Understanding BERT Rankers Under Distillation

Luyu Gao
Carnegie Mellon University
luyug@cs.cmu.edu

Zhuyun Dai
Carnegie Mellon University
zhuyund@cs.cmu.edu

Jamie Callan
Carnegie Mellon University
callan@cs.cmu.edu

ABSTRACT

Deep language models, such as BERT pre-trained on large corpora, have given a huge performance boost to state-of-the-art information retrieval ranking systems. Knowledge embedded in such models allows them to pick up complex matching signals between passages and queries. However, the high computation cost during inference limits their deployment in real-world search scenarios. In this paper, we study if and how the knowledge for search within BERT can be transferred to a smaller ranker through distillation. Our experiments demonstrate that it is crucial to use a proper distillation procedure, which produces up to nine times speedup while preserving the state-of-the-art performance.[1]

ACM Reference Format:
Luyu Gao, Zhuyun Dai, and Jamie Callan. 2020. Understanding BERT Rankers Under Distillation . In *Proceedings of the 2020 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '20), September 14–17, 2020, Virtual Event, Norway*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3409256.3409838

1 INTRODUCTION

Deep language models, such as BERT [3] learned from large-scale corpora, have pushed the state-of-the-art of search ranking to a new level. All top-performing teams in the TREC 2019 Deep Learning track used fine-tuned BERT for the final re-ranking stage [1]. Ranking with BERT is effective, but requires computation through multiple transformer layers, which is computationally complex. We seek a faster model that preserves BERT-based ranking accuracy.

Recent studies suggest that transformer models are over-parameterized and can be effectively compressed into smaller, faster transformer models through the process of distillation [5, 9, 10]. It is an open question how such distillation affects ranking accuracy.

This paper aims to understand the distillation procedure and its impact on the distilled ranker. We generate distilled rankers of various degrees of compression and with various types of knowledge being distilled. We compare them against the original BERT ranker on a widely-used benchmark dataset to measure changes to ranker accuracy and efficiency. We also study the implications of distillations at training time, exploring convergence behaviour during distillation and training time required for convergence.

The paper's contributions include the following:

- We provide a comprehensive evaluation of distilled BERT rankers. To the best of our knowledge, this is the first work to use distillation techniques to improve the efficiency of BERT rankers in information retrieval.
- We investigate various types of knowledge that can be distilled to the ranker. We show that, with a proper distillation approach, a much smaller ranker can be as effective as the state-of-the-art BERT rankers while being an order of magnitude faster.
- We show that different distillation procedures incur different training cost, and provide recommendations for system developers to trade-off between effectiveness, online evaluation time, and offline training time.

2 BACKGROUND

Deep pre-trained language models (LM) are large neural networks trained on surrounding text signals from large text corpora [3]. These models can then be fine-tuned over other target tasks. Notably, deep LMs such as BERT [3] have achieved state-of-the-art performance in several natural language tasks, including text search [2, 7]. In general, BERT rankers are trained by fine-tuning BERT over search logs, using query and passage as the two input sentences and making relevance prediction conditioned on the output sentence/word representations. Using hundreds of millions of parameters, BERT learns rich language patterns that are useful for ranking. However, the high complexity makes it computationally expensive to run BERT rankers at a large scale [8].

To compress a large neural network, Hinton et al. [4] propose *distillation*. They use the large network as a teacher to train a small network (student) by minimizing the distance between the two models' output prediction probability distributions. Recently, a family of distillation algorithms have been proposed for distilling large teacher transformers to small student transformers [5, 10]. Compared to Hinton et al. [4], they also minimize the distance between student and teacher self attention distributions in the intermediate layers [11]. To the best of our knowledge, there is no prior work studying distilling BERT for search ranking.

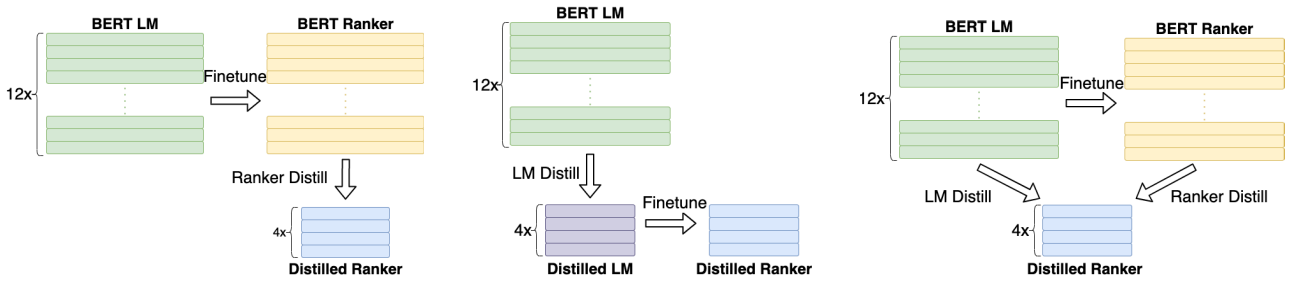
3 DISTILLING BERT FOR RANKING

The power of a BERT ranker is from two main sources: 1) *general-purpose* language modeling knowledge learned in pre-training, and 2) *search-specific* relevance modeling knowledge learned in fine-tuning. We desire to produce a smaller and faster ranker also equipped with both types of knowledge. Fine-tuning can teach search knowledge, turning an LM into a ranker, while distillation can transfer either LM knowledge (LM Distill) or search knowledge (Ranker Distill) from a large teacher to a small student model. As shown in Figure 1, here we detail three distinct methods that combine fine-tuning and distillation to arrive at a smaller ranker from an originally full-sized BERT model: 1. Ranker Distill:

[1] Code is available at https://github.com/luyug/ictir20-distill.

ICTIR '20, September 14–17, 2020, Virtual Event, Norway

ACM ISBN 978-1-4503-8067-6/20/09.
https://doi.org/10.1145/3409256.3409838



(a) Ranker Distill (b) LM Distill + Fine-tuning (c) LM Distill + Ranker Distill

Figure 1: Three approaches to using distillation to create a smaller and faster reranker from the larger BERT model.

distillation is used for search knowledge, 2. LM Distill + Fine-tuning: distillation is used for LM knowledge, and 3. LM Distill + Ranker Distill: distillation is used for both LM and search knowledge.

Ranker Distill The first method distills a BERT ranker's search knowledge to a smaller student ranker, assuming LM knowledge can be learned implicitly. Before running distillation, this method first fine-tunes BERT LM to generate a BERT ranker. It starts distillation by randomly initializing a student transformer with the desired smaller architecture. Then it uses BERT ranker to rank documents, and distills the BERT ranker's ranking behavior to the student.

LM Distill + Fine-tuning The second method teaches general-purpose LM knowledge with distillation, and search-specific knowledge with fine-tuning. It distills a pre-trained BERT language model (BERT LM) to student, assuming it transfers general-purpose LM knowledge enough that the student model can learn search-specific patterns independently through fine-tuning. It first runs a pre-trained BERT over a large text corpora and distills LM predictions from BERT to a smaller transformer, producing a distilled LM. Then, for search-specific knowledge, it treats the distilled LM similarly as the original BERT, and fine-tunes the distilled LM over search logs.

LM Distill + Ranker Distill The third method teaches both general LM knowledge and search-specific knowledge with distillation. In particular, it first uses pre-trained BERT LM as the teacher to produce a distilled LM of desired size. Next, a second round of distillation, Ranker Distill, proceeds to distill a BERT ranker onto the distilled LM, generating the final ranker. The LM Distill provides a good initialization, which may improve generalization ability and robustness. The Ranker Distill trains the student ranker with intermediate attention signals as well as output signals from the teacher ranker, providing richer training signals than the straightforward fine-tuning, which may help the student to perform more similar to the teacher ranker.

4 EXPERIMENTAL METHODOLOGY

Dataset We use the MS MARCO Passage Ranking task dataset [6]. MS MARCO contains around 8.8 million passages and around 0.5 million real queries with judgments as training data. All BERT rankers are tested on a reranking task where the model is asked to rerank the top 1000 documents retrieved by BM25. Two evaluation query sets with different characteristics are used in this work.

MS MARCO Dev Queries This evaluation query set contains 6980 queries from MS MARCO dataset's development set, which has been widely used in prior research [7]. Most of the queries have only one document judged relevant; the relevance labels are binary. Following Nguyen et al. [6], we used MRR@10 to evaluate the ranking accuracy on this query set.

TREC2019 DL Queries This evaluation query set is the official evaluation query set used in the TREC 2019 Deep Learning Track [1]. It contains 43 queries with multiple relevant documents manually judged by NIST assessors with graded relevance labels. On average, a query has 95 relevant documents. Following Craswell et al. [1], we used MRR, NDCG@10, and MAP@1000 to evaluate the ranking accuracy on this query set.

Models for Comparison The BERT ranker serves as our main performance baseline. We train a BERT ranker by fine-tuning the BERT *base-uncased* model over MS MARCO's training set following Nogueira and Cho [7].

For distilled models, we investigate two different student architectures to study impacts from applying different degrees of compression. The first, the **6-layer** model, is a six layer model with transformer hidden dimension of 768. The second, the **4-layer** model, is a four layer model with hidden dimension of 312. The settings are similar to those studied by Jiao et al. [5]. Combining the three distillation methods discussed in Section 3 (Ranker Distill, LM Distill + Fine-tuning, and LM Distill + Ranker Distill) and the two aforementioned architectures (6-layer, 4-layer), we used a total of six types of distilled rankers in the experiment.

The implementation of model distillation was based on the TinyBERT software [5]. LM Distill uses the uncased Bert-base model [3] as the teacher model. For Ranker Distill, the teacher model is the baseline BERT ranker fine-tuned on MS MARCO. We use Adam optimizer with a weight decay of 0.01 and a learning rate of 2e-5 for distillation. We use AdamW optimizer with a learning rate 2e-5 for fine-tuning both BERT ranker and distilled ranker. Training is done on 4 RTX 2080 TI GPUs, and inference on 1 GPU.

5 EXPERIMENT RESULTS

Two experiments studied the distilled ranker's performance during inference, and distillation's implications during training.

5.1 Impacts on Ranking Performance

The first set of experiments aimed to understand, *can distillation speed up a BERT reranker while retaining its effectiveness?* To answer this question, we studied the various distilled models' performance

Table 1: Performance of distilled rankers. Time cost refers to the time to generate rankings for one query by reranking the top 1000 documents. * indicates non-inferiority to BERT ranker with a 95% confidence interval [2].

Method	MARCO Dev Queries	TREC2019 DL Queries			Time (Speedup)
	MRR@10	MRR	NDCG@10	MAP@1000	
BERT ranker (12 layers)	0.3527	0.9349	0.7032	0.4836	2.97s (1.0×)
6-layer distilled rankers					
Ranker Distill	0.3380	0.9271	0.6856	0.4828	1.50s (2.0×)
LM Distill + Fine-Tuning	0.3556*	0.9651*	0.7191*	0.4918*	
LM Distill + Ranker Distill	0.3600*	0.9516*	0.6923	0.4924*	
4-layer distilled rankers					
Ranker Distill	0.3286	0.9351	0.6693	0.4589	0.33s (9.0×)
LM Distill + Fine-Tuning	0.3320	0.9496*	0.6812	0.4609	
LM Distill + Ranker Distill	0.3501*	0.9291*	0.6827*	0.4819*	

during inference time, including their effectiveness, efficiency, and robustness to various reranking depth.

Effectiveness and Efficiency Table 1 reports the ranking accuracy of various distilled rankers. We evaluate the rankers at training checkpoints 10K, 100K, 1M, 5M, 10M. For each model, we report the best performance among all checkpoints.

We found Ranker Distill gives the worst performance. LM Distill + Fine-tuning is able to reach original BERT ranker's effectiveness with a *6-layer* distilled ranker, but fails with a *4-layer* distilled ranker. LM Distill + Ranker Distill yields the strongest performance, being non-inferior to original BERT ranker in both cases with statistical significance. Observation is consistent across two evaluation sets.

Table 1 also reports the rankers' speed measured by the time needed to rank one query with 1,000 candidate documents. We test by feeding in data with batch sizes 64, 128, 256 and 512 to the GPU and record the most efficient batch size. The *6-layer*configuration can be 2 times faster than the full BERT ranker, and the *4-layer* distilled rankers can achieve 9 times speedup, thanks to reduction in both model dimension as well as number of model layers. Importantly, it demonstrates that a proper distillation procedure (LM Disitll + Ranker Distill) can compress the BERT ranker into much smaller ones *without hurting effectiveness while being 9 times faster*.

Effects of Different Distillation Procedures Rank Distill straight-forwardly distills a fine-tuned BERT onto a randomly initialized student model. However, based on our results, this is not sufficient for the distilled ranker to recover all of the teacher model's effectiveness. Using Rank Distill alone, the distilled ranker learns everything from the fine-tuned BERT ranker, without explicitly learning general-purpose language knowledge. On the other hand, the two approaches using LM Distill achieve substantially higher performance than Rank Distill, demonstrating that it is critical for the distilled ranker to learn general-purpose language modeling knowledge through LM Distill explicitly.

The LM Distll + Fine-tuning method achieves similar results as the original BERT ranker when using relatively large models (*6-layer*), indicating that it is possible to directly fine-tune a distilled LM for downstream ranking tasks. However, when using a smaller model (*4-layer*), the accuracy of a fine-tuned distilled LM is slightly lower than a fine-tuned BERT. A higher degree of compression may lose too much LM knowledge, hurting the model's ability to adapt to the downstream ranking task. On the contrary, LM Distill + Ranker Distill can be equally accurate as the original BERT ranker using a small model (*4-layer*). Ranker Distill provides richer training signals than the fine-tuning method, helping the student to recover the teacher's effectiveness even under a high compression rate.

To summarize, our results indicate that to distill a BERT ranker effectively, it is critical to explicitly distill the general-purpose language modeling knowledge first through LM Distill. The search-specific knowledge can be learned from direct fine-tuning or Ranker Distill, depending on the desired model size – while a simple fine-tuning is sufficient for larger distilled models, smaller models still need Ranker Distill to be as effective as the original BERT ranker.

Table 2: Robustness of distilled ranker to various reranking depths. Distilled ranker used LM Distill + Ranker Distill, the best configuration found in Table 1.

Depth	BERT ranker	*6-layer*	*4-layer*
10	0.2716	0.2750	0.2717
20	0.2978	0.3035	0.2979
50	0.3237	0.3303	0.3247
100	0.3367	0.3440	0.3376
200	0.3436	0.3516	0.3437
1000	0.3527	0.3600	0.3501

Robustness to Reranking Depth We also investigated the distilled reranker's performance at various reranking depths. Table 2 shows the results of using our best distilled ranker (LM Distill + Rank Distill) to rerank the top 10 to 1000 passages retrieved by BM25. As shown in Table 2, various models' performance is consistent across all reranking depths, further confirming our finding that distillation effectively compresses the model. Furthermore, dropping the reranking depth to 100 results in only a 4% drop in MRR@10. Based on our speed measurement in Table 1, this implies that in cases where a 4% of accuracy loss is acceptable, a 4-layer distilled ranker that runs at the speed of 0.03*s* per query, processing 30 queries per second on a single GPU machine, is practical.

[2]The equivalence is established by rejecting the null hypothesis that distilled ranker is at least 3% worse than original BERT ranker with a 95% confidence interval.

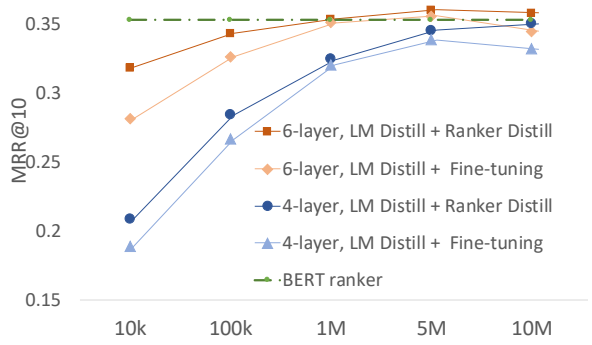


Figure 2: Effects of different amounts of training data.

5.2 Impacts on Training

The previous section shows how distillation affects ranker effectiveness and efficiency at *inference* time. This section discusses how distilled rankers behave at *training* time. As our experiment shows that Ranker Distill alone is not sufficient, here we focus on the training of LM Distill + Fine-tuning and LM Distill + Ranker Distill.

Effects of Model Size on Convergence We initialized the distilled ranker with LM Distill, and train with Ranker Distill or fine-tune using 10K, 100K, 1M, 5M and 10M of training examples. Figure 2 shows the trend of MRR@10. It shows a clear distinction between the smaller model (*4-layer*) and the bigger model (*6-layer*). The *6-layer* model converges quickly, achieving a reasonable performance at 10K training pairs, and close to the full BERT ranker with roughly 100K pairs. Meanwhile, the *4-layer* model converges slower, requiring 5M to 10M pairs to be close to the full BERT ranker. Based on these results, we conclude that smaller distilled models are overall more data-hungry due to the loss of LM knowledge.

Fine-tuning vs. Ranker Distill during Training As discussed previously, LM Distill + Fine-tuning and LM Distill + Ranker Distill can both generate reasonably effective rankers, while the latter is more effective in compression. This experiment aimed to further understand their training behavior. As shown in Figure 2, the same model requires less data to converge with distillation than with fine-tuning. Distillation also generalizes better: accuracy of fine-tuning starts to decrease after training on 5M query-document pairs, indicating potential overfitting. Meanwhile, the accuracy of Ranker Distill has not plateaued yet, showing that for a small model, learning search-specific knowledge through distillation is less prone to overfitting than directly learning from fine-tuning.

Ranker Distill and Fine-tuning also differ in *amount of training time*. Ranker Distill runs both teacher model and student model, leading to higher compute and memory costs. Table 3 shows time taken for various models to reach close to (within 5% of) BERT ranker's performance. As shown in Table 3, fine-tuning is 5 to 10 times faster than Ranker Distill when training. The *4-layer* model, though faster in inference, trains slower and takes 5 times more training steps to be close to BERT ranker's performance (Figure 2).

To summarize, highly compressed rankers require more training data, longer training time, and may fail to converge to best optimum with fine-tuning. One should choose distillation approaches by weighing the importance of off-line training time, online performance, and online response time. When the ranker requires no update, LM Distill + Rank Distill with small models offers the best online-serving speed. When the ranker needs to be frequently updated, doing LM Distill + Fine-tuning is generally preferred.

Table 3: Amount of training time for distilled models to reach the first checkpoint that is close to BERT ranker's performance, i.e., 1M training examples for the 6-layer model and 5M for the 4-layer model.

Model	*6-layer*	*4-layer*
Fine-Tuning	1.1h	1.6h
Ranker Distill	5.5h	14.8h

6 CONCLUSION

In this paper, we demonstrate that, with distillation, one can turn a BERT ranker into a substantially faster ranker, while still preserving the BERT ranker's ranking performance. We evaluated different ways to generate distilled rankers from the original BERT LM and identified that the most robust and effective method is a two-round distillation where both the general-purpose LM knowledge and search-specific knowledge are distilled. In comparison, removal of each leads to an inferior ranker. We also examined distilled models' training behaviors. We found a higher degree of compression introduces more difficulties in model optimization due to loss of knowledge. Though faster in inference, smaller distilled models consume more training data and longer training time.

ACKNOWLEDGMENTS

This work was supported by National Science Foundation (NSF) grant IIS-1815528 and The Boeing Company. Any opinions, findings, and conclusions in this paper are the authors' and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2019. Overview of the TREC 2019 deep learning track. In *TREC (to appear)*.
[2] Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In *The 42nd International ACM SIGIR Conference on Research & Development in Information Retrieval*.
[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019).
[4] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *ArXiv* abs/1503.02531 (2015).
[5] Xiaoqi Jiao, Y. Yin, Lifeng Shang, Xin Jiang, Xusong Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. TinyBERT: Distilling BERT for Natural Language Understanding. *ArXiv* abs/1909.10351 (2019).
[6] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268* (2016).
[7] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *ArXiv* abs/1901.04085 (2019).
[8] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375* (2019).
[9] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv* abs/1910.01108 (2019).
[10] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient Knowledge Distillation for BERT Model Compression. In *EMNLP/IJCNLP*.
[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *ArXiv* abs/1706.03762 (2017).