# Rethink Training of BERT Rerankers in Multi-stage Retrieval Pipeline

Luyu Gao(✉), Zhuyun Dai, and Jamie Callan

Language Technologies Institute, Carnegie Mellon University, Pittsburgh, USA
{luyug,zhuyund,callan}@cs.cmu.edu

**Abstract.** Pre-trained deep language models (LM) have advanced the state-of-the-art of text retrieval. Rerankers fine-tuned from deep LM estimates candidate relevance based on rich contextualized matching signals. Meanwhile, deep LMs can also be leveraged to improve search index, building retrievers with better recall. One would expect a straightforward combination of both in a pipeline to have additive performance gain. In this paper, we discover otherwise and that popular reranker cannot fully exploit the improved retrieval result. We, therefore, propose a Localized Contrastive Estimation (LCE) for training rerankers and demonstrate it significantly improves deep two-stage models (Our codes are open sourced at https://github.com/luyug/Reranker.).

## 1 Introduction

Recent state-of-the-art retrieval systems are pipelined, consisting of a first-stage heuristic retriever such as BM25 that efficiently produces an initial set of candidate results followed by one or more heavy rerankers that rerank the most promising candidates [11]. Neural language models (LM) such as BERT [7] have had a major impact on this architecture by providing more effective index terms [12] and term weights [5] for heuristic retriever and providing rich contextualized matching signals between query and document for rerankers [4,10].

Intuitively, a better initial ranking provides later stage neural rerankers with more relevant documents to pull up to the top of the final ranking. In a perfect world, a neural reranker recognizes the relevant documents in its candidate pool, inheriting all of the successes of previous retriever. However, simply forming the pipeline by appending a BERT reranker to an effective first-stage retriever does not guarantee an effective final ranking. An improved candidate list sometimes causes inferior reranking. When the candidate list improves, false positives can become harder to recognize as they tend to share confounding characteristics with the true positives. A discriminative reranker should be able to handle the top portion of retriever results and avoid relying on those confounding features.

In this paper, we introduce Localized Contrastive Estimation (LCE) learning. We *localize* negative sample distribution by sampling from the target retriever top results. Meanwhile, we use a *contrastive* form loss which penalizes signals

© Springer Nature Switzerland AG 2021
D. Hiemstra et al. (Eds.): ECIR 2021, LNCS 12657, pp. 280–286, 2021.
https://doi.org/10.1007/978-3-030-72240-1_26

generated from confounding characteristics, preventing the reranker from collapsing.

Experiments on the MSMARCO document ranking dataset show that LCE can better exploit the LMs capability. With the same BERT model, LCE achieves significantly higher accuracy without incurring training or inference overhead.

## 2 Background

Separation of retrieval into stages was introduced naturally due to efficiency-effectiveness trade-off among different ranking models: fast but less accurate model (e.g. BM25) retrieves from the entire corpus while slower but more accurate ones (e.g. BERT) refines ranking in the top candidate list.

Heuristic retrievers like BM25 use matching signals exclusively from exact match and therefore can use inverted list data structure for low latency full corpus retrieval. They are limited by document statistics for scoring. As a fix, deep language models can be leveraged to re-estimate term weights in search index [5,6]. An alternative is adding probable query terms to document [12].

Pre-trained deep LMs [7,14] have demonstrated strong supervised transfer performance on reranking tasks. Popular recent works [4,10] fine-tune BERT [7] with binary classification objective and show it significantly outperforms earlier models. In this paper, we however question if this simple paradigm is sufficient to realize BERT's full potential, especially for high performance deep retrievers that generate candidates consisting of harder negatives.

Alternatives to binary classification objective are the contrastive learning objectives that directly take negatives into account [8]. The popular NCE loss computes scores of a positive instance and several negatives instances, normalize them into probabilities and train the model to give higher probability to the positive instance [16]. The incorporation of negatives in loss prevents the model from collapsing. While contrastive loss has been widely studied in representation learning [2,16], there are few prior works adopting it to train deep LM rerankers.

## 3 Methodologies

**Preliminaries.** We aim to train a BERT reranker to score a query document pair,

$$s = \text{score}(q, d) = \mathbf{v}_{\text{p}}^{\intercal}\, cls(\texttt{BERT}(\textbf{concat}(q, d))) \tag{1}$$

where $cls$ extracts BERT's [CLS] vector and $\mathbf{v}_{\text{p}}$ is a projection vector. We refer to the training technique popularly adopted [4,10] as the *Vanilla method*. It samples query document pairs independently and compute on each individual query-document pair using binary cross entropy (BCE) based on query $q$ document $d$ and corresponding label $(+/-)$,

$$\mathcal{L}_v := \begin{cases} \text{BCE}(\text{score}(q, d), +) & \text{d is positive} \\ \text{BCE}(\text{score}(q, d), -) & \text{d is negative} \end{cases} \tag{2}$$

Vanilla method treats reranker training as a general binary classification problem. However, reranker is unique in nature; it deals with the very top portion of retriever results, each of which may contain many confounding signatures. The reranker is expected to,

- Exile at handling top portion of retriever results.
- Avoid collapsing onto matching with confounding features.

To this end, in this section, we introduce Localized Contrastive Estimation (LCE) loss. The contrastive loss prevents collapsing and localized negative samples focus the reranker on top retriever results.

**Localized Negatives from Target Retriever.** Given a target initial stage retriever and a set of training queries, we use the retriever to retrieve from the entire corpus, generating a set of document rankings for the queries. For each query $q$ then sample from the set $R_q^m$ of top ranked $m$ documents, $n$ non-relevant documents as negatives examples. All sampled documents together form the negative training set. As will be shown in Sect. 6, re-building training set based on the specific target retriever is critical to ensure robust training.

**Contrastive Loss.** After aggregating all negatives sampled from target retriever, we form for each query $q$ a group $G_q$ with a single relevant positive $d_q^+$ and sampled non-relevant negative documents from $R_q^m$. We treat the BERT scoring function as a deep distance function,

$$dist(q,d) = \text{score}(q,d) = \mathbf{v}_{\text{p}}^{\intercal}\, cls(\texttt{BERT}(\textbf{concat}(q,d))) \tag{3}$$

with which we define the contrastive loss for one query $q$ as,

$$\mathcal{L}_q := -log\, \frac{exp(dist(q,d_q^+))}{\sum_{d\in G_q} exp(dist(q,d))} \tag{4}$$

Importantly, here loss and gradient condition not only on the relevant pair but also the retrieved negatives. This effectively helps prevent collapsing onto simple confounding matchings.

**LCE Batch Update.** Putting it all together, we can define the Localized Contrastive Estimation (LCE) loss on a training batch of a set of query $Q$ as,

$$\mathcal{L}_{\text{LCE}} := \frac{1}{|Q|} \sum_{q\in Q, G_q \sim R_q^m} -log\, \frac{exp(dist(q,d_q^+))}{\sum_{d\in G_q} exp(dist(q,d))} \tag{5}$$

Compared to a standard noise contrastive estimation (NCE) loss, LCE uses the target retriever to localize negative samples and focus learning on top portion instead of randomly sampled noisy negatives.

**Table 1.** Document ranking performance measured on MSMARCO dev (left table) and eval set (right table). † indicates statistical significance over Vanilla using a t-test with $p < 0.05$. As the leaderboard eval set only reports aggregated metrics, we cannot report statistical significance.

| Method | MSMARCO Dev MRR@100 | | | |
|---|---|---|---|---|
| | Indri | BM25 | BM25* | HDCT |
| Vanilla | 38.34 | 36.97 | 39.28 | 40.84 |
| LCE | 39.55† | 39.66† | 42.23† | 43.38† |

| Method | MSMARCO Eval MRR@100 |
|---|---|
| PROP (ensemble)[a] | 40.1 |
| BERT-m1 (ensemble)[6] | 39.8 |
| Indri + Vanilla | 33.8 |
| HDCT + LCE (single) | 38.2 |
| HDCT + LCE (ensemble) | **40.5** (1st place) |

[a] PROP_step400K base (ensemble v0.1)
[b] BERT-m1 base + classic IR + doc2query (ensemble)

## 4 Experiment Methodologies

**Dataset and Tasks.** We use the MSMARCO [1] *document* ranking dataset. The dataset contains 3 million documents. A document consists of 3 fields (title, URL, and body) with around 900 words. Models are trained on the train set of 0.37M training pairs. As recommended by MSMARCO organizers, we use the dev set for analysis.

**Initial Stage Retriever.** We experimented with four initial retrievers: Indri, un-tuned BM25, tuned BM25 (denoted as BM25*), and HDCT [5]. The Indri search results come from MSMARCO organizers[1]. We build BM25 indices with the Anserini toolkit [17], from which we produce two sets of search results with the toolkit's default BM25 parameters and a set of tuned parameters suggested by the toolkit authors[2]. HDCT is the SOTA method for augmenting document search indices with term weights re-estimated with BERT; we use the rankings provided by the authors [3]. We input top 100 candidate lists to rerankers.

*Implementation.* Following [4]'s BERT-FirstP setup, we input the concatenated document title, url and body's first 512 queries to the rerankers. Our rerankers are built and trained in mixed precision with PyTorch [13] and based on Huggingface's BERT implementation [15]. We sample negatives from the target retriever's top ranked $m = 100$ documents similar to reranking depth. We train on 4 RTX 2080 ti GPUs, each with a batch of 8 documents. We train for 2 epochs, with a 1e−5 learning rate and a warmup portion 0.1.

## 5 Document Ranking Performance

In Table 1, we summarize ranking performance on MSMARCO document ranking Dev and Eval (leaderboard) queries. Here, both vanilla and LCE use negatives from target retriever. On the dev set, we test rerankers trained with vanilla

[1] https://microsoft.github.io/msmarco/.
[2] https://github.com/castorini/anserini.
[3] http://boston.lti.cs.cmu.edu/appendices/TheWebConf2020-Zhuyun-Dai/.

and LCE loss on each type of the first-stage retriever. We see LCE significantly improves performance with all retrievers. Meanwhile, we see that gain using LCE enlarges as the retriever grows stronger, suggesting it can capture more complicated matching in the improved candidate list, while not being confused by the harder negatives.

The leader board results confirmed the effectiveness of LCE. HDCT+LCE pipeline outperformed the vanilla basline by a large margin. Following other recent leaderboard submissions, we further incorporate model ensemble. Our ensemble entry uses an LCE trained ensemble of BERT, RoBERTa [9] and ELECTRA [3] to rerank HDCT top 100. This submission got first place, achieving the state-of-the-art performance[4].

## 6 Analysis

In this section, we first analyze the effect of number of sampled documents per query in LCE, then the influence of the negative sample localization.

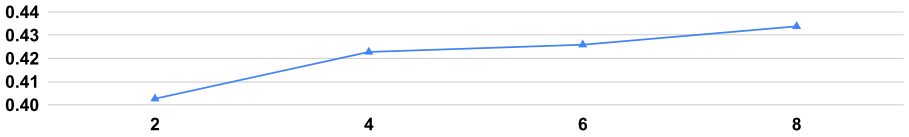


**Fig. 1.** Effect of LCE sample size We plot MRR@100 against sizes.

**Effect of LCE Sample Size.** In Fig. 1, we study the effect of varying the number of sampled documents per query in the LCE loss. We observe a big improvement from size 2 (1 positive, 1 negative) that compute loss scale with a single negative, to size 4 where loss weights are computed with 3 negatives. Further increase in sample size can generate some additional improvements.

**Influence of Negative Localization.** LCE samples negatives from top ranked documents retrieved by target retriever. Here we quantitatively evaluate its importance. Denote retriever used in training for negative sampling *train retriever* and in testing for candidate generation *test retriever*. We use all rerankers from Sect. 5 to rank candidate lists generated by all retrievers and plot results in a heat map Fig. 2. We plot rerankers using different train retrievers on the **horizontal** axis and test retrievers on the **vertical** axis. Each $4 \times 4$ sub-grid corresponds to a training strategy, and sub-grid diagonals correspond to Sect. 5 results. We observe localization benefits both LCE and vanilla methods. The performance of the Vanilla trained reranker *drops* severely when negatives are not localized by test retriever but from a weaker train retriever. Similarly,

[4] On the camera ready date (January 20th, 2021).

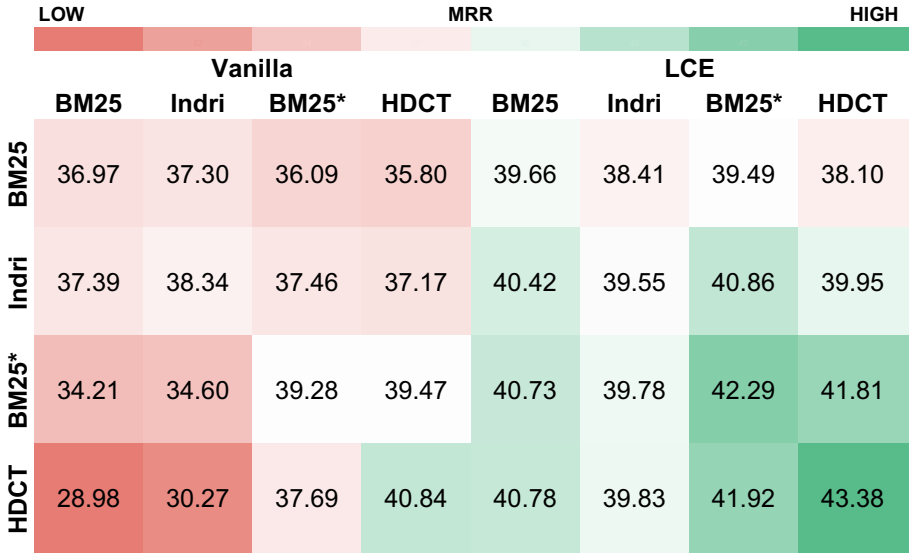


**Fig. 2.** Effects of Train Retriever. The **horizontal** axis is the retriever that generates negatives for training (train retriever); the **vertical** axis is the retriever that generates candidates for testing (test retriever).

rerankers trained with LCE loss also perform better with localization. Interestingly, we do find that the LCE loss can bring some degrees of adaptability to the reranker, making it robust when the test retriever is different from the train retriever.

## 7 Conclusion

Recent research shows promising results on using deep LMs to improve initial retrievers. However, we discovered that previous BERT rerankers could not fully exploit the improved initial rankings. We propose Localized Contrastive Estimation (LCE) learning, to localize training negatives with target retriever, and to use a contrastive loss to penalize matching with confounding characteristics.

Experimental results demonstrate that reranker trained with LCE significantly outperforms its vanilla method trained counterpart using the same LM. Our analysis shows that localizing negatives and having an expressive loss with multiple contrastive negatives are both critical for the success of LCE.

The positive results show that, instead of adopting more advanced LM, it is also possible to improve the performance of existing deep LMs with better learning methods. Meanwhile, before this work, there are few existing work studying the interaction between different deep retrievers and reranker in pipelined retrieval systems. We believe this paper will encourage the community to conduct more systematic research on pipelined IR systems.

## References


1. Campos, D.F., et al.: Ms marco: A human generated machine reading comprehension dataset. arXiv:abs/1611.09268 (2016)
2. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. arXiv:abs/2002.05709 (2020)
3. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: Electra: pre-training text encoders as discriminators rather than generators. arXiv:abs/2003.10555 (2020)
4. Dai, Z., Callan, J.: Deeper text understanding for ir with contextual neural language modeling. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (2019)
5. Dai, Z., Callan, J.P.: Context-aware document term weighting for ad-hoc search. In: Proceedings of The Web Conference 2020 (2020)
6. Dai, Z., Callan, J.: Context-aware term weighting for first stage passage retrieval. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (2020)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
8. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), vol. 2, pp. 1735–1742 (2006)
9. Liu, Y., et al.: Roberta: a robustly optimized Bert pretraining approach. arXiv:abs/1907.11692 (2019)
10. Nogueira, R., Cho, K.: Passage re-ranking with Bert. arXiv:abs/1901.04085 (2019)
11. Nogueira, R., Yang, W., Cho, K., Lin, J.: Multi-stage document ranking with bert. ArXiv abs/1910.14424 (2019)
12. Nogueira, R., Yang, W., Lin, J., Cho, K.: Document expansion by query prediction. arXiv:abs/1904.08375 (2019)
13. Paszke, A., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, vol. 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
14. Peters, M.E., et al.: Deep contextualized word representations. arXiv:abs/1802.05365 (2018)
15. Wolf, T., et al.: Huggingface's transformers: state-of-the-art natural language processing. arXiv:abs/1910.03771 (2019)
16. Wu, Z., Xiong, Y., Yu, S., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3733–3742 (2018)
17. Yang, P., Fang, H., Lin, J.: Anserini: enabling the use of Lucene for information retrieval research. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (2017)