020).

[43] Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the Use of Lucene for Information Retrieval Research. In *Proc. of SIGIR*.

[44] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. Learning To Retrieve: How to Train a Dense Retrieval Model Effectively and Efficiently. *arXiv:2010.10469* (2020).

[45] Justin Zobel. 1998. How Reliable are the Results of Large-Scale Information Retrieval Experiments?. In *Proc. of SIGIR*.