HULK: An Energy Efficiency Benchmark Platform for Responsible Natural Language Processing

Xiyou Zhou, Zhiyu Chen, Xiaoyong Jin, William Yang Wang
Department of Computer Science, University of California Santa Barbara
{xiyou, zhiyuchen, x_jin, william}@cs.ucsb.edu

Abstract

Computation-intensive pretrained models have been taking the lead of many natural language processing benchmarks such as GLUE (Wang et al., 2018). However, energy efficiency in the process of model training and inference becomes a critical bottleneck. We introduce HULK, a multi-task energy efficiency benchmarking platform for responsible natural language processing. With HULK, we compare pretrained models' energy efficiency from the perspectives of time and cost. Baseline benchmarking results are provided for further analysis. The fine-tuning efficiency of different pretrained models can differ significantly among different tasks, and fewer parameter number does not necessarily imply better efficiency. We analyzed such a phenomenon and demonstrated the method for comparing the multi-task efficiency of pretrained models. Our platform is available at `https://hulkbenchmark.github.io/`.

1 Introduction

Environmental concerns of machine learning research have been rising as the carbon emission of specific tasks like neural architecture search reached an exceptional "ocean boiling" level (Strubell et al., 2019). Increased carbon emission has been one of the key factors to aggravate global warming [1]. Research and development processes like parameter search further increase the environmental impact. When using cloud-based machines, the environmental impact is strongly correlated with the financial cost.

The recent emergence of leaderboards such as SQuAD (Rajpurkar et al., 2016), GLUE (Wang et al., 2018), and SuperGLUE (Wang et al., 2019) has greatly boosted the development of advanced models in the NLP community. Pretrained models have proven to be the key ingredient for achieving state-of-the-art in conventional metrics. However, such models can be costly to train. For example, XLNet-Large (Yang et al., 2019) was trained on 512 TPU v3 chips for 500K steps, which costs around 61,440 dollars[2], let alone staggeringly large carbon emission.

Moreover, despite impressive performance gain, the fine-tuning and inference efficiency of NLP models remain under-explored. As recently mentioned in a tweet[3], the popular AI text adventure game *AI Dungeon* has reached 100 million inferences. The energy efficiency of inference cost could be critical to both business planning and environmental impact.

Previous work (Schwartz et al., 2019; Dodge et al., 2019) on this topic proposed new metrics like FPO (floating-point operations) and other practices to report experimental results based on computing budget. Other benchmarks like (Coleman et al., 2017) and (Mattson et al., 2019) compare the efficiency of models on the classic reading comprehension task SQuAD and machine translation tasks. However, there has not been any concrete or practical reference for accurate estimation on NLP model pretraining, fine-tunning, and inference considering multi-task energy efficiency.

Energy efficiency can be reflected in many metrics, including carbon emission, electricity usage, time consumption, number of parameters, and FPO, as shown in (Schwartz et al., 2019). Carbon emission and electricity are intuitive measures yet either hard to track or hardware-dependent. The number of model parameters does not reflect the actual cost for model training and inference. FPO

[1]Source: `https://climate.nasa.gov/causes/`

[2]Source: The Staggering Cost of Training SOTA AI Models by Synced Global

[3]Source: Nick Walton's Tweet on Passing 100 Million Inferences on AI Dungeon.

Model	Hardware	Time	Cost	Params
BERT$_{\text{BASE}}$ (Devlin et al., 2018)	4 TPU v2 Pods	4 days	\$1,728	108M
BERT$_{\text{LARGE}}$ (Devlin et al., 2018)	16 TPU v2 Pods	4 days	\$6,912	334M
XLNet$_{\text{BASE}}$ (Yang et al., 2019)	–	–	–	117M
XLNet$_{\text{LARGE}}$ (Yang et al., 2019)	512 TPU v3	2.5 days	\$61,440	361M
RoBERTa$_{\text{BASE}}$ (Liu et al., 2019)	1024 V100 GPUs	1 day	\$75,203	125M
RoBERTa$_{\text{LARGE}}$ (Liu et al., 2019)	1024 V100 GPUs	1 day	\$75,203	356M
ALBERT$_{\text{BASE}}$ (Lan et al., 2019)	64 TPU v3	–	–	12M
ALBERT$_{\text{XXLARGE}}$ (Lan et al., 2019)	1024 TPU v3	32 hours	\$65,536	223M
DistilBERT* (Sanh et al., 2019)	8×16G V100 GPU	90 hours	\$2,203	66M
ELECTRA$_{\text{SMALL}}$ (Clark et al., 2020)	1 V100 GPU	96 hours	\$294	14M
ELECTRA$_{\text{BASE}}$ (Clark et al., 2020)	16 TPU v3	96 hours	\$3,072	110M

Table 1: Pretraining costs of baseline models. Hardware and pretraining time were collected from original papers, with which costs were estimated with the current TPU price at \$8 per hour with 4 core TPU v3 chips and V100 GPU at \$3.06 per hour. DistilBERT model was trained upon a pretrained BERT model. Parameter numbers are estimated using the pretrained models implemented in the Transformers (`https://github.com/huggingface/transformers`) library (Wolf et al., 2019), shown in millions. The RoBERTa model was trained for 100K steps.

is steady for models but cannot be directly used for cost estimation. Here, to provide a practical reference for model selection on real applications, especially model development outside academia, we keep track of the time consumption and actual financial cost for comparison. Cloud-based machines are employed for budget estimation as they are easily accessible and consistent in hardware configuration, price, and performance. In the following sections, we would use "time" and "cost" to denote the time elapsed and the actual budget in model pretraining, training, and inference.

In most NLP pretrained model settings, there are three phases: pretraining, fine-tuning, and inference. If a model is trained from scratch, we consider such a model has no pretraining phase but is fine-tuned from scratch. Typically pretraining takes several days and hundreds of dollars, according to Table 1. Fine-tuning takes a few minutes to hours, costing much less than the pretraining phase. Inference takes several milliseconds to seconds, similarly costing much less than the fine-tuning phase. Meanwhile, pretraining is done before fine-tuning once for all, while fine-tuning could be performed multiple times as training data updates. Inference is expected to be called numerous times for downstream applications. Such characteristics make it a natural choice to separate different phases during benchmarking.

Our HULK benchmark, as shown in Figure 1, utilizes several classic datasets that have been widely adopted in the community as benchmarking tasks to benchmark energy efficiency. The benchmark compares pretrained models in a multi-task fashion. The tasks include natural language inference task MNLI (Williams et al., 2017), sentiment analysis task SST-2 (Socher et al., 2013) and Named Entity Recognition Task CoNLL-2003 (Sang and De Meulder, 2003). Such tasks are selected to provide a thorough comparison of end-to-end energy efficiency in pretraining, fine-tuning, and inference.

With the HULK benchmark, we quantify the energy efficiency of model pretraining, fine-tuning, and inference phase by comparing the time and cost they require to reach a certain overall task-specific performance level on selected datasets. The design principle and benchmarking process are detailed in section 2. We also explore the relation between model parameters and fine-tuning efficiency and demonstrate energy efficiency consistency between different pretrained models' tasks.

2 Benchmark Overview

For the pretraining phase, the benchmark is designed to favor energy-efficient models in terms of time and cost that each model takes to reach specific multi-task performance pretrained from scratch. For example, we keep track of the time and cost of a BERT model in the following way: After every thousand pretraining steps, we clone the model for fine-tuning and see if the final performance can reach our cut-off performance on different tasks. When the level is reached, time and cost for pretraining are used for comparison. Mod-

	CoNLL 2003	MNLI	SST-2
Train Size	14,041	392,702	67,349
Dev Size	3,250	19,647	872
Cut-off	91	85	90
Metric	F1	Acc	Acc
SOTA	93.5	91.85	97.4

Table 2: Dataset Information

els faster or cheaper to pretrain are recommended.

We consider the time and cost each model takes to reach specific multi-task performance fine-tuned from given pretrained models for the fine-tuning phase. For each task with different difficulty and instance numbers, the fine-tuning characteristics may differ a lot. When pretrained models are used to deal with a non-standard downstream task, especially ad hoc application in industry, the task's fine-tuning time and cost cannot be estimated directly from any other standard task. Therefore, it is essential to compare the multi-task efficiency for model choice.

For the inference phase, each model's time and cost for making inference on a single instance on multiple tasks are compared similarly to the fine-tuning phase. Specially, we estimate the time elapsed for each inference by averaging thousands of inference samples.

2.1 Dataset Overview

The datasets we used are widely adopted in the NLP community. Quantitative details of datasets can be found in Table 2. The selected tasks are shown below:

CoNLL 2003 The Conference on Computational Natural Language Learning (CoNLL-2003) shared task concerns language-independent named entity recognition (Sang and De Meulder, 2003). The task concentrates on four types of named entities: persons, locations, organizations, and other miscellaneous entities. Here we only use the English dataset. The English data is a collection of news wire articles from the Reuters Corpus. The result is reflected as F1 score considering the label accuracy and recall on the dev set.

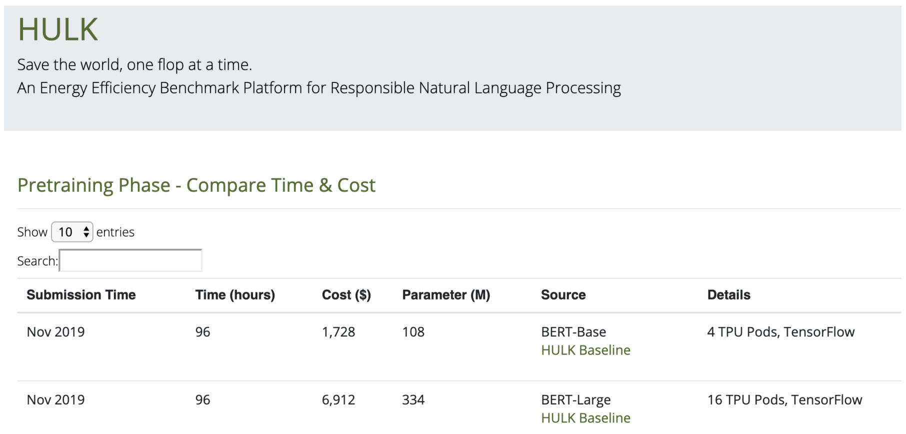
MNLI The Multi-Genre Natural Language Inference Corpus (Williams et al., 2017) is a crowdsourced collection of sentence pairs with textual entailment annotations. Given a premise sentence and a hypothesis sentence, the task is to predict whether the premise entails the hypothesis (entailment), contradicts the hypothesis (contradiction), or neither (neutral). The premise sentences are gathered from ten different sources, including transcribed speech, fiction, and government reports. The accuracy score is reported as the average of performance on matched and mismatched dev sets.

SST-2 The Stanford Sentiment Treebank (Socher et al., 2013) consists of sentences from movie reviews and human annotations of their sentiment. The task is to predict the sentiment of a given sentence. Following the setting of GLUE, we also use the two-way (positive/negative) class split and use only sentence-level labels.

The tasks are selected based on how representative the dataset is. CoNLL 2003 has been a widely used dataset for named entity recognition and requires the output of token-level labeling. NER is a core NLP task, and CoNLL 2003 has been a classic dataset in this area. SST-2 and MNLI are part of the GLUE benchmark, representing sentence-level labeling tasks. SST-2 has been frequently used in sentiment analysis across different generations of models. MNLI is a newly introduced large dataset for natural language inference. The training time for MNLI is relatively long, and the task requires a lot more training instances. We select the three tasks for a diverse yet practical benchmark for pretrained models without constraining the models on sentence-level classification tasks. Besides, their efficiency differs significantly in the fine-tuning and inference phase. Such a difference can still be reflected in the final score after normalization, as shown in Table 3. Provided with more computing resources, we can bring in more datasets for even more thorough benchmarking in the future. We illustrate the evaluation criteria in the following subsection.

2.2 Evaluation Criteria

In machine learning model training and inference, slight parameter change can subtly impact the final result. To make a practical reference for pretrained model selection, we compare models' end-to-end performance concerning the pretraining time and cost, fine-tuning time and cost, inference time and cost following the setting of (Coleman et al., 2017).


HULK

Save the world, one flop at a time.

An Energy Efficiency Benchmark Platform for Responsible Natural Language Processing

Pretraining Phase - Compare Time & Cost

Show 10 entries

Search:

Submission Time	Time (hours)	Cost ($)	Parameter (M)	Source	Details
Nov 2019	96	1,728	108	BERT-Base HULK Baseline	4 TPU Pods, TensorFlow
Nov 2019	96	6,912	334	BERT-Large HULK Baseline	16 TPU Pods, TensorFlow

Figure 1: Screenshot of the leaderboard of website.

Datasets	CoNLL 2003		SST-2		MNLI		
Model	Time	Score	Time	Score	Time	Score	Overall Score
$\text{BERT}_{\text{BASE}}$	43.43	2.08	207.15	0.45	N/A	0.00	2.53
$\text{BERT}_{\text{LARGE}}$	90.26	1.00	92.45	1.00	9,106.72	1.00	3.00
$\text{XLNet}_{\text{BASE}}$	67.14	1.34	102.45	0.90	7,704.71	1.18	3.42
$\text{XLNet}_{\text{LARGE}}$	243.00	0.37	367.11	0.25	939.62	9.69	10.31
$\text{RoBERTa}_{\text{BASE}}$	70.57	1.28	38.45	2.40	274.87	7.14	10.82
$\text{RoBERTa}_{\text{LARGE}}$	155.43	0.58	57.65	1.60	397.12	22.93	25.11
$\text{ALBERT}_{\text{BASE}}$	340.64	0.26	2,767.90	0.03	N/A	0.00	0.29
$\text{ALBERT}_{\text{LARGE}}$	844.85	0.11	3,708.49	0.02	N/A	0.00	0.13

Table 3: Multi-task Baseline Fine-tuning Costs. Time is given in seconds and score is computed by the division of $\text{Time}_{\text{BERT}_{\text{LARGE}}}/\text{Time}_{\text{model}}$.The experiments are conducted on a single RTX 2080 Ti GPU following the evaluation criterion. The overall score is computed by summing up the scores of each task. We also use the cost to compute a new score for each task for cost-based leaderboards and summarize similarly. "N/A" means fail to reach the given performance after five epochs.

For the pretraining phase, we design the protocol to explore how much computing resource is required to reach specific final multi-task performance via fine-tuning after the pretraining. Therefore, during model pretraining, after every thousand pretraining steps, we use the current pretrained model for fine-tuning and see if the fine-tuned model can reach our cut-off performance. When it does, we count the time and cost in the pretraining process for benchmarking and analysis.

For the fine-tuning phase, we want to compare the general efficiency of the pretrained model reaching cut-off performance on the selected dataset.

During fine-tuning, we evaluate the current fine-tuned model on the development set after a certain small number of fine-tuning steps. When the performance reaches our cut-off performance, we count the time and cost in this fine-tuning process for benchmarking and analysis. To be specific, for a single pretrained model, the efficiency score on different tasks is defined as the sum of normalized time and cost. Here we normalize the time and cost because they vary dramatically between tasks. To simplify the process, we compute the ratio of $\text{BERT}_{\text{LARGE}}$'s time and cost to that of each model as the normalized measure, as shown in Table 3 and Table 4.

We follow the fine-tuning principles for the inference phase, and we use the time and cost of inference for benchmarking. The models we used for inference experiments are fine-tuned in the last part. Each of the benchmarking results was calculated using the average of time and cost over one thousand samples.

2.3 Performance Cut-off Selection

The selection of performance cut-off could be critical because we consider certain models to be

Datasets	CoNLL 2003		SST-2		MNLI		
Model	Time	Score	Time	Score	Time	Score	Overall Score
BERT$_{\text{BASE}}$	2.68	3.18	2.70	3.13	2.67	3.19	9.5
BERT$_{\text{LARGE}}$	8.51	1.00	8.46	1.00	8.53	1.00	3.00
XLNet$_{\text{BASE}}$	5.16	1.65	5.01	1.69	5.10	1.67	5.01
XLNet$_{\text{LARGE}}$	14.84	0.57	14.69	0.58	15.27	0.56	1.71
RoBERTa$_{\text{BASE}}$	2.65	3.21	2.68	3.16	2.70	3.16	9.53
RoBERTa$_{\text{LARGE}}$	8.35	1.02	8.36	1.01	8.70	0.98	3.01
ALBERT$_{\text{BASE}}$	2.65	3.21	2.68	3.16	2.72	3.14	9.51
ALBERT$_{\text{LARGE}}$	8.49	1.00	8.44	1.00	8.78	0.97	2.97

Table 4: Multi-task Baseline Inference Costs. Time is given in milliseconds and score is computed by the division of $\text{Time}_{\text{BERT}_{\text{LARGE}}}/\text{Time}_{\text{model}}$.The experiments are conducted on a single RTX 2080 Ti GPU following the evaluation criterion similar to the fine-tuning part. The inference time between tasks is more consistent compared to the fine-tuning phase.

"qualified" after reaching specific performance on the development set. Meanwhile, particular tasks can reach a "sweet point" where after a relatively smaller amount of training time, the model reaches performance close to the final converged performance with a negligible difference. The cut-off must be high enough to make sure any model that surpasses the threshold can be competent for the task. On the other hand, if the cut-off is too high, we will not have enough data points to evaluate the model's multi-task performance.

Here, our cut-offs were selected by observing the recent state-of-the-art model's performance on the selected dataset for the task[4]. A wise choice would be choosing the performance of some classic methods like LSTM-CRF or Bi-LSTM models as the cut-off. In this way, we can easily compare the efficiency of most models with a solid performance bar.

2.4 Submission to Benchmark

Submissions can be made to our benchmark through sending code and result to our HULK benchmark CodaLab competition[5] following the guidelines in both our FAQ part of website and competition introduction. We require the submissions to include detailed end-to-end model training information, including model run time, cost (cloud-based machine only), parameter number, and part of the development set output for result validation. A training / fine-tuning log, including time consumption and dev set performance after certain steps, is also required. For inference, development set output, time consumption, and hardware/software details should be provided. For model reproducibility, source code is also required.

3 Baseline Settings and Analysis

We adopt the reported resource requirements in the original papers as the pretraining phase baselines for computation-intensive tasks.

For fine-tuning and inference phase, we conduct extensive experiments on given hardware (RTX 2080Ti GPU) with different model settings as shown in Table 3 and Table 4. We also collect the development set performance with time in fine-tuning to investigate how the model is fine-tuned for different tasks.

In our fine-tuning setting, we are given a specific hardware and software configuration. We adjust the hyper-parameter using grid search to minimize the time fine-tuning towards cut-off performance. For example, we choose the proper batch size and learning rate for BERT$_{\text{BASE}}$ to make sure the model converges and can reach expected performance as soon as possible with parameter searching.

As shown in Figure 2, the fine-tuning performance curve differs a lot among pretrained models. The x-axis denoting time consumed is shown in log-scale for a better comparison of different models. None of the models take the lead in all tasks. However, if two pretrained models are in the same family, such as BERT$_{\text{BASE}}$ and BERT$_{\text{LARGE}}$, the model with a smaller number of parameters tend to converge a bit faster than the other in the NER and SST-2 task. In the MNLI task, such a trend does

[4]For example, we referred to the performance data points on Papers With Code for candidates.

[5]The CodaLab competition is available on the website.

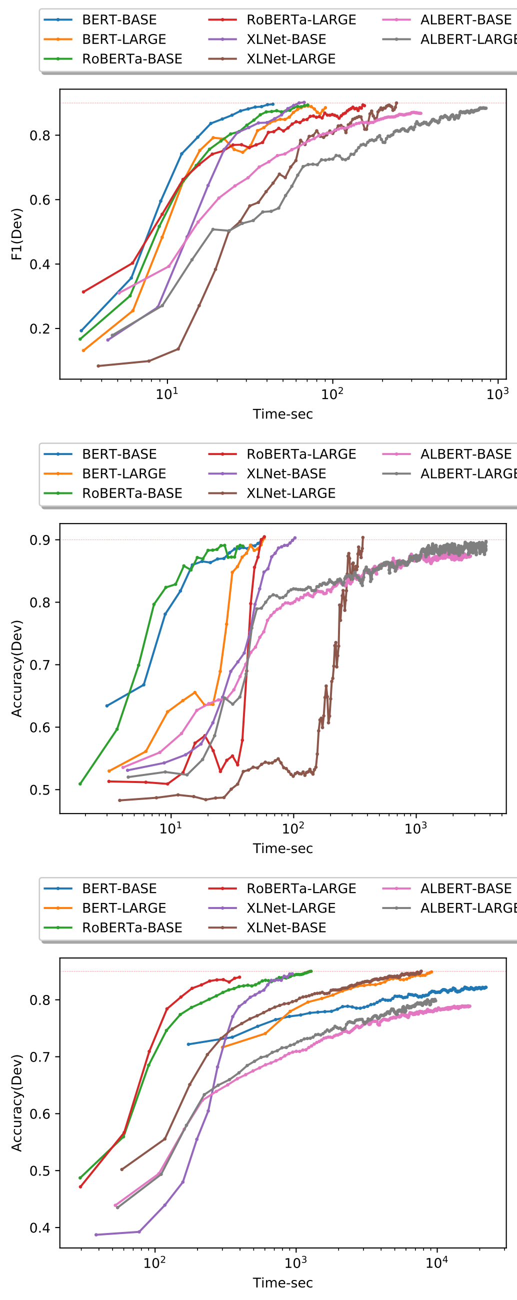


Figure 2: The comparison between different pretrained models for CoNLL 2003, SST-2, and MNLI datasets trained on a single RTX 2080Ti GPU. The curves are smoothed by computing the average with two adjacent data points. The experiments are conducted by selecting hyper-parameters to minimize the time consumption, yet making sure the model can converge after a certain amount of time. Results are demonstrated using performance on the development set after being fine-tuned for specific steps on the training dataset.

not apply because of the increased difficulty level and the number of training instances, which favors a larger model capacity.

Even though ALBERT model has a lot fewer parameters than BERT, according to Table 1, the ALBERT model's fine-tuning time is significantly more than BERT models because ALBERT uses large hidden size and more expensive matrix computation. The parameter sharing technique makes it harder to fine-tune the model. RoBERTa$_{\text{LARGE}}$ model is relatively stable in all tasks.

4 Related Work

GLUE benchmark (Wang et al., 2018) is a popular multi-task benchmarking and diagnosis platform providing score evaluating multi-task NLP models considering multiple single-task performances. SuperGLUE (Wang et al., 2019) further develops the task and enriches the evaluation dataset, making tasks more challenging. These multi-task benchmarks do not consider computation efficiency but innovates the development of pretrained models.

MLPerf (Mattson et al., 2019) compares training and inference efficiency from a hardware perspective, providing helpful resources on hardware selection and model training. The benchmark focused on several typical applications, including image classification and machine translation. However, it does not separate different training phases, thus making it hard to find the reference for fine-tuning only applications.

Previous work (Schwartz et al., 2019; Dodge et al., 2019) on related topic working towards "Green AI" proposes new metrics like FPO and new principle in efficiency evaluation. We further make more detailed and practical contributions to model energy efficiency benchmarking.

Other work like DAWNBenchmark (Coleman et al., 2017) looks into the area of end-to-end model efficiency comparison for both computer vision and NLP task SQuAD. The benchmark is very detailed and intuitive. However, it does not compare multi-task efficiency performance and covered only one NLP task. Similar to MLPerf, it does not separate fine-tuning efficiency from training efficiency.

The *Efficient NMT* shared task of The 2nd Workshop on Neural Machine Translation and Generation proposed an efficiency track to compare the inference time of neural machine translation models. Our platform covers more phases and supports multi-task comparison.

5 Conclusion

We developed the HULK platform focusing on the energy efficiency benchmarking of NLP models based on their end-to-end performance on selected NLP tasks. The HULK platform compares models in the pretraining, fine-tuning, and inference phase, making it clear to follow and propose more training-efficient and inference-efficient models. We have compared the fine-tuning efficiency of given models during baseline testing and demonstrated more parameters lead to slower fine-tuning when using the same model design but do not hold when the model architecture changes.We expect more submissions in the future to flourish and enrich our benchmark.

Acknowledgments

This work is supported by the Institute of Energy Efficiency (IEE) at UCSB's seed grant in Summer 2019 to improve AI and machine learning's energy efficiency.[6].

[6]https://iee.ucsb.edu/news/making-ai-more-energy-efficient

References

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Cody Coleman, Deepak Narayanan, Daniel Kang, Tian Zhao, Jian Zhang, Luigi Nardi, Peter Bailis, Kunle Olukotun, Chris Ré, and Matei Zaharia. 2017. Dawnbench: An end-to-end deep learning benchmark and competition. *Training*, 100(101):102.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A Smith. 2019. Show your work: Improved reporting of experimental results. *arXiv preprint arXiv:1909.03004*.

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. *arXiv preprint arXiv:1909.11942*.

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. *arXiv preprint arXiv:1907.11692*.

Peter Mattson, Christine Cheng, Cody Coleman, Greg Diamos, Paulius Micikevicius, David Patterson, Hanlin Tang, Gu-Yeon Wei, Peter Bailis, Victor Bittorf, et al. 2019. Mlperf training benchmark. *arXiv preprint arXiv:1910.01500*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2019. Green ai. *arXiv preprint arXiv:1907.10597*.

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.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint arXiv:1905.00537*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

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.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.