– IN5550 –
Neural Methods in Natural Language Processing

Sustainability 2: Towards Green(er) NLP

Lilja Øvrelid

(With thanks to Jeremy Barnes)

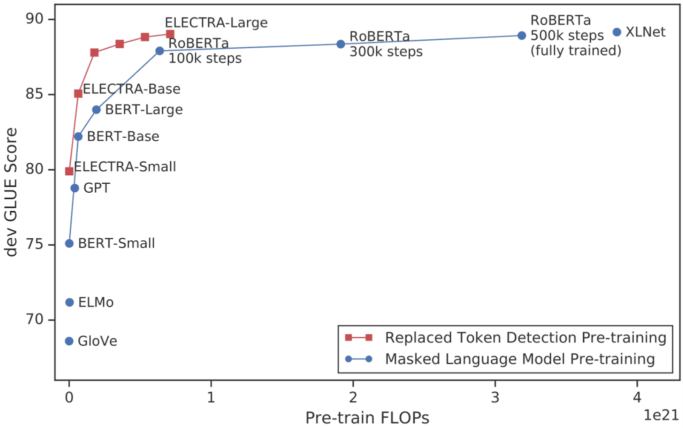
Towards Green NLP?

How can we mitigate the negative effects of large LMs?

- Enhance reporting of computational budgets
- Efficiency as a core evaluation metric

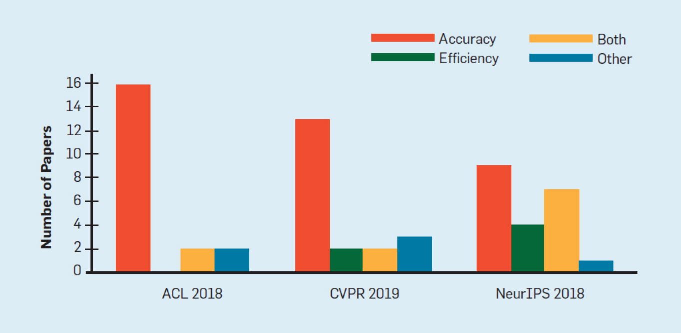
* from Schwartz et al (2020): Green AI

Improved reporting



* from Clark et al (2020): ELECTRA

Beyond accuracy



* from Schwartz et al (2020): Green AI

Efficient methods

What is made more efficient?

- Training
- Inference
- Model selection

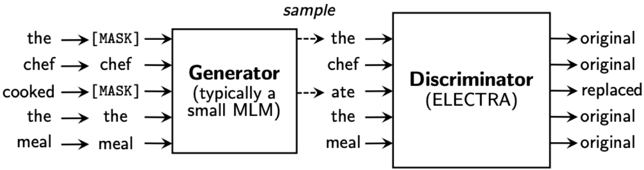
How do we measure it?

- Space
- Time
- Energy

Efficient training

ELECTRA (Clark et al, 2020)

- Modifies pre-training objective
- Trained to distinguish "real" input tokens vs "fake" input tokens generated by another neural network
- Strong results even when trained on a single GPU.



* from Clark et al (2020): ELECTRA

Sharing is caring

To avoid retraining lots of models

- We can share the trained models
- Nordic Language Processing Laboratory (NLPL) is a good example
- Huggingface
- But it's important to get things right
  - METADATA!!!
  - same format for all models

Efficient inference

What if we can reduce the size of these giant models?

Efficient inference

What if we can reduce the size of these giant models?

- Often, overparameterized transformer models lead to better performance, even with less data

What if we can reduce the size of these giant models?

- Often, overparameterized transformer models lead to better performance, even with less data
- Lottery-ticket hypothesis: for large enough models, there is a small chance that random initialization will lead to a submodel that already has good weights for the task

Efficient inference

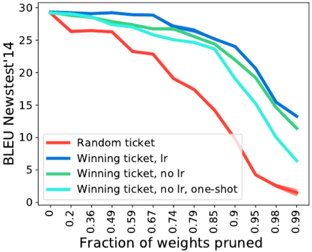
What if we can reduce the size of these giant models?

- Often, overparameterized transformer models lead to better performance, even with less data
- Lottery-ticket hypothesis: for large enough models, there is a small chance that random initialization will lead to a submodel that already has good weights for the task
- But interestingly, you can often remove a large number of the parameters for only a small decrease in performance

Efficient inference

What if we can reduce the size of these giant models?

- Often, overparameterized transformer models lead to better performance, even with less data
- Lottery-ticket hypothesis: for large enough models, there is a small chance that random initialization will lead to a submodel that already has good weights for the task
- But interestingly, you can often remove a large number of the parameters for only a small decrease in performance



Reduce model size?

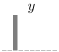
Reduce model size?

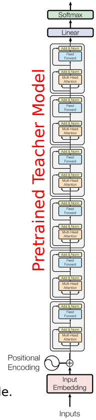
		Compression	Performance	Speedup	Model	Evaluation
	BERT-base (Devlin et al., 2019)	$\times 1$	100%	$\times 1$	BERT$_{12}$	All GLUE tasks, SQuAD
	BERT-small	$\times 3.8$	91%	–	BERT$_{4}\dagger$	All GLUE tasks
Distillation	DistilBERT (Sanh et al., 2019)	$\times 1.5$	90%§	$\times 1.6$	BERT$_{6}$	All GLUE tasks, SQuAD
	BERT$_6$-PKD (Sun et al., 2019a)	$\times 1.6$	98%	$\times 1.9$	BERT$_{6}$	No WNLI, CoLA, STS-B; RACE
	BERT$_3$-PKD (Sun et al., 2019a)	$\times 2.4$	92%	$\times 3.7$	BERT$_{3}$	No WNLI, CoLA, STS-B; RACE
	Aguilar et al. (2019), Exp. 3	$\times 1.6$	93%	–	BERT$_{6}$	CoLA, MRPC, QQP, RTE
	BERT-48 (Zhao et al., 2019)	$\times 62$	87%	$\times 77$	BERT$_{12}^{*\dagger}$	MNLI, MRPC, SST-2
	BERT-192 (Zhao et al., 2019)	$\times 5.7$	93%	$\times 22$	BERT$_{12}^{*\dagger}$	MNLI, MRPC, SST-2
	TinyBERT (Jiao et al., 2019)	$\times 7.5$	96%	$\times 9.4$	BERT$_{4}^{\dagger}$	No WNLI; SQuAD
	MobileBERT (Sun et al., 2020)	$\times 4.3$	100%	$\times 4$	BERT$_{24}^{\dagger}$	No WNLI; SQuAD
	PD (Turc et al., 2019)	$\times 1.6$	98%	$\times 2.5^{\ddagger}$	BERT$_{6}^{\dagger}$	No WNLI, CoLA and STS-B
	WaLDORf (Tian et al., 2019)	$\times 4.4$	93%	$\times 9$	BERT$_{8}^{\dagger\Vert}$	SQuAD
	MiniLM (Wang et al., 2020b)	$\times 1.65$	99%	$\times 2$	BERT$_{6}$	No WNLI, STS-B, MNLI$_{mm}$; SQuAD
	MiniBERT(Tsai et al., 2019)	$\times 6^{**}$	98%	$\times 27^{**}$	mBERT$_{3}^{\dagger}$	CoNLL-18 POS and morphology
	BiLSTM-soft (Tang et al., 2019)	$\times 110$	91%	$\times 434^{\ddagger}$	BiLSTM$_{1}$	MNLI, QQP, SST-2
Quanti-zation	Q-BERT-MP (Shen et al., 2019)	$\times 13$	98%¶	–	BERT$_{12}$	MNLI, SST-2, CoNLL-03, SQuAD
	BERT-QAT (Zafrir et al., 2019)	$\times 4$	99%	–	BERT$_{12}$	No WNLI, MNLI; SQuAD
	GOBO (Zadeh and Moshovos, 2020)	$\times 9.8$	99%	–	BERT$_{12}$	MNLI
Pruning	McCarley et al. (2020), ff2	$\times 2.2^{\ddagger}$	98%‡	$\times 1.9^{\ddagger}$	BERT$_{24}$	SQuAD, Natural Questions
	RPP (Guo et al., 2019)	$\times 1.7^{\ddagger}$	99%‡	–	BERT$_{24}$	No WNLI, STS-B; SQuAD
	Soft MvP (Sanh et al., 2020)	$\times 33$	94%¶	–	BERT$_{12}$	MNLI, QQP, SQuAD
	IMP (Chen et al., 2020), rewind 50%	$\times 1.4$–2.5	94–100%	–	BERT$_{12}$	No MNLI-mm; SQuAD
Other	ALBERT-base (Lan et al., 2020)	$\times 9$	97%	–	BERT$_{12}^{\dagger}$	MNLI, SST-2
	ALBERT-xxlarge (Lan et al., 2020)	$\times 0.47$	107%	–	BERT$_{12}^{\dagger}$	MNLI, SST-2
	BERT-of-Theseus (Xu et al., 2020)	$\times 1.6$	98%	$\times 1.9$	BERT$_{6}$	No WNLI
	PoWER-BERT (Goyal et al., 2020)	N/A	99%	$\times 2$–4.5	BERT$_{12}$	No WNLI; RACE

From Rogers et al. (2020) A Primer in BERTology: What We Know About How BERT Works.

Reduce model size?

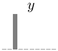
Model distillation

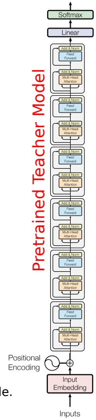


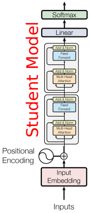


This is an example.

Model distillation

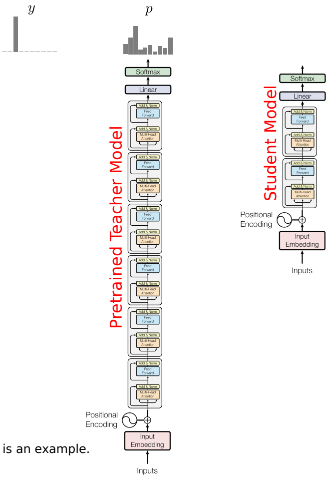






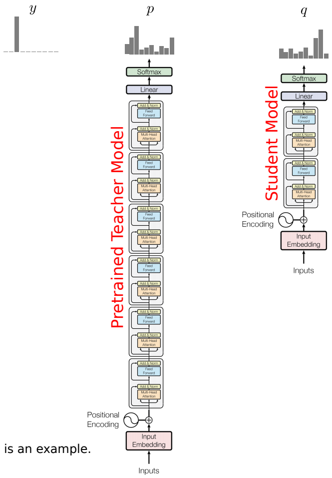
This is an example.

Model distillation



This is an example.

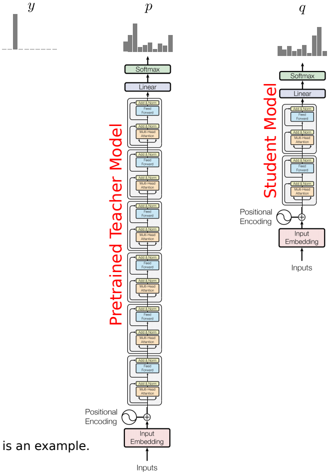
Model distillation



This is an example.

Model distillation

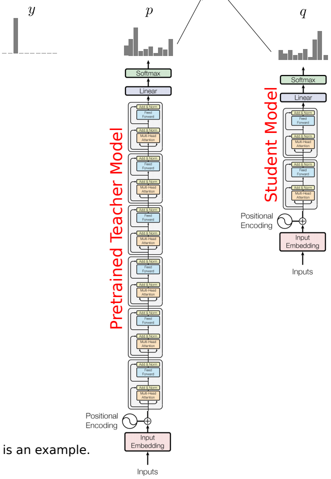
$$KL(p||q) = \sum_i p_i * log(p_i) - \sum_i p_i * log(q_i)$$



This is an example.

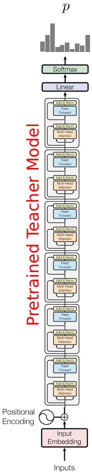
Model distillation

$$KL(p||q) = \sum_i p_i * log(p_i) - \sum_i p_i * log(q_i)$$



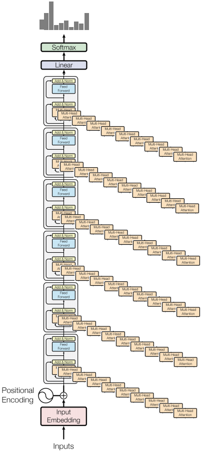
This is an example.

Head pruning



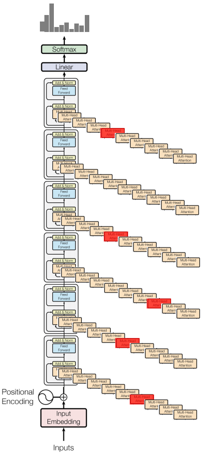
Head pruning

Performance = 93.7



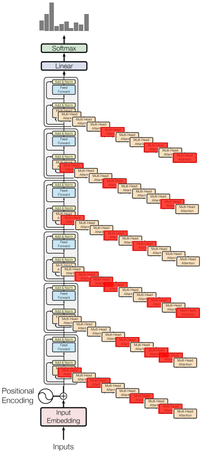
Head pruning

Performance = 93.7



Head pruning

Performance = 93.0



But how can we NLPers contribute to sustainability?

But how can we NLPers contribute to sustainability?

- When possible use pre-trained models

But how can we NLPers contribute to sustainability?

- When possible use pre-trained models
- If you train a strong model, similarly make it available to the community.

But how can we NLPers contribute to sustainability?

- When possible use pre-trained models
- If you train a strong model, similarly make it available to the community.
- Try to reduce the amount of hyperparameter tuning we do (for example, by working with models that are more robust to hyperparameters)

But how can we NLPers contribute to sustainability?

- When possible use pre-trained models
- If you train a strong model, similarly make it available to the community.
- Try to reduce the amount of hyperparameter tuning we do (for example, by working with models that are more robust to hyperparameters)
- Improved reporting, take into account efficiency

But how can we NLPers contribute to sustainability?

- When possible use pre-trained models
- If you train a strong model, similarly make it available to the community.
- Try to reduce the amount of hyperparameter tuning we do (for example, by working with models that are more robust to hyperparameters)
- Improved reporting, take into account efficiency
  - budget/performance curves

But how can we NLPers contribute to sustainability?

- When possible use pre-trained models
- If you train a strong model, similarly make it available to the community.
- Try to reduce the amount of hyperparameter tuning we do (for example, by working with models that are more robust to hyperparameters)
- Improved reporting, take into account efficiency
  - budget/performance curves

Model	Train / Infer FLOPs	Speedup	Params	Train Time + Hardware	GLUE
ELMo	3.3e18 / 2.6e10	19x / 1.2x	96M	14d on 3 GTX 1080 GPUs	71.2
GPT	4.0e19 / 3.0e10	1.6x / 0.97x	117M	25d on 8 P6000 GPUs	78.8
BERT-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	75.1
BERT-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	82.2
ELECTRA-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	79.9
50% trained	7.1e17 / 3.7e9	90x / 8x	14M	2d on 1 V100 GPU	79.0
25% trained	3.6e17 / 3.7e9	181x / 8x	14M	1d on 1 V100 GPU	77.7
12.5% trained	1.8e17 / 3.7e9	361x / 8x	14M	12h on 1 V100 GPU	76.0
6.25% trained	8.9e16 / 3.7e9	722x / 8x	14M	6h on 1 V100 GPU	74.1
ELECTRA-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	85.1