

# Knowledge Distillation vs. Pretraining from Scratch under a Fixed (Computation) Budget

Minh Duc Bui<sup>▽</sup> Fabian David Schmidt<sup>◇</sup>

Goran Glavas<sup>◇</sup> Katharina von der Wense<sup>▽♣</sup>

<sup>▽</sup>Johannes Gutenberg University Mainz, Germany

<sup>◇</sup>Center For Artificial Intelligence and Data Science, University of Würzburg, Germany

<sup>♣</sup>University of Colorado Boulder, USA

{minhducbui, k.vonderwense}@uni-mainz.de

{fabian.schmidt, goran.glavas}@uni-wuerzburg.de

## Abstract

Compared to standard language model (LM) pretraining (i.e., from scratch), Knowledge Distillation (KD) entails an additional forward pass through a teacher model that is typically substantially larger than the target student model. As such, KD in LM pretraining materially slows down throughput of pretraining instances vis-a-vis pretraining from scratch. Scaling laws of LM pretraining suggest that smaller models can close the gap to larger counterparts if trained on more data (i.e., processing more tokens)—and under a fixed computation budget, smaller models *are able* to process more data than larger models. We thus *hypothesize that KD might, in fact, be suboptimal to pretraining from scratch for obtaining smaller LMs, when appropriately accounting for the compute budget*. To test this, we compare pretraining from scratch against several KD strategies for masked language modeling (MLM) in a *fair* experimental setup, with respect to amount of computation as well as pretraining data. Downstream results on GLUE, however, do *not* confirm our hypothesis: while pretraining from scratch performs comparably to ordinary KD under a fixed computation budget, more sophisticated KD strategies, namely TinyBERT (Jiao et al., 2020) and MiniLM (Wang et al., 2023), outperform it by a notable margin. We further find that KD yields larger gains over pretraining from scratch when the data must be repeated under the fixed computation budget.<sup>1</sup>

## 1 Introduction

Knowledge distillation (KD; Hinton et al., 2015; Jiao et al., 2020) during LM pretraining has emerged as the primary mean of compressing the capabilities of a large pretrained teacher model into a task agnostic smaller student model. KD is praised for yielding high-performing task agnostic small models, mitigating the need for pretraining

Name	Identical	
	Architect.	Compute
DistilBERT (Sanh et al., 2020)	No	No
TinyBERT (Jiao et al., 2020)	Yes	No
MobileBERT (Sun et al., 2020)	No	No
MiniLM (Wang et al., 2020)	No	No
Our Work	Yes	Yes

Table 1: Assessing the fairness of evaluation setups in previous works for task-agnostic masked language models, trained with KD and without KD.

(small models) from scratch, which is typically considered more expensive. The body of existing KD work for MLM (Jiao et al., 2020; Wang et al., 2023), however, typically does not compare KD against pretraining from scratch in a *fair setup*: (i) with the same target models (exactly the same architecture) and (ii) under the same computation budget. Compared to just training the target model from scratch, KD comes with a computational overhead of forward passes through the typically considerably larger teacher model. This, under the same computation budget, allows pretraining from scratch to consume more data (i.e., more tokens) than KD, which leads to the central research question of this work: *in a fair setup where both are given equal overall computation budget, is KD still more effective than pretraining from scratch (No-KD)?* We hypothesize that, under a fair evaluation setup, No-KD may be as effective as KD, rendering KD inconsequential. Our reasoning is based on two observations:

**1) Fair KD Comparison.** A fair comparison, in which both setups are given identical computation budgets (as well as identical target models) eludes existing work on KD. Jiao et al. (2020) compare their model to BERT<sub>Tiny</sub> (Turc et al., 2019), which has the same architecture but employs significantly different training resources than their TinyBERT<sub>Tiny</sub>, preventing a fair comparison. Similarly, Sanh et al. (2020) compare their distilled stu-

<sup>1</sup>Code is available at [https://github.com/MinhDucBui/revisiting\\_distillation](https://github.com/MinhDucBui/revisiting_distillation).

dent solely against the teacher, whereas Sun et al. (2020); Wang et al. (2020) only add comparison against larger pretrained models and competing KD strategies. Even the body of work that focuses on comparing different KD strategies has only recently sought to standardize training and thus enable fair comparisons (Lu et al., 2022; Wang et al., 2023).

**2) Scaling Laws.** Scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022), reveal that, under a fixed computation budget, only a marginal correlation exists between the LM size and its performance: smaller models compensate their lower learning efficiency with the ability to process more tokens within the same budget. While there are ongoing refinements to the scaling law (Hoffmann et al., 2022), it has been consistently reaffirmed by several studies (Geiping and Goldstein, 2023; Bansal et al., 2022; Clark et al., 2022). For instance, Geiping and Goldstein (2023) showcases this behavior by training multiple BERT models with varying architecture sizes for a fix 24 hour duration, resulting in similar loss values across all sizes.

**Contribution** Motivated by the recent findings in the realm of scaling laws and recognizing the absence of a fair comparison between KD and No-KD, our primary contribution lies in the comparison of No-KD against KD strategies for MLM while ensuring a fair setup with regards to compute budget and pretraining data. We initially assess No-KD in an optimal setup, where unlimited pretraining tokens are available within a fixed compute budget. Additionally, we examine the scenario when data is constrained within a fixed compute budget.

Our results reveal that, in the optimal setting, No-KD performs indeed comparably to vanilla-KD, exhibiting an average improvement over vanilla-KD of 0.4 and 0.1 points for 6-, and 12-layer models on GLUE. However, No-KD falls short of surpassing more advanced KD strategies, exemplified by the comparison with TinyBERT and MiniLM. When available data is limited within the fixed compute budget, KD strategies outperform No-KD by an even larger margin: No-KD, though faster, needs more epochs, whereas KD strategies extract more information from limited data.

## 2 Distillation Strategies

**Vanilla-KD** Vanilla-KD for MLM pretraining is set up as follows. A small MLM student is trained

to mimick the predictions for a particular training instance of a large pretrained MLM teacher: The distillation objective is to minimize the soft cross-entropy between the logits  $\mathbf{z}^T$  of the MLM teacher and the logits  $\mathbf{z}^S$  of the MLM student, with a temperature factor  $t$ :  $\mathcal{L}_{\text{pred}} = \text{CE}(\mathbf{z}^T/t, \mathbf{z}^S/t)$ . Following Hinton et al. (2015), the final training loss equally combines  $\mathcal{L}_{\text{pred}}$  with the MLM loss  $\mathcal{L}_{\text{CE}}$  during pretraining.

**TinyBERT** Jiao et al. (2020) distill knowledge by minimizing the mean-squared error (MSE) between latent representations of the MLM student  $S$  and the MLM teacher  $T$  by model layers as follows. First, the embedding matrices of the student ( $\mathbf{E}_S$ ) and the teacher ( $\mathbf{E}_T$ ) are aligned by minimizing the loss  $\mathcal{L}_{\text{embd}} = \text{MSE}(\mathbf{E}^S \mathbf{W}_e, \mathbf{E}^T)$ . The authors further fit the unnormalized attention scores per head  $h$  of the MLM student  $S$  to the MLM teacher  $T$  by optimizing  $\mathcal{L}_{\text{att}} = 1/h \sum_{i=1}^h \text{MSE}(\mathbf{A}_i^S, \mathbf{A}_i^T)$ . Lastly, the output hidden states  $\mathbf{H}^S$  of transformer layers of the student are also regressed onto the corresponding teacher output representations  $\mathbf{H}^T$  by optimizing  $\mathcal{L}_{\text{hid}} = \text{MSE}(\mathbf{H}^S \mathbf{W}_h, \mathbf{H}^T)$ .<sup>2</sup>

**MiniLM** Wang et al. (2020) also mimic the self-attention modules of the MLM teacher. Unlike TinyBert, MiniLM focuses on the last attention module. Wang et al. (2020) minimize the KL-divergence between the self-attention distributions of the MLM teacher and the MLM student. They further minimize the KL-divergence between the value relations of the MLM teacher  $T$  and MLM student  $S$ , i.e.  $\mathcal{L}_{VR} = \frac{1}{A_h |x|} \sum_{a=1}^{A_h} \sum_{t=1}^{|x|} D_{KL}(\mathbf{VR}^T || \mathbf{VR}^S)$ . The value-relation denotes the outer product of values  $\mathbf{V}$  across heads in the last attention module, i.e.  $\mathbf{VR} = \text{softmax}(\frac{\mathbf{V}\mathbf{V}^T}{\sqrt{d}})$ .

## 3 Experiment Setup

**Model Architectures** We experiment with two different teacher and student sizes: First, we use a 12-layer pretrained BERT<sub>base</sub> (Devlin et al., 2019) model (L=12, H=768, A=12, Total Parameters=110M) as the teacher and a randomly initialized 6-layer BERT<sub>6</sub> model (L=6, H=768, A=12, Total Parameters=67M) as the student. We then scale the setting up to a pretrained BERT<sub>Large</sub> (L=24,

<sup>2</sup>The distillation of embeddings  $\mathbf{E}$  and output hidden states  $\mathbf{H}$  is learned up to projection matrices  $\mathbf{W}_{e,h}$  matrices to bridge varying dimensionalities of representations across architectures.

	<b>Total Token Throughput</b>	<b>QNLI</b> (Acc)	<b>SST-2</b> (Acc)	<b>MNLI</b> (Acc)	<b>MRPC</b> (F1)	<b>QQP</b> (Acc)	<b>RTE</b> (Acc)	<b>CoLA</b> (Mcc)	<b>Avg</b>	<b><math>\Delta</math></b>
<i>6-Layer: Unlimited Pretraining Tokens within Fixed Compute Budget</i>										
No-KD <sub>6</sub>	4.6B	86.5	89.8	79.2	87.3	90.1	60.7	<b>47.3</b>	77.3	–
Vanilla-KD <sub>6</sub>	2.6B	87.3	89.2	78.8	87.1	89.7	61.7	44.8	76.9	–0.4
MiniLM <sub>6</sub>	2.6B	88.5	90.6	81.7	90.0	90.3	64.3	43.9	78.5	+1.2
TinyBERT <sub>6</sub>	2.6B	<b>89.5</b>	<b>91.0</b>	<b>82.2</b>	<b>90.3</b>	<b>90.4</b>	<b>67.2</b>	40.8	<b>78.8</b>	+1.5
<i>12-Layer: Unlimited Pretraining Tokens within Fixed Compute Budget</i>										
No-KD <sub>12</sub>	4.6B	87.8	90.1	80.6	86.5	90.3	60.3	<b>51.1</b>	78.1	–
Vanilla-KD <sub>12</sub>	2.6B	86.3	90.5	79.8	88.8	89.9	62.1	48.6	78.0	–0.1
MiniLM <sub>12</sub>	2.6B	<b>90.0</b>	91.2	<b>83.3</b>	90.1	<b>90.9</b>	<b>69.0</b>	49.1	<b>80.5</b>	+1.4
TinyBERT <sub>12</sub>	2.6B	89.5	<b>91.4</b>	82.0	<b>90.8</b>	90.6	65.5	41.1	78.7	+0.6
<i>6-Layer: Limited Pretraining Tokens within Fixed (Increased) Compute Budget</i>										
No-KD <sub>6</sub>	27.9B	88.8	91.2	81.3	88.0	90.4	59.6	50.5	78.5	–
Vanilla-KD <sub>6</sub>	15.4B	86.9	91.1	81.1	89.5	90.3	61.7	<b>58.3</b>	79.8	+1.3
MiniLM <sub>6</sub>	15.6B	90.0	91.5	83.0	<b>90.3</b>	90.6	65.7	50.7	80.3	+1.8
TinyBERT <sub>6</sub>	15.6B	<b>90.5</b>	<b>92.3</b>	<b>83.3</b>	90.2	<b>90.8</b>	<b>67.5</b>	51.8	<b>80.9</b>	+2.4

Table 2: Upper part: optimal scenario for No-KD – unlimited pretraining tokens within a fixed compute budget. Lower part: limited data within a fixed compute budget. We present the performance results on the GLUE development set, maintaining a consistent pretraining wall-clock time across all models within each group. The column **Avg** represents the average performance across all tasks, while  **$\Delta$**  quantifies the average difference between No-KD<sub>xx</sub> and the other distillation strategies.

H=1024, A=16, Total Parameters=340M) teacher and a randomly initialized 12-layer BERT<sub>12</sub> student. To speed up the training pipeline and convergence, we use the implementation of Izsak et al. (2021) for the models.

**Data** We follow BERT (Devlin et al., 2019) and pretrain all models on the Toronto BooksCorpus (Zhu et al., 2015) and English Wikipedia.<sup>3</sup> After MLM pretraining, we finetune and evaluate the models on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018), a collection of diverse natural language understanding tasks.<sup>4</sup>

**Pretraining** We first pretrain a BERT<sub>6</sub> model from scratch (without KD). This model is denoted as No-KD<sub>6</sub>. We further apply the KD strategies (cf. §2), for which BERT<sub>base</sub> (BERT<sub>large</sub>) is the MLM teacher for the 6-layer (12-layer) MLM student. The resulting 6-layer models are indicated with a subscript 6, while the 12-layer models are marked with a subscript 12. Notably, we only employ the last layer for layer-wise distillation, as we confirm the findings of Wang et al. (2023) that distilling knowledge of multiple layers does not yield con-

sistent performance improvements. We refer to Appendix A.2 for additional hyperparameter, hardware details and wall-clock training time.

**Downstream Finetuning** We perform a grid search over batch sizes {16, 32} and learning rates {1e-5, 3e-5, 5e-5, 8e-5} to identify the ideal hyperparameters for each task on the GLUE benchmark. We train all configurations for 5 epochs. We utilize a polynomial learning rate schedule and a maximum sequence of 128.

## 4 Results

### 4.1 Setting: Unlimited Data with Fixed Compute

We assess No-KD for a single epoch and fix the resulting training wall-clock time for the distillation strategies. Within this compute budget, we train on unlimited pretraining tokens without the need for sample repetition. We report our main results in the upper segment of Table 2.

**Low KD Token Throughput** We find that the token throughput of No-KD<sub>6</sub> and No-KD<sub>12</sub> is approximately 1.8 times greater than that of the distillation models. This observation underscores that the presence of a teacher model greatly reduces the speed of pretraining.

<sup>3</sup>In November 2023, we crawled the English Wikipedia using Attardi (2015). The official BookCorpus is no longer accessible; however, it was re-crawled by Kobayashi (2018).

<sup>4</sup>We refer to Appendix A.3 for more information about the GLUE datasets.

**Performance of 6-layer Students** We observe that No-KD<sub>6</sub> surpasses Vanilla-KD<sub>6</sub> by an average of 0.4 points. This result indicates that Vanilla-KD<sub>6</sub> does not exceed pretraining from scratch in a fair setting. However, more advanced KD strategies exhibit notable performance gains over No-KD. On average, TinyBERT<sub>6</sub> outperforms No-KD<sub>6</sub> by 1.5 points, while MiniLM<sub>6</sub> achieves a 1.2 point advantage. These findings suggest that pretraining from scratch falls short in outperforming sophisticated distillation strategies in a fair setup, even when exposed to a higher volume of tokens. The only exception to this trend is CoLA (Corpus of Linguistic Acceptability) (Warstadt et al., 2019), on which No-KD<sub>6</sub> excels.

**Performance of 12-layer Students** We find the same pattern when we double the number of transformer layers in MLM students: Vanilla-KD<sub>12</sub> fails to outperform No-KD<sub>12</sub>, yet it is surpassed by MiniLM<sub>12</sub> by an average of 1.4 points. Notably, No-KD<sub>12</sub> once again exhibits superior performance on the CoLA task compared to other strategies.

**CoLA Performance** No-KD<sub>6</sub> demonstrates superior performance on CoLA, surpassing the next most effective strategy by 2.5 and 2.0 points for 6- and 12-layer models, respectively. We hypothesize that CoLA benefits significantly from masked language modelling, as evidenced by the improved performance of Vanilla-KD on CoLA compared to other distillation strategies, aligning with findings by Wang et al. (2023). Another contributing factor could be the scalability of CoLA with respect to tokens encountered during pretraining. This observation contradicts the results of Liu et al. (2021), who suggest that CoLA can be learned relatively quickly compared to other downstream tasks. However, it aligns with the conclusions of Geiping and Goldstein (2023), who also note that their BERT version, exposed to less data, exhibits subpar performance on CoLA.

## 4.2 Setting: Limited Data with Fixed Compute

To extend our findings, we increase the compute budget while retaining a fixed dataset size. We evaluate this setup with 6-layer MLM students and the 12-layer MLM teacher. The analysis provides an estimate of the viability of No-KD when data repetition is necessary within the fixed compute budget. The results are presented in the lower section of Table 2.

The No-KD<sub>6</sub> model is underperforming, compared to all distillation strategies, including Vanilla-KD<sub>6</sub> by 1.3 points. The performance gap widens even more when compared to MiniLM<sub>6</sub> and TinyBERT<sub>6</sub>, with a substantial difference of 1.8 and 2.4 points on average. We attribute this to the fact that while No-KD benefits from exposure to a larger number of tokens, it also necessitates a larger dataset for effective scaling. Although this requirement can be met in high-resource languages with up-to-date datasets (Kudugunta et al., 2023), it presents a significant challenge in mid to low-resource scenarios. Additionally, No-KD<sub>6</sub> is now being outperformed even on CoLA. These results suggest that CoLA’s performance indeed needs to process a certain quantity of tokens during pretraining to scale effectively, regardless of additional token repetitions: e.g., the performance of Vanilla-KD<sub>6</sub> increases by 13.5 points if scaled from 2.6B unique to 15.4B non-unique pretraining tokens. Interestingly, our findings reveal that Vanilla-KD<sub>6</sub> exhibits the best performance on CoLA, underscoring the advantageous impact of masked language modelling on this particular dataset.

## 5 Discussion

While our study provides insights into a fair evaluation of No-KD and KD for encoder-only models of moderate sizes, revealing negative results for No-KD, it may not cover the full spectrum of model sizes and architectures. For instance, Jha et al. (2023) show that for large decoder-only language models, No-KD performs comparably to Vanilla-KD, aligning with our findings. However, advanced KD strategies like MiniLM exhibit poorer performance than No-KD and Vanilla-KD, challenging both our results and common beliefs about KD regarding large decoder models. This disparity underscores the need for further investigation into a fair KD evaluation across a range of *architectures and scales*. Additionally, we recommend investigating the impact of the teacher budget on performance in the fair setting, a consideration not closely examined in our current work.

## 6 Conclusion

In this work, we investigate our hypothesis that, provided a fair training scenario, model pretraining from scratch yields similar results as KD during pretraining. Our rationale is grounded in recent advancements in scaling laws for language models



and that the literature lacks a fair comparison between No-KD and KD. Our findings demonstrate that our initial assumption does *not* hold true: while, in an optimal setting for No-KD, No-KD performs on par with ordinary KD, it falls short when compared to more sophisticated KD strategies.

## Limitations

Firstly, we acknowledge that assessing the compute budget based on training wall-clock time comes with inherent limitations. As outline in [Kaddour et al. \(2023\)](#), wall-clock time can fluctuate even on identical hardware. This fluctuation may arise from factors such as the utilization of non-deterministic operations or hidden background processes. Nevertheless, we only see negligible variations across different runs for the same training pipeline.

Another limitation of our work pertains to data size. Exploring larger pretraining corpora than ours might be worthwhile, although we note that even within our current data scale, KD consistently outperforms No-KD by a significant margin. Even with potential increases in data size, KD remains valuable as it provides a stronger starting point compared to No-KD.

Lastly, we acknowledge that the pretraining corpus is the same as what the teacher used. This shared corpus might influence KD strategies either positively or negatively.

## Ethics Statement

We acknowledge our exclusive focus on the English language, overlooking the many challenges of other languages. Additionally, we recognize our sole emphasis on performance metrics, neglecting considerations related to the fairness of the resulting models. We also note that our research extensively employed GPU hours for both pretraining and finetuning, with a keen awareness of the environmental and resource implications associated with such usage.

## Acknowledgement

The research in this paper was funded by the Carl Zeiss Foundation, grant number P2021-02-014 (TOPML project).

## References

Giusepppe Attardi. 2015. Wikiextractor. <https://github.com/attardi/wikiextractor>.

Yamini Bansal, Behrooz Ghorbani, Ankush Garg, Biao Zhang, Colin Cherry, Behnam Neyshabur, and Orhan Firat. 2022. [Data scaling laws in NMT: The effect of noise and architecture](#). In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1466–1482. PMLR.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. [SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation](#). In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Aidan Clark, Diego De Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, George Bm Van Den Driessche, Eliza Rutherford, Tom Hennigan, Matthew J Johnson, Albin Cassirer, Chris Jones, Elena Buchatskaya, David Budden, Laurent Sifre, Simon Osindero, Oriol Vinyals, Marc’Aurelio Ranzato, Jack Rae, Erich Elsen, Koray Kavukcuoglu, and Karen Simonyan. 2022. [Unified scaling laws for routed language models](#). In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 4057–4086. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bill Dolan and Chris Brockett. 2005. [Automatically constructing a corpus of sentential paraphrases](#). In *Third International Workshop on Paraphrasing (IWP2005)*. Asia Federation of Natural Language Processing.

Jonas Geiping and Tom Goldstein. 2023. [Cramming: Training a language model on a single GPU in one day](#). In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 11117–11143. PMLR.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. [Distilling the knowledge in a neural network](#).

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. 2022. [An empirical analysis](#)

- of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030. Curran Associates, Inc.
- Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. [How to train BERT with an academic budget](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10644–10652, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Ananya Harsh Jha, Tom Sherborne, Evan Pete Walsh, Dirk Groeneveld, Emma Strubell, and Iz Beltagy. 2023. [How to train your \(compressed\) large language model](#).
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. [TinyBERT: Distilling BERT for natural language understanding](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.
- Jean Kaddour, Oscar Key, Piotr Nawrot, Pasquale Minervini, and Matt J. Kusner. 2023. [No train no gain: Revisiting efficient training algorithms for transformer-based language models](#).
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. [Scaling laws for neural language models](#).
- Sosuke Kobayashi. 2018. [Homemade bookcorpus](https://github.com/soskek/bookcorpus). <https://github.com/soskek/bookcorpus>.
- Sneha Kudugunta, Isaac Caswell, Biao Zhang, Xavier Garcia, Christopher A. Choquette-Choo, Katherine Lee, Derrick Xin, Aditya Kusupati, Romi Stella, Ankur Bapna, and Orhan Firat. 2023. [Madlad-400: A multilingual and document-level large audited dataset](#).
- Zeyu Liu, Yizhong Wang, Jungo Kasai, Hannaneh Hajishirzi, and Noah A. Smith. 2021. [Probing across time: What does RoBERTa know and when?](#) In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 820–842, Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Chengqiang Lu, Jianwei Zhang, Yunfei Chu, Zhengyu Chen, Jingren Zhou, Fei Wu, Haiqing Chen, and Hongxia Yang. 2022. [Knowledge distillation of transformer-based language models revisited](#).
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. [SQuAD: 100,000+ questions for machine comprehension of text](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. [Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter](#).
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. [Recursive deep models for semantic compositionality over a sentiment treebank](#). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.
- Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. [MobileBERT: a compact task-agnostic BERT for resource-limited devices](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.
- Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [Well-read students learn better: On the importance of pre-training compact models](#).
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. [GLUE: A multi-task benchmark and analysis platform for natural language understanding](#). In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.
- Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. [Minilm: deep self-attention distillation for task-agnostic compression of pre-trained transformers](#). In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS’20*, Red Hook, NY, USA. Curran Associates Inc.
- Xinpeng Wang, Leonie Weissweiler, Hinrich Schütze, and Barbara Plank. 2023. [How to distill your BERT: An empirical study on the impact of weight initialization and distillation objectives](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1843–1852, Toronto, Canada. Association for Computational Linguistics.
- Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. [Neural network acceptability judgments](#). *Transactions of the Association for Computational Linguistics*, 7:625–641.
- Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. [A broad-coverage challenge corpus for sentence understanding through inference](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. [Aligning books and movies: Towards story-like visual explanations by watching movies and reading books](#). In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.

## A Appendix

### A.1 Implementation Details

We use the code base from [Izsak et al. \(2021\)](#) for both pretraining and finetuning of No-KD models. In the case of KD models, we utilize the code base introduced by [Wang et al. \(2023\)](#), which itself builds upon the work of [Izsak et al. \(2021\)](#). Our code is available at [https://github.com/MinhDucBui/revisiting\\_distillation](https://github.com/MinhDucBui/revisiting_distillation).

### A.2 Pretraining Details

Our pretraining pipeline employs a batch size of 1024, employing gradient accumulation with a batch size of 256. We adopt a time-based learning rate schedule with a linear curve. The peak learning rate is set to  $5e-4$  for distillation strategies and  $1e-3$  for No-KD. We opt for a warmup proportion of 0.06 for both scenarios. Utilizing the AdamW optimizer with  $(\beta_1, \beta_2) = (0.9, 0.98)$  and  $\epsilon = 1e-6$ , we conduct training with mixed precision techniques.

We measure compute budget by wall-clock time. All experiments are conducted on NVIDIA A100. Training our 6-layer model for a single epoch requires around 4 hours of wall-clock training time, while the 12-layer model demands approximately 11 hours. Scaling up the 6-layer model to 27.9B tokens extends the training duration to about 24 hours. Fine-tuning on GLUE with a single A100 GPU, coupled with grid-hyperparameter search, consumes up to 50 hours for the 6-layer models and nearly 100 hours for the 12-layer variants.

### A.3 GLUE Details

We provide a brief overview of each dataset within GLUE. For additional information regarding each data split, evaluation metric and more, see [Wang et al. \(2018\)](#).

**CoLA** The Corpus of Linguistic Acceptability ([Warstadt et al., 2019](#)) comprises English acceptability judgments sourced from books and journal articles on linguistic theory.

**SST-2** The Stanford Sentiment Treebank ([Socher et al., 2013](#)) includes sentences extracted from movie reviews, along with human annotations of their binary sentiment.

**MRPC** The Microsoft Research Paraphrase Corpus ([Dolan and Brockett, 2005](#)) consists of sentence pairs automatically extracted from online news sources, with human annotations indicating whether the sentences are semantically equivalent.

**QQP** The Quora Question Pairs dataset is a compilation of question pairs from the community question-answering website. The objective is to determine whether a pair of questions are semantically equivalent.

**STS-B** The Semantic Textual Similarity Benchmark ([Cer et al., 2017](#)) contains sentence pairs with a human annotated similarity score ranging from 1 to 5.

**MNLI** The Multi-Genre Natural Language Inference Corpus ([Williams et al., 2018](#)) is a crowdsourced collection of sentence pairs with textual entailment annotations. The task involves predicting whether a premise sentence entails, contradicts, or is neutral with respect to a hypothesis.

**QNLI** The Stanford Question Answering Dataset ([Rajpurkar et al., 2016](#)) is a question-answering dataset comprising question-paragraph pairs, with the task of determining whether the context sentence contains the answer to the question.

**RTE** The Recognizing Textual Entailment (RTE) datasets originate from annual textual entailment challenges. The dataset is standardized to a two-class split, collapsing neutral and contradiction into "not entailment" for consistency.