

Structured Pruning of a BERT-based Question Answering Model

J.S. McCarley and Rishav Chakravarti and Avirup Sil

IBM Research AI

Yorktown Heights, NY

{jsmc, rchakravarti, avi}@us.ibm.com

Abstract

The recent trend in industry-setting Natural Language Processing (NLP) research has been to operate large scale pretrained language models like BERT under strict computational limits. While most model compression work has focused on “distilling” a general-purpose language representation using expensive *pretraining distillation*, much less attention has been paid to creating smaller task-specific language representations which, arguably, are more useful in an industry setting. In this paper, we investigate compressing BERT- and RoBERTa-based question answering systems by structured pruning of parameters from the underlying trained transformer model. We find that an inexpensive combination of *task-specific structured pruning* and *task-specific distillation*, without the expense of *pretraining distillation*, yields highly-performing models across a range of speed/accuracy tradeoff operating points. We start from full-size models trained for SQuAD 2.0 or Natural Questions and introduce gates that allow selected parts of transformers to be individually eliminated. Specifically, we investigate (1) structured pruning to reduce the number of parameters in each transformer layer, (2) applicability to both BERT- and RoBERTa-based models, (3) applicability to both SQuAD 2.0 and Natural Questions, and (4) combining structured pruning with distillation. We find that pruning a combination of attention heads and the feed-forward layer yields a near-doubling of inference speed on SQuAD 2.0, with less than a 1.5 F1-point loss in accuracy. Furthermore, we find that a combination of distillation and structured pruning almost doubles the inference speed of RoBERTa-large based model for Natural Questions, while losing less than 0.5 F1-point on short answers.

1 Introduction

The recent surge in NLP model complexity has outstripped Moore’s law: ELMO: 93×10^6 (Peters et al., 2018), BERT: 340×10^6 (Devlin et al., 2019) and T5: 11000×10^6 parameters (Raffel et al., 2019). Deeply stacked layers of transformers (including BERT, RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2019b), ALBERT (Lan et al., 2019), and T5 (Raffel et al., 2019)) have greatly improved state-of-the-art accuracies across a variety of NLP tasks, but the computational intensity raises concerns in the cloud-computing economy. Numerous techniques developed to shrink neural networks including distillation (Distilbert, TinyBert) (Sanh et al., 2019; Jiao et al., 2019), quantization (Shen et al., 2019), and pruning (Michel et al., 2019) are now being applied to transformers.

Among Natural Language Processing (NLP) tasks, question answering (QA), in particular, has immediate applications in real-time systems. A relatively new field in the open domain question answering (QA) community is machine reading comprehension (MRC) which aims to read and comprehend a given text, and then answer questions based on it. MRC is one of the key steps for natural language understanding (NLU). MRC also has wide applications in the domain of conversational agents and customer service support. Transformer-based models have led to striking gains in accuracy on MRC tasks recently, as measured on the SQuAD v1.1 (Rajpurkar et al., 2016) and SQuAD v2.0 (Rajpurkar et al., 2018) leaderboards. We briefly mention three MRC tasks: SQuAD v1.1 consists of reference passages from Wikipedia with answers and questions constructed by annotators after viewing the passage. SQuAD v2.0 augmented the SQuAD v1.1 collection with additional questions that did not have answers in the

model	params	SQuAD 1.1	SQuAD 2.0	NQ
BERT-large	356M	90.9 (c)	83.52	56.14
BERT-base	125M	88.4 (a) 88.5 (b)	76.4 (a)	52.75
DistilBert	63M	86.2 (a) 86.9 (b)	69.5 (a)	50.46
TinyBert	63M	87.5 (a)	73.4 (a)	44.64

Table 1: Comparison of published F1 scores of well-known distillation’s of BERT on several question-answering tasks. Though not strictly comparable, we observe that on SQuAD 1.1 smaller models approach BERT-large in accuracy, whereas the smaller models underperform notably on both SQuAD 2.0 and Natural Questions (NQ). We show the short answer F1 for NQ. Sources: (a)=(Jiao et al., 2019), (b)=(Sanh et al., 2019), (c)=(Devlin et al., 2019).

reference passage. Natural Questions (NQ) (Kwiatkowski et al., 2019) removed the observational bias by starting from questions submitted to Google and providing annotated answers from appropriate passages.

MRC seems to be a particularly difficult task to speed up. While distillation papers have advertised impressive speedups with near-negligible loss in accuracy on GLUE benchmarks, published applications of distillation to MRC have been less impressive (often relegated to the appendix.) In Table 1, we compare the accuracies (F1 score) of Distilbert and TinyBert with baseline BERT-large and BERT-base models on three MRC tasks, using the number of parameters as a crude proxy for speed. Compared to BERT-large, models with fewer parameters achieved modest losses on SQuAD 1.1. The shortfalls on the more challenging SQuAD 2.0 were much larger. We also found that the shortfalls of smaller models were large on NQ. SQuAD is also seen as a worst-case performance loss for speed up techniques based on quantization, (Shen et al., 2019) while the difficulty of distilling a SQuAD model (compared to sentence-level GLUE tasks) is acknowledged in (Jiao et al., 2019). We speculate that these difficulties are because answer selection requires token level rather than passage level annotation, and requires long range attention between query and passage.

The contributions of this paper are (1) application of structured pruning techniques to the feed-forward layer and the hidden dimension of the transformers, not just the attention heads, (2) the combination of distillation and pruning, (3) thereby significantly pruning the question answering system with minimal loss of accuracy and considerable speedup, all without the expense of revisiting pretraining

Furthermore we survey multiple pruning techniques (both heuristic and trainable) and providing recommendations specific to transformer-based question answering models. During the course of the investigation, we also learn that an optimal pruning learns a structure consisting of *non-identical* transformers, namely lightweight transformers near the top and bottom while retaining more complexity in the intermediate layers, instead of the typically 12-24 layers of *identically* sized transformers, common in widely distributed pre-trained models

2 Related work

While distillation (student-teacher) of BERT has produced notably smaller and faster models (Tang et al., 2019; Turc et al., 2019; Tsai et al., 2019; Yang et al., 2019a), the focus has been on sentence level annotation tasks (e.g. GLUE) that do not require long-range attention links between query and passage. Application of distillation to MRC has been much more limited: DistilBERT (Sanh et al., 2019) used pretraining distillation to obtain 60% speedups on GLUE tasks while retaining 97% of the accuracy. MRC results, after additional task-specific distillation, included a modest speedup and small performance loss on SQuAD 1.1. TinyBERT (Jiao et al., 2019) used both pretraining and task-specific distillation to obtain $9.4\times$ speedups on GLUE. However, they restricted SQuAD evaluation to using BERT-base as a teacher, and deferred deeper investigation to future work. The authors are not aware of any results from distilled models on NQ.

Investigations into pruning BERT have also omitted MRC. Michel et al. (2019) applied simple gating heuristics to prune BERT attention heads and achieve speedups on MT and MNLI. Voita et al. (?) introduced L_0 regularization to BERT while focusing on linguistic interpretability of attention heads but did

not report speedups. L_0 regularization was combined with matrix factorization to prune transformers in (Wang et al., 2019). Gale et al. (2019) induced unstructured sparsity on a transformer-based MT model, but did not report speedups. Kovaleva et al. (2019) also focused on interpreting attention, and achieved small accuracy gains on GLUE tasks by disabling (but not pruning) certain attention heads. Structured pruning as a form of dropout is explored in (?). They prune entire layers of BERT, but suggest that smaller structures could also be pruned. They evaluate on MT, language modeling, and generation-like tasks, but not SQuAD.

Other approaches to speeding up transformers include AIBERT (Lan et al., 2019), which shared parameters across layers in order to accelerate training, but did not report timings of inference.

PoWER-BERT (Goyal et al., 2020) eliminated intermediate word vectors from upper transformer layers without loss of accuracy on classifier tasks, but it is not clear that this technique is applicable to MRC tasks that require labeling of word positions.

QBERT (Shen et al., 2019) and Q8BERT (Zafrir et al., 2019) aggressively quantized floating point calculations to ultra-low precision in order to compress BERT. They noted that SQuAD was harder to quantize (greater performance drop) than other tasks.

Finally, (Li et al., 2020) investigated both unstructured pruning and quantization of RoBERTa as a function of model size, and found that both pruning and quantization were complementary, an important reminder that multiple types of compression are not mutually exclusive.

3 Pruning transformers

3.1 Gate placement

Our approach to pruning each aspect of a transformer is similar. We insert three *masks* into each transformer. Each mask is a vector of gate variables $\gamma_i \in [0, 1]$, pointwise multiplied into a slice of transformer parameters, where $\gamma_i = 0$ indicates a slice to be pruned, and $\gamma_i = 1$ indicates a slice to remain active. We describe the placement of each mask with the terminology of (Vaswani et al., 2017), indicating relevant sections of that paper.

In each self-attention sublayer, we place a mask, Γ^{attn} of size n_H which selects attention heads to remain active. (section 3.2.2)

In each feed-forward sublayer, we place a mask, Γ^{ff} of size d_I which selects ReLU/GeLU activations to remain active. (section 3.3)

The final mask, Γ^{emb} , of size d_E , selects which embedding dimensions, (section 3.4) remain active. This gate is applied *identically* to both input and residual connections in each transformer layer.

Here n_H is the number of heads per transformer layer (12 or 16), d_E is the size of the embeddings (768 or 1024) as well as the inner hidden dimension, and d_I is the size of the intermediate activations in the feed-forward part of the transformer (3072 or 4096.) Sizes are for (BERT-base, BERT-large).

3.2 Determining Gate Values

We investigate four approaches to determining the gate values.

(1) Random: each γ_i is sampled from a Bernoulli distribution of parameter p , where p is manually adjusted to control the sparsity. This method is the naive baseline, and is expected to be worse than other methods.

(2) Gain: We follow the method of (Michel et al., 2019) and estimate the influence of each gate γ_i on the training set likelihood \mathcal{L} by treating each γ_i as a continuous parameter and computing the mean value of

$$g_i = \left| \frac{\partial \mathcal{L}}{\partial \gamma_i} \right|_{\gamma_i=0} \quad (1)$$

(“head importance score”) during one pass over the training data. We threshold g_i to determine which transformer slices to retain.

(3) Leave-one-out: We again follow the method of (Michel et al., 2019) and evaluate the impact on the devset $F1$ score of a system with exactly one gate set to zero. We control the sparsity during decoding by

retaining those gates for which $\delta F1$ is largest. Note that since this procedure requires one pass through the data per gate, it is only practical for masks with relatively few gates (i.e. the attention heads Γ^{attn} .)

(4) L_0 regularization: Following the method described in (?), during training time the gate variables γ_i

$$\gamma_i \sim \text{hc}(\alpha_i) \quad (2)$$

are sampled from a hard-concrete distribution $\text{hc}(\alpha_i)$ (Maddison et al., 2017) parameterized by a corresponding variable $\alpha_i \in \mathbb{R}$. The task-specific objective function is penalized in proportion to the expected number instances of $\gamma = 1$. The proportionality constants λ^{attn} in the penalty terms, e.g.

$$-\lambda^{\text{attn}} \mathbb{E} \left[\sum_i \delta_{\gamma_i, 1} \right] \quad (3)$$

(and similarly for λ^{ff} and λ^{emb} .) are used to manually adjust the sparsity. The expectation is over the same hard-concrete distribution from which we sample. We resample the γ_i with each minibatch. The full objective function is differentiable with respect to the α_i because of the reparameterization trick. (Kingma and Welling, 2014; Rezende et al., 2014) The α_i are updated by backpropagation for up to one training epoch on the task training data, with all other parameters held fixed. The final values for the gates γ_i are obtained by thresholding the α_i . We note that either the log-likelihood or a distillation-based objective can be penalized as in Eq. (3).

3.3 Structured Pruning

After the values of the γ_i have been determined by one of the above methods described above, we prune the model. Attention heads corresponding to $\gamma_i^{\text{attn}} = 0$ are removed. Slices of both feed forward linear transformations corresponding to $\gamma_i^{\text{ff}} = 0$ are removed. Slices of the embedding matrix corresponding to $\gamma_i^{\text{emb}} = 0$ are removed. The pruned model no longer needs masks, and now consists of transformers of varying, *non-identical* sizes. For experiments on some hardware, matrices are forced to have sizes that are round numbers rather than strictly respecting the threshold.

3.4 Continued training

We note that task-specific training of all transformer parameters may be continued further with the pruned model. In particular, in some experiments we continue training by incorporating distillation: the unpruned model is the teacher, and the pruned model is the student.

4 Experiments

4.1 Overall Setup and outline

We evaluate our proposed method on two benchmark QA datasets: SQuAD 2.0 (Rajpurkar et al., 2018) and Natural Questions (NQ) (Kwiatkowski et al., 2019). SQuAD 2.0 is a dataset of QA examples of questions with answers designed and answered by human annotators about Wikipedia passages. NQ is a dataset of Google search queries with answers from Wikipedia pages provided by human annotators. Out of the two, NQ is more natural, as the questions were asked by humans on Google without having seen the passage. On the other hand, SQuAD annotators read the Wikipedia passage first and then formulated the questions.

We address several empirical questions here: 1. Do BERT-base experiments transfer to BERT-large? 2. Do the proposed techniques transfer across datasets? 3. Does incorporating a distillation objective farther improve our models performance?

To answer these we tune our hyper-parameters on a subset of SQuAD 2.0 using a BERT-base model, and then test them on the full SQuAD 2.0 with a BERT-large model. Further, we show that the same techniques are applicable on the NQ dataset not just with BERT but also with RoBERTa. Finally we show that incorporating distillation achieves even stronger and more flexible results. When practical we report numbers as an average of 5 seeds.

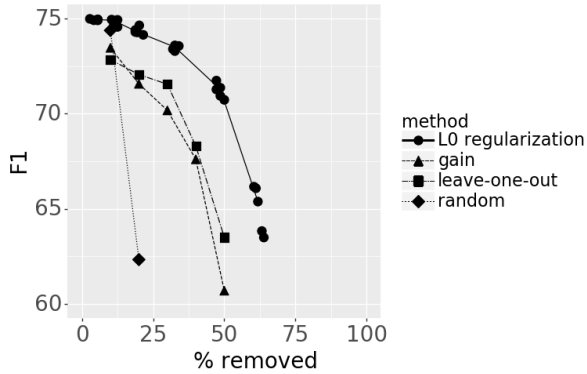


Figure 1: F1 vs percentage of attention heads pruned

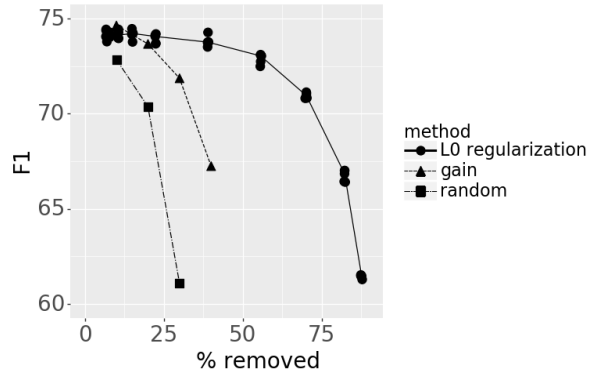


Figure 2: F1 vs percentage of feed-forward activations pruned

4.2 SQuAD 2.0

4.2.1 Experimental Setup and hyper-parameters

For careful selection of hyper-parameters (learning rate and penalty weight exploration) and in order to minimize overuse of the official dev-set, we use 90% of the official SQuAD 2.0 training data for training gates, and report results on the remaining 10%. This resulting model (*base-qa*) is initialized from a *bert-base-uncased* SQuAD 2.0 system trained on the 90% with a baseline performance of $F1 = 75.0$ on the 10% dataset. Experiments described were implemented using code from the HuggingFace repository (Wolf et al., 2019) and incorporated either *bert-base-uncased* or *bert-large-uncased* with a standard task-specific head.

Our final SQuAD 2.0 model (*large-qa*) use the standard training/dev split of SQuAD 2.0 and is initialized from a *bert-large-uncased* system (Glass et al., 2019) that achieves an $F1 = 84.6$ on the official dev set.

The gate parameters of the L_0 regularization experiments are trained for one epoch starting from the models above, with all transformer and embedding parameters fixed. The cost of training the gate parameters is comparable to extending fine tuning for an additional epoch. We investigated learning rates of 10^{-3} , 10^{-2} , and 10^{-1} on *base-qa*, and chose 10^{-1} for presentation and results on *large-qa*. This is notably larger than typical learning rates to tune BERT parameters. We used a minibatch size of 24 and otherwise default hyperparameters of the BERT-Adam optimizer. We used identical parameters for our *large-qa* experiments, except with gradient accumulation of 3 steps.

4.2.2 Accuracy as function of pruning

In figure 1 we plot the *base-qa* F1 as a function of the percentage of heads removed. The performance of ‘random’ decays abruptly. ‘Leave-one-out’ and ‘Gain’ are better, but substantially similar. L_0 regularization is best, allowing 48% pruning at a cost under 5 F1-points.

Also in figure 2 we plot the (accuracy) F1 of removing feed-forward activations. We see broadly similar trends as above, except that the performance is robust to even larger pruning. Leave-one-out requires a prohibitive number of passes ($n_L \times d_I$) through the data and was hence omitted. As before L_0 regularization is best, allowing 70% pruning at cost under 5 F1-points.

We found that when pruning embedding dimensions with Gain, the accuracy falls much more rapidly than when pruning attention heads or feed forward activations. Attempts to train L_0 regularization were unsuccessful - we speculate that the strong cross-layer coupling may necessitate different hyperparameters. For this reason we did not pursue pruning embedding dimensions as part of our model combination.

4.2.3 Validating these results

On the basis of the development experiments, we select an operating point, namely the largest values of λ^{attn} and λ^{ff} with < 5 F1-point loss. We train the feed-forward and attention gates of *large-qa* with

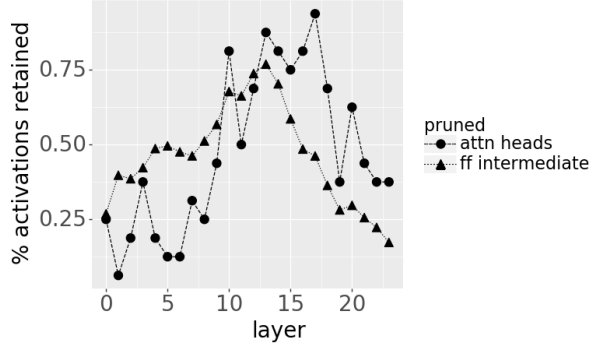


Figure 3: Percentage of attention heads and feed forward activations remaining after pruning, by layer

model	time (sec)	f_1	% attn removed	% ff removed	size (MiB)	f_1 no retrain
no pruning	2712	84.6	0	0	1279	
attn ₁	2288	84.2	44.3	0	1112	
ff ₁	2103	83.2	0	48.1	908	
ff ₁ + attn ₁	1718	82.6	44.0	48.1	740	
ff ₁ + attn ₁ + retrain	1667	83.7	44.0	48.1	740	82.6
ff($\times 2$) + attn($\times 2$) + retrain	1391	83.2	53.1	64.9	576	80.9
ff($\times 3$) + attn($\times 3$) + retrain	1213	82.4	57.6	73.7	492	76.8
ff($\times 4$) + attn($\times 4$) + retrain	1128	81.5	60.1	78.4	441	67.8

Table 2: Decoding times, accuracies, and space savings achieved at sample operating points on *large-qa*

these penalties, as well as multiples $2\times$, $3\times$, and $4\times$. The decoding times, accuracies, and model sizes are summarized in Table 2. Accuracies are medians of 5 seeds, and timings are medians of 5 decoding runs with the median seed, on a single Nvidia K80 with batch size 1. Models in which both attention and feed-forward components are pruned were built from the *independently trained* gate configurations of attention and feed forward. For corresponding penalty weights, the *large-qa* was pruned somewhat less than *base-qa*, and the $F1$ loss due to pruning was smaller.

Much of the performance loss can be recovered by continuing the training for an additional epoch after the pruning, even though the accuracy without retraining (final column) decreases substantially as more is pruned. We find the operating point $\text{ff}(\times 2) + \text{attn}(\times 2) + \text{retrain}$ nearly doubles the decoding speed, with a loss of less than 1.5 $F1$ points.

The speedup in decoding due to pruning the model is not simply proportional to the amount pruned. Both the attention and feed-forward part of each transformer have computations that remain unpruned, for example layer normalization.

4.2.4 Impact of pruning each layer

In Fig. 3 we show the percentage of attention heads and feed forward activations remaining after pruning, by layer. We see that intermediate layers retained more, while layers close to the embedding and close to the answer were pruned more heavily.

4.3 Natural Questions

We address three questions in this section:

- (1) Are the pruning techniques developed for the SQuAD 2.0 task also applicable to the NQ task?
- (2) Do pruning techniques developed for BERT also apply to RoBERTa?
- (3) Can we combine distillation and pruning to achieve even smaller, faster models?

4.3.1 Transfer of gates

We take the pruned BERT-large models described above and use the same model topologies and parameters as the initialization for continued training (using the cross-entropy objective function) of an NQ

model	% attn removed	%ff removed	LA (F1)	SA (F1)
BERT-large	0	0	66.1	54.7
pruned	44	48	65.9	51.7
pruned x2	53	65	64.2	49.6

Table 3: pruned on SQuAD, continued training on NQ

model	% attn removed	% ff removed	LA (F1)	SA (F1)
RoBERTa-large	0	0	70.3	58.8
pruned	42	40	68.3	57.7
pruned x2	53	56	67.8	55.5
pruned x3	68	75	65.2	52.2

Table 4: pruning RoBERTa on NQ, continued cross-entropy training on NQ

model. In other words, the gate variables are trained on SQuAD 2.0, and the only use of the NQ training data is in the continued training of the remaining transformer parameters. The results are shown in Table 3, while far from optimal, are encouraging. They suggest that the redundancies in BERT that are removed by pruning are not tied so specifically to a given task or domain of data that a pruned becomes brittle and inapplicable to another domain.

4.3.2 RoBERTa

RoBERTa-based models have achieved notably higher accuracy than BERT-based models across a variety of tasks, including MRC. For example, on NQ short answers, our RoBERTa-large model achieves 58.8 - over 4 F1-points better than the comparable BERT-large model, which achieved 54.7. RoBERTa has the same topology as BERT. It differs slightly in such aspects as tokenization, training data (during pretraining) and training procedure. The nature of these differences suggests that the pruning techniques developed for BERT should continue to work largely unchanged with RoBERTa. However, as noted by (Liu et al., 2019), BERT is significantly undertrained, which raises the concern that RoBERTa might achieve its better performance by more effectively utilizing the transformer parameters that were under-utilized and prunable in BERT.

We pruned this RoBERTa-large NQ model, using the same techniques as described above, selecting the gate values by L0 regularization for one epoch on approximately 20% of the NQ training data, and continued training for epoch on the full NQ training set. In Table 4 we show the accuracy and the amount pruned. For the same values of λ^{attn} and λ^{ff} , the amount pruned is similar to BERT. The tradeoff loss in accuracy also shows a similar falloff.

4.3.3 Combining distillation and pruning

Distillation may be combined with structured pruning in two distinct ways. First, in the final phase of continued training (after the model has been pruned) the cross-entropy objective function may be replaced by a distillation objective function in which the unpruned model is the teacher and the pruned model is the student. Alternately, the pruning phase itself may be driven by the distillation objective rather than the cross-entropy objective. In Table 5, we show results only using distillation in the continued training phase. The 2.9 F1-point gain of pruned $\times 2$ is especially notable. Timings are median of 5 decoding runs over the entire NQ development set on an NVidia V100 using 16-bit floating point with batch size 64. In this experiment, matrices were forced to have sizes that are round numbers, resulting in small changes ($< 1\%$) in reported pruning fractions. We also include for comparison RoBERTa-base model than has been similarly distilled, using RoBERTa-large as a teacher.

When the pruning phase itself is driven by the distillation objective, the results are not directly comparable because the same values of λ^{attn} and λ^{ff} yield significantly less pruning for the distillation

model	% attn removed	% ff removed	LA (F1)	SA (F1)	time sec.
RoBERTa-large	0	0	70.3	58.8	2789
pruned	42	40	69.8	58.4	1867
pruned x2	53	55	69.3	58.4	1523
pruned x3	68	75	67.6	55.4	1135
RoBERTa-base	NA	NA	67.3	55.9	1151

Table 5: pruning RoBERTa on NQ, continued distillation training on NQ

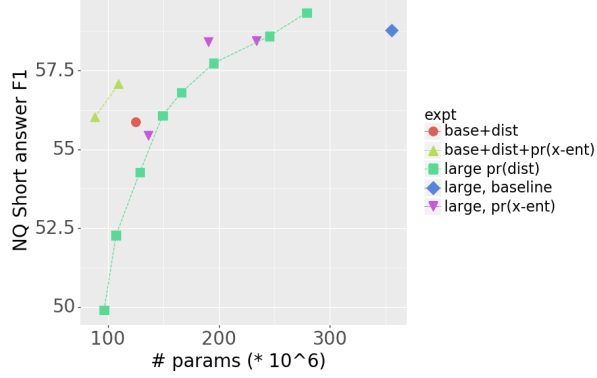


Figure 4: Short answer accuracy vs number of parameters (millions)

objective than for the cross-entropy objective. In Fig. 4 we plot the performance of various pruned models as a function of the number of remaining parameters. (We have found that the number of parameters is well-correlated with the decoding time for this set of experiments.) The initial RoBERTa-large model is the point *large, baseline* at the far right of the graph. The points labeled *large pr(dist)* represent various degrees of (distillation pruning, distillation continued training) of this initial model. The other points labeled *large pr(x-ent)* represent various degrees of (cross-entropy pruning, distillation continued training) of the same initial model. The distillation pruning does not provide a notable improvement over the cross-entropy driven pruning, unlike the case of distillation-driven continued training vs cross-entropy driven continued training. The point labeled *base+dist* is the RoBERTa-base model trained with the same distillation technique as our pruned models. It lies above and to the left of the envelope of *large pr(dist)* points, which suggests that the pruning-distillation process is not quite achieving its full potential. On the other hand, the pruning-distillation process offers much more flexibility of operating points, since the operating points we have presented here do not required the expensive masked language model pretraining but can instead leverage an existing pretraining, in this case RoBERTa-large as distributed.

The pruning-distillation process may also be applied to the RoBERTa-base model *base+dist*, and this is illustrated by the points *base+dist+pr(x-ent)* in Fig 4. These points lie even further above and to the left of the envelope of *base+dist* points, and point the way to even smaller and faster NQ models achievable by a combination of distillation and structured pruning. For comparison, the results we have obtained for DistilBERT (50.46) and TinyBERT (44.64) are at or below the bottom edge of this graph.

Averaging across five different initializations (random seeds) of gate parameters, a sample operating point for *large pr(dist)* has attention heads pruned by $60.0 \pm 1.0\%$, feed-forward activations pruned by $71.9 \pm 0.1\%$ yielding long-answer (LA) F1 of $68.2 \pm 0.2\%$ and short-answer (SA) F1 of $56.2 \pm 0.2\%$. Similarly, a sample operating point for *base+dist+pr(x-ent)* has attention heads pruned by $20.3 \pm 1.7\%$, feed-forward activations pruned by $17.7 \pm 0.4\%$, yielding long-answer (LA) F1 of $68.0 \pm 0.2\%$ and short-answer (SA) F1 of $57.0 \pm 0.2\%$.

5 Conclusions

We investigate various methods to prune transformer-based models, and evaluate the accuracy-speed tradeoff for this pruning. We find that both the attention heads and especially the feed forward layer can be pruned considerably with minimal loss of accuracy, while pruning the embedding/hidden dimension is ineffective because of a loss in accuracy.

We find that L_0 regularization pruning is particularly effective for pruning these two transformer components, compared to the more heuristic methods, Gain, and Leave-one-out. The pruned feed-forward layer and the pruned attention heads are easily combined. Especially after retraining, this combination yields a considerably faster question answering model with minimal loss in accuracy. One operating point nearly doubles the decoding speed on SQuAD 2.0, with a loss of less than 1.5 F1-points.

The same methods that worked with a BERT-based SQuAD 2.0 model also yield promising results when applied to a RoBERTa-based NQ model. The best performance is achieved by combining distillation with structured pruning. One operating point almost doubles the inference speed of RoBERTa-large based model for Natural Questions, while losing less than 0.5 F1-point on short answers.

These methods only require revisiting task-specific training data and do not require revisiting transformer pretraining. Furthermore our observation that the resulting transformer layers are non-identical may inform future efforts at pruning and distillation.

References

- 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, June. Association for Computational Linguistics.
- Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The state of sparsity in deep neural networks. *CoRR*, abs/1902.09574.
- Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, G P Shrivatsa Bhargav, Dinesh Garg, and Avirup Sil. 2019. Span selection pre-training for question answering.
- Saurabh Goyal, Anamitra Roy Choudhary, Venkatesan Chakaravarthy, Saurabh ManishRaje, Yogish Sabharwal, and Ashish Verma. 2020. Power-bert: Accelerating bert inference for classification tasks.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding.
- Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational Bayes. *International Conference on Learning Representations*.
- Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. *CoRR*.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations.
- Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E. Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.
- Christos Louizos, Max Welling, and Diederik P. Kingma. 2017. Learning sparse neural networks through l_0 regularization.

- Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
- Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *CoRR*, abs/1905.10650.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.
- Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Beijing, China, 22–24 Jun. PMLR.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.
- Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2019. Q-bert: Hessian based ultra low precision quantization of bert.
- Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks.
- Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Arivazhagan, Xin Li, and Amelia Archer. 2019. Small and practical bert models for sequence labeling.
- Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.
- Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *CoRR*, abs/1905.09418.
- Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2019. Structured pruning of large language models.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.
- Ze Yang, Linjun Shou, Ming Gong, Wutao Lin, and Daxin Jiang. 2019a. Model compression with multi-task knowledge distillation for web-scale question answering system.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.
- Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert.