Distilling BERT: Transfer Dataset Size and Composition Effects

Steven van de Graaf

July 26, 2021

University of Amsterdam

Introduction

Motivation

- "ImageNet moment" for *Natural Language Processing* (NLP) thanks to Transformers
- Transformers are getting ever bigger:

Paper	**Date**	**Name**	**Number of Parameters**
Radford et al. (2018)	06-2018	GPT	110 000 000
Devlin et al. (2018)	10-2018	$\text{BERT}_{\text{LARGE}}$	320 000 000
Radford et al. (2019)	02-2019	GPT-2	1 542 000 000
Shoeybi et al. (2019)	09-2019	Megatron	8 300 000 000
Raffel et al. (2019)	10-2019	T5-11B	11 000 000 000
Microsoft (2020)	02-2020	Turing-NLG	17 000 000 000
Brown et al. (2020)	05-2020	GPT-3	175 000 000 000
Fedus et al. (2021)	01-2021	Switch-C	1 571 000 000 000

Table 1: Sizes of recent Transformer(-based) architectures in terms of the number of parameters.

Motivation (cont.)

- Increasing size comes at significant costs in terms of *sustainability* and *accessibility*.
- Increasing importance of methods that make *Neural Networks* (NNs) more efficient (“Green AI”).
- *Knowledge Distillation* (KD) is one such promising method to “compress” NNs.

Knowledge Gap

- In the last two years or so, there have been various efforts to apply KD to Transformer-based architectures.
- The resulting "student networks" are both *smaller* and *faster*, while still achieving *comparable performance*.
- While promising, this new line of research is still immature, as reflected by the lack of established best-practices when it comes to the various aspects of KD.

Scope

- While the biggest NNs (containing hundreds of billions of parameters) stand to gain the most from being compressed, this research focuses its efforts solely on the popular $BERT_{BASE}$ network by Devlin et al. (2018).
- When it comes to what aspects of the KD process to investigate, this research has settled on the *transfer dataset* $\mathcal{D}_T$ used during pre-training.

Research Questions

RQ$_1$ What are the effects of the *size* of the transfer dataset $\mathcal{D}_T$ used during pre-training in the Knowledge Distillation process as applied to $\text{BERT}_{\text{BASE}}$, measured in performance on downstream performance task(s)?

RQ$_2$ What are the effects of the *composition* of the transfer dataset $\mathcal{D}_T$ used during pre-training in the Knowledge Distillation as applied to $\text{BERT}_{\text{BASE}}$, measured in performance on downstream performance task(s)?

Background

BERT$_{\text{BASE}}$

- Almost three years ago, Devlin et al. (2018) introduced their new language representation NN *Bidirectional Encoder Representations from Transformers* (BERT).
- BERT is a Transformer-based architecture that uses (only) the *Encoder* stack of the Transformer architecture (Vaswani et al., 2017).
- BERT was pre-trained using a *Masked Language Model* (MLM) objective: It doesn't take a *[MASK]* to figure this out.

Knowledge Distillation

- First introduced by Buciluǎ et al. (2006) and popularized by Hinton et al. (2015).
- A smaller (untrained) *student network* is trained to mimic a larger, pre-trained *teacher network* by means of *knowledge transfer*.

Soft target loss

The student network is trained on some *transfer dataset* $\mathcal{D}_T$ to minimize a loss function in which the target is the distribution of class probabilities as output by the teacher network (*soft target loss*):

$$\mathcal{L}_{\text{soft}} = \mathcal{L}\left(\boldsymbol{z}_s, \boldsymbol{z}_t\right) = KL\left(\sigma\left(\boldsymbol{z}_s\right) || \sigma\left(\boldsymbol{z}_t\right)\right). \tag{1}$$

Advantages:

- The data in $\mathcal{D}_T$ can be entirely unlabeled.
- Provides a richer signal during training when compared to only using the ground truth label as your target.

Hard target loss

When some or all of the data in $\mathcal{D}_T$ is labeled, Hinton et al. (2015) have found it beneficial to train the student network to also predict the ground truth labels $\boldsymbol{y}$ (*hard target loss*):

$$\mathcal{L}_{\text{hard}} = \mathcal{L}\left(\boldsymbol{z}_s, \boldsymbol{y}\right) = \mathcal{H}\left(\sigma\left(\boldsymbol{z}_s\right), \boldsymbol{y}\right). \tag{2}$$

Combined loss function

Combining the loss functions defined in Eqs. (1) and (2) yields the following combined loss function:

$$\begin{aligned}\mathcal{L}_{\text{combined}} &= \mathcal{L}\left(\boldsymbol{z}_s, \boldsymbol{z}_t, \boldsymbol{y}\right), \\ &= \alpha \cdot KL\left(\sigma\left(\boldsymbol{z}_s\right) || \sigma\left(\boldsymbol{z}_t\right)\right) \\ &\quad + \beta \cdot \mathcal{H}\left(\sigma\left(\boldsymbol{z}_s\right), \boldsymbol{y}\right).\end{aligned} \tag{3}$$

Related Work

Variables

KD as described by Hinton et al. (2015) provides only a general recipe that leaves much to be specified, such as:

- What NN architecture(s) do you use?
- What transfer dataset do you use?
- What composition of the combined loss function do you use?

Variables (cont.)

For Transformer(-based) architectures specifically, at least one more important variable can be added to the mix:

- At what stage (pre-training, fine-tuning or both) is KD applied?

The last variable to consider is more generally applicable to all scientific research:

- How do you evaluate your approach / method?

Summary (condensed)

Paper	Teacher network	Stage	$\mathcal{D}_T$	Evaluation
Tang et al. (2019)	$\text{BERT}_{\text{LARGE}}$	FT	×	Subset of GLUE
Liu et al. (2019)	MT-DNN	FT	×	GLUE
Yang et al. (2019)	$\text{BERT}_{\text{BASE}}$	FT	×	DeepQA
Sun et al. (2019)	$\text{BERT}_{\text{BASE}}$	FT	×	Subset of GLUE
Jiao et al. (2019)	$\text{BERT}_{\text{BASE}}$	Both	"large-scale text corpus"	GLUE
Turc et al. (2019)	$\text{BERT}_{\text{BASE}}$	FT	TBC + EnWiki	Subset of GLUE
Sanh et al. (2019)	$\text{BERT}_{\text{BASE}}$	PT	TBC + EnWiki	GLUE & SQuAD
Sun et al. (2020)	$\text{IB-BERT}_{\text{LARGE}}$	PT	TBC + EnWiki	GLUE & SQuAD

Table 2: Condensed summary of related works along the questions posed previously.

Experimental Design

NN Architecture

Following the examples of Sun et al. (2019); Sanh et al. (2019), we use *$BERT_{BASE}$* for our *teacher network* and a shallower version of it, using 6 layers instead of 12, for our *student network*, which we refer to as *$BERT_{STUDENT}$*.

Network	$\#P$	**Size-on-disk**	**Inference time**
$\text{BERT}_{\text{BASE}}$	109.5 M (1.0×)	417 MB (1.0×)	1.47 s (1.0×)
$\text{BERT}_{\text{STUDENT}}$	66.4 M (1.65×)	253 MB (1.65×)	0.80 s (1.83×)

Table 3: Comparison of network size (in $\#P$ and size-on-disk) and inference time (in s) between our teacher network, $\text{BERT}_{\text{BASE}}$, and our student network, $\text{BERT}_{\text{STUDENT}}$.

Pre-training

- Like Jiao et al. (2019); Sanh et al. (2019); Sun et al. (2020), we apply KD during pre-training.
- For our composite loss function, we combine the hard target loss and soft target loss seen previously, with a cosine embedding loss that leverages the knowledge encoded in intermediate layers:

$$\begin{aligned}\mathcal{L} &= \alpha_{\text{soft}} \cdot \mathcal{L}_{\text{soft}} + \alpha_{\text{hard}} \cdot \mathcal{L}_{\text{hard}} + \alpha_{\cos} \cdot \mathcal{L}_{\cos}, \\ &= \alpha_{\text{soft}} KL\left(\sigma\left(\boldsymbol{z}_s\right) || \sigma\left(\boldsymbol{z}_t\right)\right) \\ &\quad + \alpha_{\text{hard}} \cdot \mathcal{H}\left(\sigma\left(\boldsymbol{z}_s\right), \boldsymbol{y}\right) \\ &\quad + \alpha_{\cos} \cdot \left(1 - \cos\left(\boldsymbol{h}_s, \boldsymbol{h}_t\right)\right).\end{aligned} \tag{4}$$

Data

Following the pre-training procedures of both $\mathrm{BERT}_{\mathrm{BASE}}$ (Devlin et al., 2018) and DistilBERT (Sanh et al., 2019), we use a concatenation of (a replica of) the *Toronto BookCorpus* (TBC) dataset (Zhu et al., 2015) and *English Wikipedia* for our composite pre-training corpus.

Toronto BookCorpus dataset

- The TBC dataset (Zhu et al., 2015) is no longer publicly available.
- Smashwords still exists, however.
- Following the procedure of Van de Graaf (2019b), we have developed and released *Replicate Toronto BookCorpus*, a collection of scripts that achieve the following 3 steps:
    1. Finding those books that meet our criteria.
    2. Downloading these books.
    3. Pre-processing these books.

Toronto BookCorpus dataset (cont.)

Following these steps, we were able to create a faithful replica of the TBC dataset:

Dataset	**Size**	**Nº sentences**	**Nº words**	**mean w.p.s.**	**median w.p.s.**
TBC (Zhu et al., 2015)	4.7 GB	74.0 M	984.8 M	13.3	11
TBC replica	5.1 GB	65.4 M	897.1 M	13.7	11

Table 4: Comparison of datasets in terms of size and other important statistics. "w.p.s." stands for "words per sentence".

English Wikipedia

- While used at least equally frequently in *Language Model* (LM) pre-training, English Wikipedia also requires much pre-processing.
- For this, we follow the procedure of Van de Graaf (2019a):
  1. Download the latest "dump" of all pages and articles of English Wikipedia.
  2. Extract only the text passages (ignoring any lists, tables and headers).
  3. Tokenize the sentences.
  4. Concatenate all pages and articles into a single text-file.

- What we refer to as "our corpus" or $\mathcal{D}_{\text{Corpus}}$ is now a concatenation of our TBC replica and English Wikipedia.
- $\mathcal{D}_{\text{Corpus}}$ consists of 109.3 M *sequences* or 4.0 B *tokens* in total.

Baselines

In order to verify the significance of our results, we compare the performance of $\text{BERT}_{\text{STUDENT}}$ to two baselines:

1. $\textbf{BERT}_{\textbf{BASE}}$ (Devlin et al., 2018)
2. **DistilBERT** (Sanh et al., 2019)

Results

Methods

- We take three random samples (without replacement) from our corpus $\mathcal{D}_{\text{Corpus}}$ using three different sample sizes in order to obtain three differently sized transfer datasets $\mathcal{D}_T^N$.
- The sample sizes used for the random sampling from our corpus are proportional to its size (109.3 M sequences or 4.0 B tokens) by different orders of magnitude:
    1. We take a first random sample that is 10% of its size (10.9 M sequences or 402.6 M tokens)
    2. We take a second random sample that is 1% of its size (1.1 M sequences or 40.3 M tokens)
    3. We take a third and final random sample that is 0.1% of its size (109.4 k sequences or 4.0 M tokens)

Methods (cont.)

Dataset	**Size**	**Nº sequences**	**Nº tokens**
$\mathcal{D}_{\text{Corpus}}$	20.0 GB	109.3 M	4.0 B
$\mathcal{D}_{T}^{10\%}$	2.0 GB	10.9 M	402.6 M
$\mathcal{D}_{T}^{1\%}$	204.8 MB	1.1 M	40.3 M
$\mathcal{D}_{T}^{0.1\%}$	20.6 MB	109.4 k	4.0 M

Table 5: Comparison between our corpus and our transfer datasets (which are differently sized random samples of our corpus).

Results - GLUE

Network	Transfer dataset	Score
Ours:		
	$\mathcal{D}_T^{10\%}$	**72.5**
$\text{BERT}_{\text{STUDENT}}$	$\mathcal{D}_T^{1\%}$	**72.4**
	$\mathcal{D}_T^{0.1\%}$	**70.0**
Baselines:		
$\text{BERT}_{\text{BASE}}$ (Devlin et al., 2018)		**74.9**
DistilBERT (Sanh et al., 2019)		**74.3**

Table 6: Comparison of performance on the GLUE benchmark using transfer datasets of various sizes. Results of $\text{BERT}_{\text{BASE}}$ and DistilBERT as reported by Sanh et al. (2019).

Results - SQuAD

Network	Transfer dataset	Score EM	Score **F1**
Ours:			
	$\mathcal{D}_T^{10\%}$	56.9	**68.5**
BERT$_{\text{STUDENT}}$	$\mathcal{D}_T^{1\%}$	54.7	**66.4**
	$\mathcal{D}_T^{0.1\%}$	45.0	**56.5**
Baselines:			
BERT$_{\text{BASE}}$ (Devlin et al., 2018)		73.9	**82.3**
DistilBERT (Sanh et al., 2019)		69.6	**78.7**

Table 7: Comparison of performance on SQuAD using transfer datasets of various sizes. Results of BERT$_{\text{BASE}}$ and DistilBERT as reported by Sanh et al. (2019).

Results

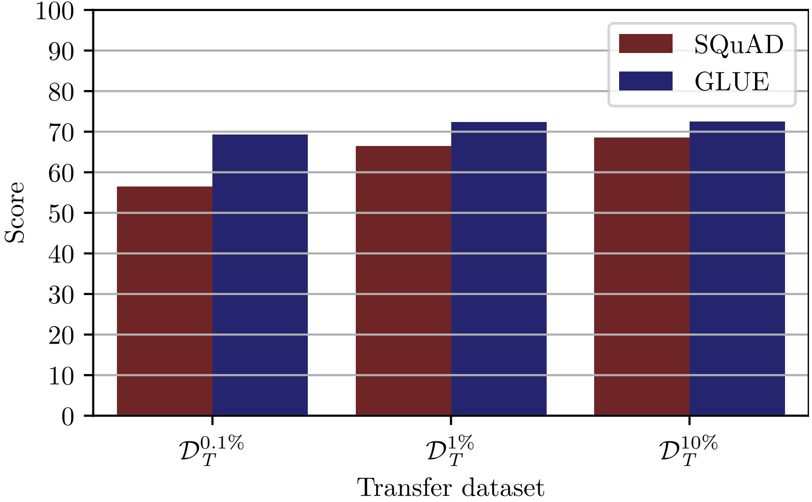

Figure 1: Plot of performance of $\text{BERT}_{\text{STUDENT}}$ on GLUE and SQuAD using transfer datasets of various sizes.

Methods

We create two new transfer datasets $\mathcal{D}_T$ of different compositions by adding "noise" to $\mathcal{D}_T^{10\%}$:

1. The first new transfer dataset we create is an randomization of $\mathcal{D}_T^{10\%}$, wherein each sequence of tokens in $\mathcal{D}_T^{10\%}$ is randomized (i.e. shuffled). This dataset is labeled as $\mathcal{D}_T^{\text{Rand.}}$.
2. The second new transfer dataset we create consists of entirely generated data (labeled as $\mathcal{D}_T^{\text{Gen.}}$). For this, we first compute two statistics of $\mathcal{D}_T^{10\%}$, which are then used to generate completely new sequences.

Methods (cont.)

Transfer dataset	**Size**	**Nº sequences**	**Nº tokens**
$\mathcal{D}_T^{10\%}$	2.0 GB	10.9 M	402.6 M
$\mathcal{D}_T^{\text{Rand.}}$	2.0 GB	10.9 M	402.6 M
$\mathcal{D}_T^{\text{Gen.}}$	2.0 GB	10.9 M	402.6 M

Table 8: Comparison of the transfer datasets used in the current experiment (which are of different compositions).

Results - GLUE

Transfer dataset	**Score**
$\mathcal{D}_T^{10\%}$	**72.5**
$\mathcal{D}_T^{\text{Rand.}}$	**64.8**
$\mathcal{D}_T^{\text{Gen.}}$	**59.2**

Table 9: Comparison of performance on the GLUE benchmark using transfer datasets of various compositions.

Results - SQuAD

Transfer dataset	**Score**	
	EM	**F1**
$\mathcal{D}_T^{10\%}$	56.9	**68.5**
$\mathcal{D}_T^{\text{Rand.}}$	8.7	**16.2**
$\mathcal{D}_T^{\text{Gen.}}$	27.0	**37.3**

Table 10: Comparison of performance on SQuAD using transfer datasets of various compositions.

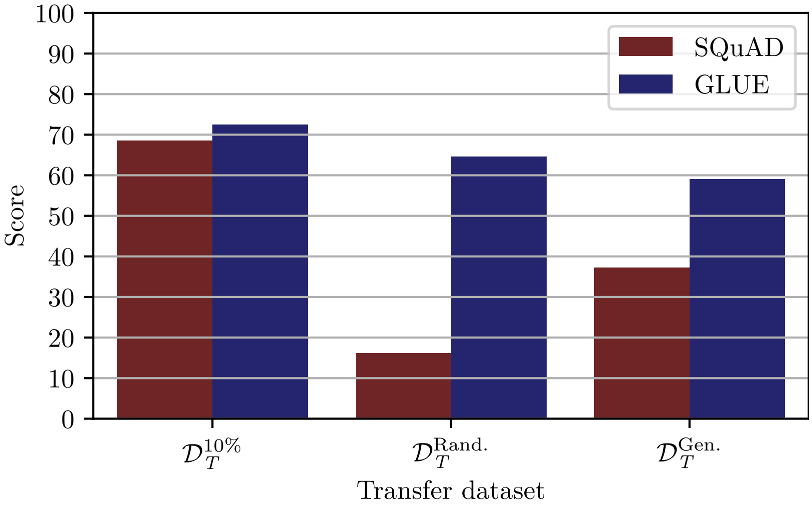

Figure 2: Plot of performance of $\text{BERT}_{\text{STUDENT}}$ on GLUE and SQuAD using transfer datasets of various compositions.

Conclusion

Summary of results

- When it comes to the *size* of the transfer dataset $\mathcal{D}_T$, we observed competitive performance on GLUE and acceptable performance on SQuAD when a transfer dataset is used of "only" $\sim 100\,\mathrm{k}$ sequences or $\sim 4\,\mathrm{M}$ tokens. Whereas performance on GLUE does not seem to benefit much from using a larger transfer dataset, performance on SQuAD does.
- With respect to the *composition* of the transfer dataset $\mathcal{D}_T$, we observed diminishing (though still acceptable) performance on GLUE and poor performance on SQuAD when a transfer dataset is used that does not consist of proper *Natural Language* (NL) (i.e. randomized and/or generated) data. More surprisingly, we observe diminishing performance on GLUE when a less "informative" transfer dataset is used for KD, whereas for SQuAD, we observe an increase in performance.

Answer to Research Question 1

- The general effect on downstream task performance of using a smaller transfer dataset $\mathcal{D}_T$ to distill the knowledge of $\text{BERT}_{\text{BASE}}$ into $\text{BERT}_{\text{STUDENT}}$ is a *slight decrease* in performance.
- We consider the decrease in performance on GLUE to be marginal, especially if you also weigh the costs (both financially and environmentally, as well as temporally).
- Performance on SQuAD benefits significantly more from using a (relatively) larger transfer dataset $\mathcal{D}_T$ during pre-training in the KD process.

Answer to Research Question 2

- The general effect on downstream task performance of using a transfer dataset $\mathcal{D}_T$ composed of non NL data to distill the knowledge of $\text{BERT}_{\text{BASE}}$ into $\text{BERT}_{\text{STUDENT}}$ is a *significant decrease* in performance.
- This is the case for GLUE (though performance is still acceptable), but even more so for SQuAD. This result makes intuitive sense, as you would expect that (pre-)training with less informative data would yield a less informed NN.
- Nonetheless, this answer is by no means conclusive, as there are still aspects unexplored, which we leave for future work. Moreover, for low-resource languages, randomizing and/or generating sequences might still prove worthwhile.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Buciluǎ, C., Caruana, R., and Niculescu-Mizil, A. (2006). Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fedus, W., Zoph, B., and Shazeer, N. (2021). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*.

van de Graaf, S. (2019a). Pre-processing a wikipedia dump for nlp model training — a write-up. https://towardsdatascience.com/pre-processing-a-wikipedia-dump-for-nlp-model-training-a-write-up-3b9176fdf67. Accessed on 13-12-2019.

van de Graaf, S. (2019b). Replicating the toronto bookcorpus dataset — a write-up. https://towardsdatascience.com/replicating-the-toronto-bookcorpus-dataset-a-write-up-44ea7b87d091. Accessed on 06-12-2019.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. (2019). Tinybert: Distilling bert for natural language understanding.

Liu, X., He, P., Chen, W., and Gao, J. (2019). Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv preprint arXiv:1904.09482*.

Microsoft (2020). Turing-nlg: A 17-billion-parameter language model by microsoft. https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/. Accessed on 09-06-2020.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Sun, S., Cheng, Y., Gan, Z., and Liu, J. (2019). Patient knowledge distillation for bert model compression. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., and Zhou, D. (2020). Mobilebert: a compact task-agnostic bert for resource-limited devices.

Tang, R., Lu, Y., Liu, L., Mou, L., Vechtomova, O., and Lin, J. (2019). Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Turc, I., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Well-read students learn better: The impact of student initialization on knowledge distillation. *arXiv preprint arXiv:1908.08962*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yang, Z., Shou, L., Gong, M., Lin, W., and Jiang, D. (2019). Model compression with multi-task knowledge distillation for web-scale question answering system.

References iv

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.