All The Ways You Can Compress BERT

Nov 18, 2019

Model compression reduces redundancy in a trained neural network. This is useful, since BERT barely fits on a GPU (BERT-Large does not) and definitely won't fit on your smart phone. Improved memory and inference speed efficiency can also save costs at scale.

In this post I'll list and briefly taxonomize all the papers I've seen compressing BERT. Don't see yours? Feel free to shoot me an email.

Update 3/3/20: A survey paper of BERT compression methods has been released by Ganesh et al.

Methods

Pruning - Removes unnecessary parts of the network after training. This includes weight magnitude pruning, attention head pruning, layers, and others. Some methods also impose regularization during training to increase prunability (layer dropout).

Weight Factorization - Approximates parameter matrices by factorizing them into a multiplication of two smaller matrices. This imposes a low-rank constraint on the matrix. Weight factorization can be applied to both token embeddings (which saves a lot of memory on disk) or parameters in feed-forward / self-attention layers (for some speed improvements).

Knowledge Distillation - Aka "Student Teacher." Trains a much smaller Transformer from scratch on the pre-training / downstream-data. Normally this would fail, but utilizing soft labels from a fully-sized model improves optimization for unknown reasons. Some methods also distill BERT into different architectures (LSTMS, etc.) which have faster inference times. Others dig deeper into the teacher, looking not just at the output but at weight matrices and hidden activations.

Weight Sharing - Some weights in the model share the same value as other parameters in the model. For example, ALBERT uses the same weight matrices for every single layer of self-attention in BERT.

Quantization - Truncates floating point numbers to only use a few bits (which causes round-off error). The quantization values can also be learned either during or after training.

Pre-train vs. Downstream - Some methods only compress BERT w.r.t. certain downstream tasks. Others compress BERT in a way that is task-agnostic.

Papers

Bibtex

Paper	Prune	Factor	Distill	W. Sharing	Quant.	Pre-train	Downstream
Compressing BERT: Studying the Effects of Weight Pruning on Transfer Learning	☑					☑	☑
Are Sixteen Heads Really Better than One?	☑						☑
Pruning a BERT-based Question Answering Model	☑						☑
Reducing Transformer Depth on Demand with Structured Dropout	☑					☑	
Reweighted Proximal Pruning for Large-Scale Language Representation	☑					☑	
Structured Pruning of Large Language Models		☑					☑
ALBERT: A Lite BERT for Self-supervised Learning of Language Representations		☑		☑		☑	
Extreme Language Model Compression with Optimal Subwords and Shared Projections			☑			☑	
DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter			☑			☑	
Distilling Task-Specific Knowledge from BERT into Simple Neural Networks			☑				☑
Distilling Transformers into Simple Neural Networks with Unlabeled Transfer Data			☑				☑
Attentive Student Meets Multi-Task Teacher: Improved Knowledge Distillation for Pretrained Models			☑			Multi-task	☑
Patient Knowledge Distillation for BERT Model Compression			☑				☑
TinyBERT: Distilling BERT for Natural Language Understanding			☑			☑	☑

MobileBERT: Task-Agnostic Compression of BERT by Progressive Knowledge Transfer			☑			☑	
Q8BERT: Quantized 8Bit BERT					☑		☑
Q-BERT: Hessian Based Ultra Low Precision Quantization of BERT					☑		☑

Comparison of Results

We're just going to do our best here, and report whatever the papers claim. Mainly, we'll look at parameter reduction, inference speed-up[1], and accuracy.[2][3]

If you're looking for practical winners, I would go with ALBERT, DistilBERT, MobileBERT, Q-BERT, LayerDrop, RPP. You might be able to stack some of these methods.[4] But some of the pruning papers are more scientific than practical, so maybe check out those, too.

Paper	**Reduction**	**Of**	**Speed-up**	**Accuracy?**	**Comments**
Compressing BERT: Studying the Effects of Weight Pruning on Transfer Learning	30%	params	?	Same	Some interesting ablation experiments and fine-tuning analysis
Are Sixteen Heads Really Better than One?	50-60%	attn heads	1.2x	Same	
Pruning a BERT-based Question Answering Model	50%	attn Heads + FF	2x	-1.5 F1	
Reducing Transformer Depth on Demand with Structured Dropout	50-75%	layers	?	Same	
Reweighted Proximal Pruning for Large-Scale Language Representation	40-80%	params	?	Same	
Structured Pruning of Large Language Models	35%	params	?	Same	
ALBERT: A Lite BERT for Self-supervised Learning of Language Representations	90-95%	params	6-20x	Worse	Allows training larger models (BERT-xxlarge), so effectively 30% param reduction and 1.5x speedup with **better acc.**

Extreme Language Model Compression with Optimal Subwords and Shared Projections	80-98%	params	?	worse to much worse	
DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter	40%	params	2.5x	97%	🤗 Huggingface
Distilling Task-Specific Knowledge from BERT into Simple Neural Networks	99%	params	15x	ELMO equiv.	Distills into Bi-LSTMs
Distilling Transformers into Simple Neural Networks with Unlabeled Transfer Data	96%	params	?	?	Low-resource only
Attentive Student Meets Multi-Task Teacher: Improved Knowledge Distillation for Pretrained Models	90%	params	14x	better than Tang^	Distills into BiLSTMs.
Patient Knowledge Distillation for BERT Model Compression	50-75%	layers	2-4x	Worse	But better than vanilla KD
TinyBERT: Distilling BERT for Natural Language Understanding	87%	params	9.4x	96%	
MobileBERT: Task-Agnostic Compression of BERT by Progressive Knowledge Transfer	77%	params	4x	competitive	
Q8BERT: Quantized 8Bit BERT	75%	bits	?	negligible	"Need hardware to show speed-ups" but I don't think anyone has it
Q-BERT: Hessian Based Ultra Low Precision Quantization of BERT	93%	bits	?	"at most 2.3% worse"	^ probably same

If you found this post useful, please consider citing it as:

```
@misc{gordon_2019,
    author = "Mitchell A. Gordon",
    title = "All The Ways You Can Compress BERT",
    year = "2019",
    howpublished="http://mitchgordon.me/machine/learning/2019/11/18/all-the-wa
}
```

Bonus Papers / Blog Posts

Sparse Transformer: Concentrated Attention Through Explicit Selection

Lightweight and Efficient Neural Natural Language Processing with Quaternion Networks

Adaptively Sparse Transformers

Compressing BERT for Faster Prediction

Update 11/19/19: Bibtex and bonus papers

Update 11/24/19: Added section with comparison of results

Update 11/25/19: Added “Attentive Student Meets…”

1. Note that not all compression methods make models faster. Unstructured pruning is notoriously difficult to speed-up via GPU parallelization. One of the papers claims that in Transformers, the computation time is dominated by the softmax computation, rather than matrix multiplication. ↩
2. It would be nice if we could come up with a single number to capture what we really care about. Like F1. ↩
3. Some of these percentages are measured against BERT-Large instead of BERT-Base, FYI. ↩
4. How different compression methods interact is an open research question. ↩

Mitchell A. Gordon

Mitchell A. Gordon
mitchell.gordon95@gmail.com

mitchellgordon95

Thinking Thoughts